“iAM Smart” API Specification

Version: *2.5.2*

FEBRUARY 2026

Document Revision History

Ver. No.	Date	Section Affected	Summary of Changes
1.0.0	26 July 2019		Initial release
1.0.1	6 September 2019	Section 4 – 6	Incorporate comments received for the initial release
1.1.0	23 October 2019	Section 4.3	Add new APIs for PDF digital signing
		Section 4.4	Rename step-up authentication to re-authentication.
		Section 4.5	Add new sections to describe anonymous form filling workflow
		Section 4.6	Add new sections to describe anonymous digital signing
		Section 5.4	Describe the difference of accessToken obtained from normal APIs and anonymous APIs
		Section 6.3.1.2	Add new API for revoking CEK before expiry
		Section 6.3.3	In userType parameter, rename basic version of eID to "default" and full version of eID to "sign"
		Section 6.3.11, 6.4.7	Add new APIs for PDF digital signing
		Section 6.3.12, 6.3.13	Add new API for anonymous form filling
		Section 6.3.14, 6.3.15	Add new API for anonymous hash digital signing
		Section 6.3.16, 6.3.17	Add new API for anonymous PDF digital signing
1.2.0	21 February 2020	Whole document	Rebranding of eID to “iAM Smart”
		Section 6	Update definition of *source* parameter
		Section 6.3.5, 6.3.12, 6.3.14, 6.3.16	Minimise the number of required API parameters at Section 6.3.12, 6.3.14 and 6.3.16. Add new optional API parameters at Section 6.3.5
		Appendix B	Refine *source* parameter to support callback Online Service App via URL scheme or Universal Link / App Link

Ver. No.	Date	Section Affected	Summary of Changes
1.2.1	1 April 2020	Section 6.3.4 and 6.3.5	Elaborate the possible values of <context>
1.2.2	27 October 2020	Section 6.3.8 and 6.3.14	Add new optional API parameter to specify signature algorithm
		Appendix A	Add PhaseNo and FloorNum
1.2.3	22 October 2021	Section 1, 6.3.7, 6.3.12, 6.3.13 and 6.4.4	Add the description of the new billing address in the "e-ME" profile and the related fields addressDocInfo and addressDocFile
		Section 6.3.7 and 6.3.12	Update the description of formName, formNum and formDesc
		Appendix A	Add new billing address fields addressDocInfo and addressDocFile
1.2.4	11 Apr 2022	Section 5.1	Add the description on Universal Link (iOS) /App Link (Android) for Online Service App
1.2.5	15 July 2022	Section 5.3, 6.3.2, 6.3.5, 6.3.7, 6.3.12, 6.3.13 and 6.4.4	Update Form Filling scope, requests, response, and callback to V2. Online Service can request both eMEFields and profileFields with these V2 APIs
		Appendix D	Add the information about API Deprecation
1.2.6	14 September 2022	Section 4.1.6, 6.3.18	Add new API for verifying CCIC user
		Section 5.4	Update description of timeout value
		Section 6.1.2	New section for "Proper Handling of Encryption and Decryption"
		Section 6.3.7 and 6.4.4	Add optional request parameter callbackContentType (6.3.7) to support callback with the multipart/form-data type (6.4.4.2)
		Appendix A	Update billing address fields docFile and addressDocFile to support callback with the multipart/form-data type
2.0.0	05 June 2023	Whole document	Restructured the format and section of the API Specification Changed "e-Service" to Online Service

Ver. No.	Date	Section Affected	Summary of Changes
		Section 1, 1.1 and 1.2	Updated the description of Overview, Introduction of API Functions and "iAM Smart" Account Version and User Profiles
		Section 2.4.3	Updated the error code and description
		Section 2.5	Updated the Authorisation Scopes used in different scenarios
		Section 3.3.2.2 and 4	Added new API for Direct Login v2 and Profiles for Streamline Workflow
		Section 3.5, 4.6 and 5.6	Updated the details for the API deprecation
		Appendix A	Added description of "profileFields" and "eMEFields"
2.1.0	18 October 2023	Section 1.1, 2.5, 9 and 10	Added Bulk Digital Signing API description
		Section 4.6 and 5.6	Updated the API Deprecation shutdown date to 31 December 2025
		Whole document	The terms "government", "government and related organisations" and "GRO" can be used interchangeably in the whole document
2.2.0	31 January 2024	Section 2.4.3	Updated the error code and description
		Section 9.5.4	Added totalSignNum and failSignNum field for Callback API
		Section 9.5.5	Restructured the format of the signatures field in the API description.
		Section 10.5.7	Added totalSignNum and failSignNum field for Callback API.
		Section 10.5.8	Restructured the format of the signatures field in the API description.
		Section 3.4.10	add get last login status API

Ver. No.	Date	Section Affected	Summary of Changes
2.3.0	10 April 2024	Section 1, 1.1, 1.3.1, 3.3.4.1, 3.4.2, 4.1.1, 4.1.2, 5.5.3, 7.5.4, 9.5.2, 10.5.4	Updated description
		Section 3.3.4.1, 3.4.3 and 3.4.8	Added support of multiple android app for one Online Service for Authentication with package name verification
		Section 3.3.4.2, 3.4.5 and 3.4.9	Added Direct Login v2 (App) API description with new API for invoking Online Service App
		Section 4.2	Relocated Form Filling (v2) API
		Section 4.6.2	Updated the details for the API deprecation
		Section 4.6.2.3 and 4.6.3	Added support of multiple android app for one Online Service for Form Filling (v2) with package name verification
		Section 5.4.4, 5.5.4 and 5.5.8	Added support of multiple android app for one Online Service for Anonymous Form Filling (v2) with package name verification
		Section 6.4.3, 6.5.3, 6.5.7 and 6.5.8	Added support of multiple android app for one Online Service for Digital Signing with package name verification
		Section 7.4.4, 7.5.5 and 7.5.11	Added support of multiple android app for one Online Service for Anonymous Digital Signing with package name verification
		Section 8.4.3, 8.5.2 and 8.5.4	Added support of multiple android app for one Online Service for Re-authentication with package name verification
		Section 9.4.2. 9.5.2 and 9.5.8	Added support of multiple android app for one Online Service for Bulk Digital Signing with package name verification
		Section 10.4.4, 10.5.4 and 10.5.11	Added support of multiple android app for one Online Service for Anonymous Bulk Digital Signing with package name verification

Ver. No.	Date	Section Affected	Summary of Changes
		Appendix B	Segmented the relevant source parameter based on the use cases and scenarios
2.3.1	25 July 2024	Section 1, 3.5.1.3, 4.2.5.3.1, 4.2.3.5.2	Update OGCIO to DPO, and all URLs to Digital Policy Office theme page
		Section 4.1.5.1, 4.2.5.3.1, 4.2.5.3.2, 5.5.7, 5.6.2.2	Update the usage of "@ogcio.gov.hk" to "@digitalpolicy.gov.hk" in Examples
		Section 6.5.1, 6.5.2, 6.5.7, 6.5.8, 7.5.1, 7.5.2, 9.5.1, 9.5.8, 10.5.1	Update the usage of "Office of the Chief Information Officer" to "Digital Policy Office" in Example requests
		Section 5.5.8, 7.5.11, 10.5.11	Update example scheme for invoking the "iAM Smart" app
2.3.2	11 September 2024	Section 9.5.4, 10.5.7	Correct the typo of "sigMetadata" to "sigMetada" . Correct the typo of "hashCode" to "hash" for callback.
2.3.3	15 January 2025	Section 4.1.5, 4.2.5, 5.5.7	Update the response parameters and example response for the e-Me 2.0 and Profiles API regarding address data retrieved from CDEG.
2.4.0	29 April 2025	Section 2.4.3	Updated the error code and description
		Section 2.5	Added scope and updated description for Step-up Authentication
		Section 6.5.1, 6.5.2, 6.5.7, 6.5.8, 7.5.1, 7.5.2	Added support of Step-up Authentication on digital signing
		Section 11, 12	New section for Step-up Authentication
		Appendix B	Add supported in-app browser values

Ver. No.	Date	Section Affected	Summary of Changes
2.5.0	30 April 2025	Section 3.4.8, 4.2.5, 5.5.8, 6.5.7, 6.5.8, 7.5.11, 8.5.4, 9.5.8, 10.5.11, 11.5.4, 12.5.8	Updated the description of *clientRedirectURI* for appv2 APIs
		Section 4.2.5.1	Updated the description of *formName*, *formNum* and *formDesc*
		Section 13	New section for CDEG integration
2.5.1	31 October 2025	Section 2.4.3	Updated the return code of Signing Request and Step-up Authentication
2.5.2	05 February 2026	Section 2.4.1, 6.2, 7.2, 6.5, 7.5, 11.5, 12.5	Added rateLimitFactor in common API request header for Signing and Step-up Authentication
		Section 2.4.3, 13.4.2, 13.4.3, 13.5.2, 13.5.3, 13.5.4	New section for CDEG integration

Table of Contents

1. OVERVIEW

"iAM Smart" is a one-stop personalised digital services platform which enables users to log in and use online services by their personal mobile phone in a smart and convenient way. "iAM Smart" users can currently conduct authentication, digital signing and "e-ME" form-filling functions via "iAM Smart" mobile app or website services. They can also set up personalised notification services for receiving government service updates with "iAM Smart" mobile app. Users can register "iAM Smart" accounts using the "iAM Smart" mobile app, at the self-registration kiosks and at the registration counters. As of end April 2024, there were over 370 government and commercial online services adopted "iAM Smart" with over 2 million "iAM Smart" registered users.

To build a smart government, which in turn contributes to the development of Hong Kong into a smart city, the Chief Executive announced in the 2022 Policy Address the initiative to turn all government services online in two years by end-2024 and provide one-stop digital services by fully adopting "iAM Smart" within three years by 2025 so as to realise "single portal for online government services (一網通辦)".

To support this initiative of building a smart government, the Digital Policy Office (DPO), formerly known as the Office of the Government Information Officer (OGCIO), will upgrade "iAM Smart" to enable Government and Related Organisations ("GRO") to integrate their existing online services with it in a more convenient and simple manner, simplify the workflows and develop more services that can bring convenience to the public. It also allows citizens to enjoy the use of various online services more conveniently and swiftly, reducing their needs to visit the relevant offices to submit applications and deal with various businesses in person. Moreover, "iAM Smart" will be re-positioned not only for the sole purpose as a digital identity but also a portal for easy access to key GRO information and online services without the need of account registration.

1.1 Introduction of API Functions

"iAM Smart" functions are built in the form of RESTful Application Program Interface (API), which could be accessed by registered Online Services upon "iAM Smart" user authorisation. "iAM Smart" makes reference to OAuth 2.0 for authentication and authorisation amongst "iAM Smart" users, Online Services and the "iAM Smart" System. Online services adopting "iAM Smart" are required to provide RESTful callback APIs to receive API responses from the "iAM Smart" System. "iAM Smart" APIs support the following functions:

- Authentication

  Online services can make use of the Authentication API provide by "iAM Smart" to verify the identities of the users in a simple and secure way. The API can be used in various scenarios, such as user login, membership system, booking system, etc.

- Form Filling with Service Login

  Online services can make use of the Profiles API to retrieve data of "profileFields" and "eMEFields" in Streamline Workflow. The API can be used in various scenarios, such as account opening, account linkup, form filling, etc. GRO online services shall adopt Profiles API, while other online services shall adopt Form Filling(v2).

- Form Filling without Service Login (aka Anonymous Form Filling)

  Online services can make use of the Anonymous Form Filling API to request user's personal information without the need of prior authentication to Online Service with "iAM Smart".

- Digital Signing with Service Login

  Online Services can make use of Digital Signing API to enable "iAM Smart" user to complete online digital signing with legal backing after user has authenticated with Online Service using "iAM Smart". It can be used in many cases, such as digital signing an online application form and digital signing a contract and agreement.

- Digital Signing without Service Login (aka Anonymous Digital Signing)

  Online services can make use of the Anonymous Digital Signing API to enable "iAM Smart" user to complete online digital signing without the need of prior authentication with "iAM Smart".

- Re-authentication with Service Login

  Online services can make use of the Re-authentication API provided by “iAM Smart” to re-confirm the “iAM Smart” user’s identity before completing a critical transaction (e.g., confirm submission of tax return).

  After an “iAM Smart” user has authenticated Online Service with “iAM Smart”, Online Service can leverage “Request Re-authentication” API to request the same “iAM Smart” user to re-authenticate himself/herself to “iAM Smart” System via “iAM Smart” Mobile App.

- Bulk Digital Signing with Service Login

  Online Services can make use of Bulk Digital Signing API to enable “iAM Smart” user to complete online digital signing for multiple documents with legal backing after user has authenticated with Online Service using “iAM Smart”.

- Bulk Digital Signing without Service Login (aka Anonymous Bulk Digital Signing)

  Online services can make use of the Anonymous Bulk Digital Signing API to enable “iAM Smart” user to complete online digital signing for multiple documents without the need of prior authentication with “iAM Smart”.

- In-App Browser

  Online services can make use of the In-App Browser API to providing seamless experience to users while accessing their websites in “iAM Smart”.

- Step-up Authentication

  Online Services can utilize the existing Hong Kong Identity Card (HKIC) reading capability via Near Field Communication (NFC), combined with facial recognition provided by the "iAM Smart" OCR/FR system and the Immigration Department (ImmD) Checking system, to enable enhanced authentication for online services.

1.2 "iAM Smart" Account Version and User Profiles

"iAM Smart" Account Version

There are two versions of "iAM Smart", namely "iAM Smart" and "iAM Smart+". "iAM Smart" will provide general identity authentication and majority of the functions (such as "e-ME" form filling and personalised notifications), while "iAM Smart+" will provide the additional function of digital signing. Residents may download the mobile app and register for "iAM Smart" or "iAM Smart+" according to their needs.

Registration of "iAM Smart"

- Remote registration for "iAM Smart" requires applicant to use personal mobile phone for taking photos of the HKIC and selfies for identity verification purpose. The whole process can be completed online using mobile phone.

Registration of "iAM Smart+"

- Resident who has the new smart identity (ID) card can use his/her personal mobile phone with NFC (Near Field Communication) function for the remote registration of "iAM Smart+" via latest version of the "iAM Smart" mobile app.
- In-person registration for "iAM Smart+" can be made at self-registration kiosks in specified government premises and public locations by inserting HKIC for data retrieval and taking selfies to perform identity verification.
- In-person registration for "iAM Smart+" can also be made at registration service counters or "iAM Smart" mobile registration teams. The registration staff will check and verify the HKIC of the applicant. No photo-taking of the HKIC or selfie would be required during the registration process.

More information can be found in “iAM Smart” Thematic Website (https://www.iamsmart.gov.hk/reg.html).

“iAM Smart” User Profiles

A “iAM Smart” User Profiles contains both the account information and the personal data provided voluntarily in “e-ME” profile of corresponding “iAM Smart” account. Online Services can request “profileFields” and/or "eMEFields" of "iAM Smart" User Profiles via “iAM Smart” Profiles and Form filling API for the purpose of identity verification and form filling respectively.

- “profilefields”

  The “profileFields” consist of the account information including Hong Kong Identity Card (HKIC) number, English Name, Chinese Name (if available), date of birth and gender. “profileFields” are solely used for the purpose of identity verification, such as account opening, account matching, remote account opening, etc.

- "eMEFields"

  In addition to the account information, “iAM Smart” user can voluntarily input and update his/her personal information for the purpose of online form filling in “e-ME profile”. All these information for “iAM Smart” form filling are named as "eMEFields" which include HKIC number, English Name, marital status, phone number, email address, residential address and billing address[1], etc.

Details of the “iAM Smart” “profileFields” and “e-MEFields” can be found in Appendix A of the “iAM Smart API Specification”.

Both User Profiles of “iAM Smart” and “iAM Smart+” account version consist of HKIC number, English Name, date of birth and gender. In most cases, the field “Chinese Name” is also available in “iAM Smart” User Profiles, however, only the “iAM Smart+” version has verified the user’s “Chinese Name”[2]. Depending on the

[1] The billing address information is retrieved from one of the address data providers, including the electricity and gas companies and the Water Supplies Department. The billing address information consists of a PDF e-bill file and the related data, such as the provider name, owner name, service address, postal address, etc. This information could be used by Online Services as a kind of address proof, subject to the need for individual services.

[2] For the “iAM Smart+” account, the value of "chName" is verified, and the additional property "chNameVerified" with ImmD characters code point (with Private Use Areas (PUA), unable to render) will also be returned. Online Service shall use the “chName” for rendering purposes.

"iAM Smart" version, the account information would be verified with records stored at Immigration Department (ImmD) during registration. Regular checks for the "iAM Smart" user's account information with ImmD will also be conducted[3].

The account information of specific "iAM Smart" user from "profileFields" and "eMEFields" are the same (e.g., Online Service would get the same HKIC number of specific "iAM Smart" user, no matter the information is from "profileFields" or "eMEFields". However, both authentication and form filling statistic report in "iAM Smart" system would be updated if Online Service requests both "profileFields" and "eMEFields" in one API transaction. In addition, unlike "profileFields" for the purpose of identity verification, Online Service can allow "iAM Smart" user to modify online form data filled with "eMEFields".

[3] When changes are found during regular check, the "iAM Smart" user will be notified to re-register his/her "iAM Smart" account in order to update the profile information. The last modification date ("lastModifiedDate" provided in the POST response of API "Request accessToken & Tokenised ID") of his/her "iAM Smart" Profile will also be updated to the current date upon completion of re-registration in this situation.

1.3 Online Service Onboarding

1.3.1 Onboarding Journey

Testing Environment

1. Online Service shall prepare the following information before submitting the application form.
   - The public key certificate that is issued by Hong Kong Recognised Certification Authorities (Encipherment Cert);
   - Internet-facing server endpoint for "iAM Smart" callback with HTTPS enabled (depends on API used);
   - Prepare Tester's Apple ID and Google Play Account for Testing App;
   - Setup Universal Link (iOS Online Service App); and
   - Provide Android package name, activity class name (Android Online Service App).

2. Upon application approval, the Support Team will create the Client ID and issue an email for the Online Service administrator in the self-service portal. The support team will provide a testing account for Testing App.

3. Online Service administrator, creator and approver shall complete the following tasks in the self-service portal:
   - Setup Creator and Approver accounts;
   - Setup KEK certificate (Encipherment Cert) from HKRCA[4];
   - Check the Client's Secret;
   - Setup callback/ redirectURI/ Direct Login v2 whitelist;
   - Setup Android Package Name and SHA-256 APP fingerprint for Android Online Service App; and
   - Setup iOS Universal Link for iOS Online Service App

Production Environment

1. Online Service shall prepare recordings for UI and Workflow Verification Process and submit them to the support team.
2. Apply KEK certificate (Encipherment Cert) from HKRCA
3. Upon UI and Workflow Verification approval, Online ervice shall submit production application forms and related documents to the support team.
4. Upon application approval, Online Service shall

[4] Recognized Certification Authorities in Hong Kong
https://www.digitalpolicy.gov.hk/en/our_work/digital_infrastructure/legal_framework/regulation/eto/ordinance/ca_in_hk/

- Setup Creator and Approver accounts;
- Setup callback/ redirectURI/ Direct Login v2 whitelist;
- Setup Android Package Name and SHA-256 APP fingerprint for Andoird Online Service App; and
- Setup iOS Universal Link for iOS Online Service App

1.3.2 Self-Service Portal

Online Service has to set up a KEK certificate (Encipherment Cert) and callback whitelist in the portal. Please refer to the Self-Service Portal User Guide for details.

1.3.3 Environments

1.3.3.1 Testing Environment

Domain Name	apigw-isit.staging-eid.gov.hk
URL Scheme of "iAM Smart" mobile app (iOS and Android)	hk.gov.iamsmart.testapp://
"iAM Smart" App Installation	By invitation with AppStore and PlayStore with the tester list.
Self-Service Portal URL	https://portal-isit.staging-eid.gov.hk/ESP/index.html
KEK Certificate (Encipherment Cert)	Trial Cert from Hong Kong RCA is allowed
HTTPS Callback Endpoint	Https with production cert

1.3.3.2 Production Environment

Domain Name	apigw.iamsmart.gov.hk
URL Scheme of "iAM Smart" mobile app (iOS and Android)	hk.gov.iamsmart://
"iAM Smart" App Installation	Officially download from App Store, Google Play, AppGallery or apk file from "iAM Smart" thematic page
Self-Service Portal URL	https://portal.iamsmart.gov.hk/ESP/index.html
KEK Certificate (Encipherment Cert)	Production Cert from Hong Kong RCA
HTTPS Callback Endpoint	Https with production cert

2. SECURITY

2.1 HTTPS

Online Service endpoint shall support TLS 1.2 or compatible for using “iAM Smart” APIs.

2.2 Callback Verification From “iAM Smart” System

“iAM Smart” System supports Online Service to verify on the transport layer that the callback request it receives actually comes from “iAM Smart” System. If online service chooses to verify, it can request the public key certificate of “iAM Smart” System during the onboarding process and configures the certificate in the corresponding location of its HTTPS server to implement the client authentication function. Specific configurations vary depending on the HTTPS server or the reverse proxy used, and the corresponding administration manual needs to be consulted. For example, if using Nginx, online service should set ssl_client_certificate and ssl_verify_client in its configuration file to authenticate client requests.

2.3 API Data Encryption and Decryption

HTTPS will be used to secure communication channels between Online Services and the "iAM Smart" System. However, there is still a chance that the data transferred by HTTPS could be eavesdropped or manipulated by malicious service providers in some special situations. To better protect the data in transit, an additional layer of data encryption will be applied to all APIs POST request (except the one that online service request for getting symmetric encryption key). Callbacks from "iAM Smart" System will also be encrypted using the method described herein.

The overall idea of API data encryption is:

1. Online service requests for data encryption key from "iAM Smart" System;
2. "iAM Smart" System generates a symmetric data encryption key;
3. "iAM Smart" System stores the generated key and returns to the online service who initiated the request in secure manner;
4. For subsequent interactions between online service and "iAM Smart" System, business data will be encrypted by the generated key;
5. Both "iAM Smart" System and online service will use the same key to decrypt the API data.
6. If the generated key is expired, online service have to request a new key and repeat the above process.

2.3.1 Encryption and Decryption Workflow

The following diagram describes detailed workflow on how online service can request data encryption key, perform encryption and decryption with the key, and request for another new encryption key if the existing key is expired:

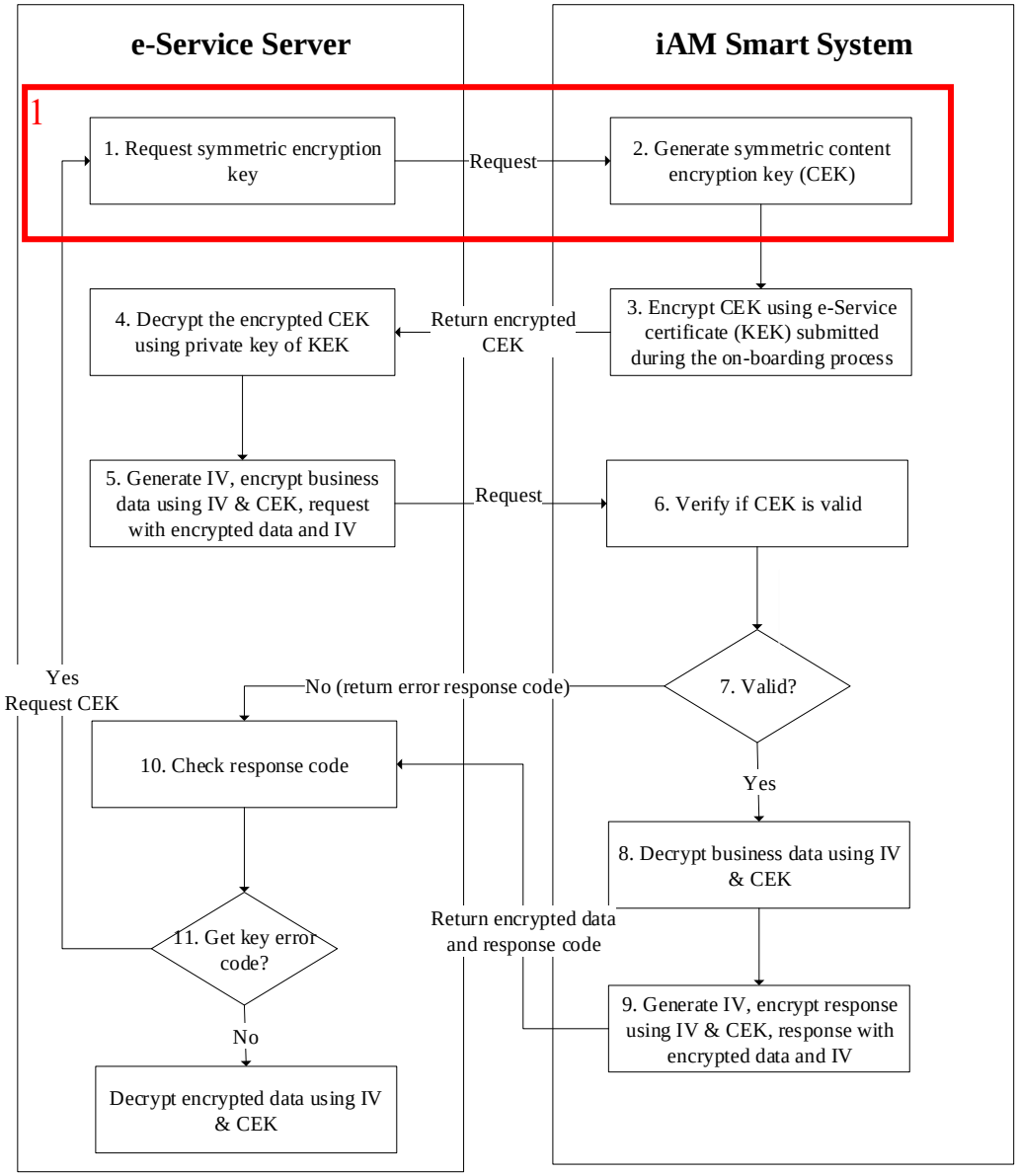

Figure-1 Application of Content Encryption Key and Procedures of Encryption and Decryption (Perform by Online Service)

The APIs for online service to request Content Encryption Key are marked in red.

No.	API Name	API Reference (Section)
1	Request Symmetric Content Encryption Key	2.3.6.1

High Level Workflow Description of encrypting API request data using Content Encryption Key:

- During the onboarding process, online service have to upload its certificate which was issued by Recognised Certification Authorities. The certificate contains a public key (denoted as KEK, the Key Encryption Key) that will be used in the following steps.
- "iAM Smart" System provides API to online service for retrieving a symmetric key (denoted as CEK, the Content Encryption Key). Such key will be used to encrypt/decrypt subsequent API data. AES256 symmetric encryption algorithm will be adopted for encrypting the data. CEK will be expired after specified period (refer to parameter at 2.3.6.1) of generation.
- When "iAM Smart" System received request from online service for retrieving CEK, "iAM Smart" System will employ the RSA algorithm and use Online Service's KEK to encrypt the CEK. Then "iAM Smart" System will send the encrypted CEK to Online Service. Online service can use the private key of the KEK to decrypt and obtain the symmetric key CEK.
- Before online service calling APIs of "iAM Smart" System, online service should check whether the symmetric data encryption key (CEK) exists and is valid. If online service has no such CEK or the CEK is expired, then online service should request a new CEK from "iAM Smart" System. The CEK will be used to encrypt subsequent API data until the key was expired.
- Upon receiving the request from Online Service, "iAM Smart" System will check whether Online Service's data encryption key CEK exists and is still valid. If CEK is not exist or expired, error message will be returned to Online Service. If CEK is still valid, the key will be used to decrypt the request data. Same CEK will also be used to encrypt respective response or callback data for returning to Online Service.

High Level Workflow Description of Callback Encryption and Decryption:

- Before "iAM Smart" System initiates a callback, it will check if the data encryption key (CEK) of the corresponding online service is still valid. If the CEK is expired or does not exist, "iAM Smart" System will regenerate a new data encryption key for that Online Service.
- "iAM Smart" System uses the CEK to encrypt the callback body. Then such CEK will be encrypted by KEK of the Online Service. The encrypted

callback data and the encrypted CEK will be returned to online service through the callback.

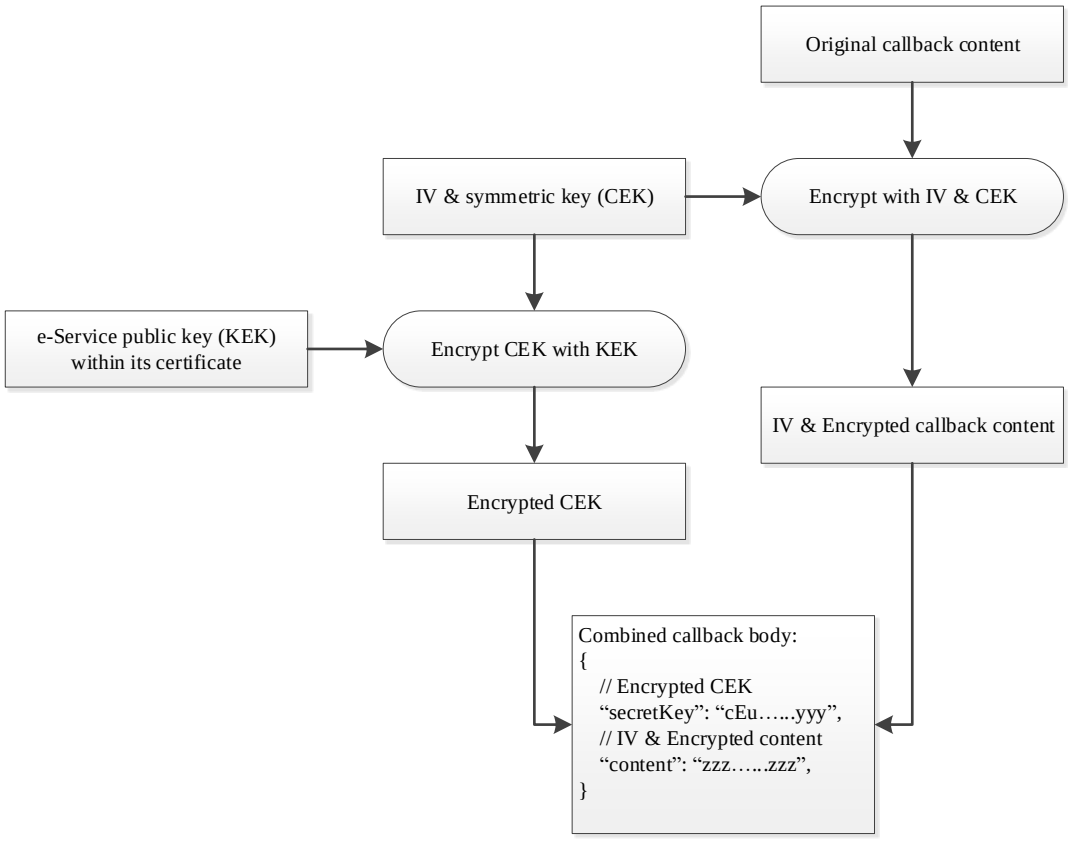

Figure-2 Callback Encryption Process (Perform by "iAM Smart" System)

- After online service receiving the callback, online service should use the private key of the KEK to decrypt the encrypted data encryption key (CEK). Then the decrypted data encryption key (CEK) can be used to decrypt the callback data.

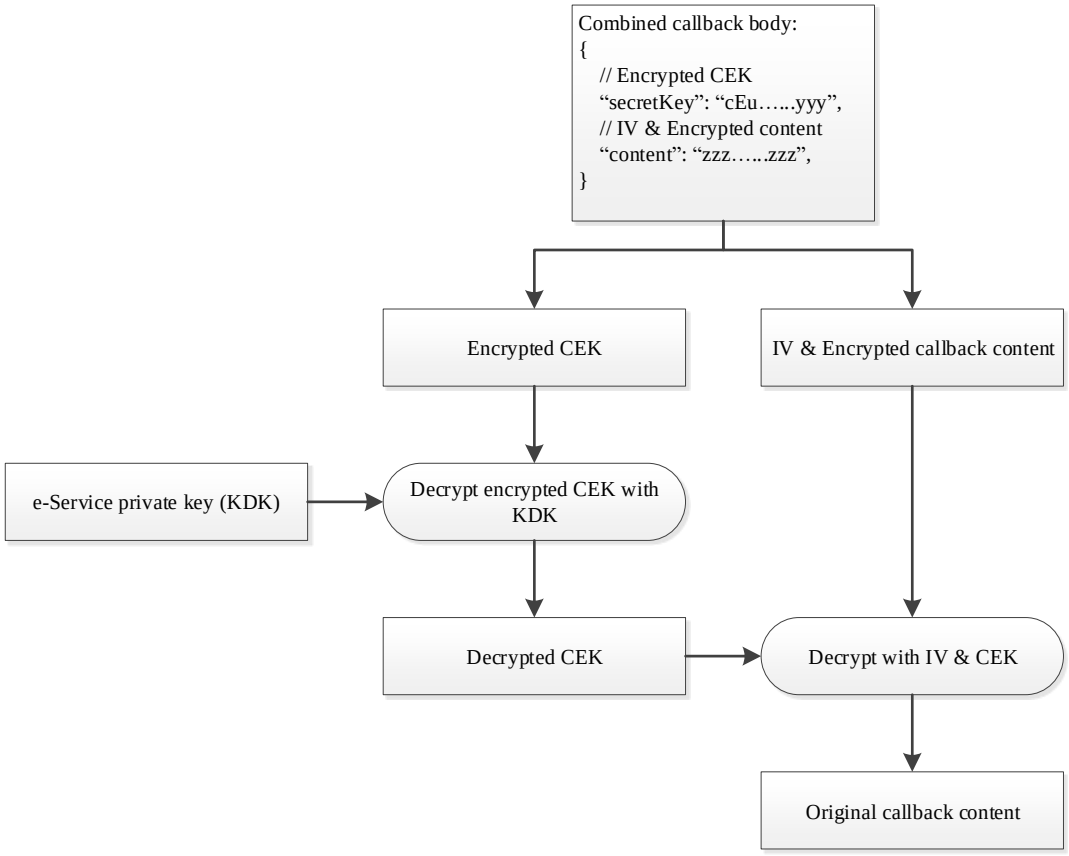

Figure-3 Callback Decryption Process (Perform by Online Service)

2.3.2 Proper Handling of Encryption and Decryption

Online service shall handle the following scenarios properly in their system design:

- Due to unexpected reasons, online service may receive Encryption/Decryption Error Code. Online service shall perform retry of the "Request Symmetric Content Encryption Key" API specified in section 2.3.6.1 of this document to obtain the latest CEK and try to decrypt content with such CEK when it is received.
- Due to operational needs, the "expiresIn" response parameter of the "Request Symmetric Content Encryption Key" API specified in section 2.3.6.1 of this document may dynamically change, online service shall

calculate the expiry time of the CEK with “issueAt” and “expiresIn” response parameters.

- Under certain circumstances, “iAM Smart” API service may be unavailable for a certain period of time. The online service shall prepare the fallback procedures (e.g. switch to paper mode, prevent users from accessing “iAM Smart” functions, etc.).
- When there are too frequent API requests from an Online Service, the online service may receive HTTPS status Code 429, “Too Many Requests”. Online service shall store the CEK to avoid unnecessary “Request Symmetric Content Encryption Key” API request (see section 2.3.6.1 of this document).
- Even a new KEK is configured for effective in a specified period, the existing CEK will still be returned through the “Request Symmetric Content Encryption Key” API specified in section 2.3.6.1 of this document and encrypted by the old KEK until the expiry of the existing CEK. To use the new KEK, online service shall request a new CEK through the “Revoke Symmetric Content Encryption Key” and “Request Symmetric Content Encryption Key” APIs specified in sections 2.3.6.2 and 2.3.6.1 respectively of this document or to request a new CEK after itsits expiry through the same APIs.

2.3.3 Encryption and Decryption Scope

API Type	Parameter Type	Parameter Type/Field	Will Encrypt?	Encryption Rules
Online service request and response	request parameters	request body	YES	1. Convert the request body into json string 2. Use "AES/GCM/NoPadding" to encrypt the byte array of the json string 3. Base64 encode the encrypted result and set the encoded string to the content field of the request body
		request header	NO	N/A
		url parameters	NO	N/A
	response parameters	txID	NO	N/A
		code	NO	N/A
		message	NO	N/A

		content	YES	1. Convert the content field of response body into json string 2. Use "AES/GCM/NoPadding" to encrypt the byte array of the json string 3. Base64 encode the encrypted result and set the encoded string to the content field of the response body
"iAM Smart" callback	request parameters	request body	YES	1. Convert the content field of request body into json string 2. Using "AES/GCM/NoPadding" to encrypt the byte array of the json string 3. Base64 encode the encrypted result and set the encoded string to the content field of the request body 4. Encrypt the symmetric data encryption key (CEK) with the public key certificate (KEK). Base64 encode the encryption result and set the encoded string to the key field of the request body
		request header	NO	N/A
		url parameters	NO	N/A
	response parameters	N/A	N/A	N/A

2.3.4 Request and Callback Examples

- online service calls "iAM Smart" API

Request Parameter

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/auth/getToken
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
```

```
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
// Clear text before encryption
// {
//   "code": "xxxa42e76bf4cb0846a68e6d83d6096",
//   "grantType": "authorization_code"
// }
// iv + ciphertext
{
  "content":
"AAAADMBVmPKZZmm6/DBASQjG/0Nsl1oBVKk8OcaQLsjiM13XD/c1sZAkQfQQLiTFCxutL9TDC
yVxh1pUNQk//9VtPKXqmPbB1S/SgeNHYXhgbnQWvNW6Sx4pUcI440JbmUuU5JFWEnGViBjKmRG
Z5QJRYXG89BI0bcjmrpMOnim0TKQ="
}
```

Example Success Response

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  // Clear text before encryption
  // "content": {
  //   "accessToken": "0ad186353c424c64897fcc00445c9ba1",
  //   "tokenType": "Bearer",
  //   "issueAt": 1557053922938,
  //   "expiresIn": 14400000,
  //   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
  //   "lastModifiedDate": 1560849218006,
  //   "userType": "sign",
  //   "scope": "eidapi_auth eidapi_formFilling"
  // }
  "content":
"AAAADIF/FmVrY9dfl28k2fWjAFrRr29kTBVZfQTvJX+eN454TGl8X9SQ1DkY+W9ntIwmwXRz8
fajpk4XxYoM9702pFjK4JBzu6+0dCHo0TlNruMRD10nMstMthpT7O0hkouEWYmBnS8YJVpQz2e
```

```
kaxTq0jWc0X0rj1B16Ir+mbs9h3qDEonbHSceDthfdKZsqLd67wfc9KwXVgVCMdr0vo3H4tdjY
YfV1utTLNdZbhO9Z0C0Tnr64/7xZq5lZMnQ+ttEhksBeD9/zmPGVs77CzoWJihYW7S7dfZHAcZ
foSIeS2IbBeXnVb0Yg/wkN1FFEPcOSf6DkO4Yck/0iuN/iuXchGbrf5hz/mkhKdi2goxNt/vvU
+N8Lk6U8LdwZNmaAZLYXd3jL9ZHnUBkuygcHN8OyaxEkLlJOJ4FFPZpCf2O19aPyCxYJDK1gzp
9+k+eD8wn3XH40ezYYUJrIqkW9+sS6KX3"
}
```

Example error response

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D30002",
  "message": "content encryption key not exist or expired",
}
```

- "iAM Smart" calls back Online Service

Request Parameter

```
// Line breaks are for legibility only.
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>
// Callback Body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  // Clear text before encryption
  // "content": {
  //   "businessID": "2YotnFZFEjr1zCsicMWpAA",
  //   "state": "unesidkd",
  //   "hashCode": "tGzv3JOkF0XG5Qx2TlKWIA",
  //   "timestamp": 1556450176000,
  //   "signature": "nnoadisauflanefhykdjf",
  //   "cert": "sdfGSDGsdfaGDEHfjslgGQG……GSGjljlkjwmh",
  // }
  // Base64 encoded encrypted CEK
  "secretKey":
"XlUu98zV6/W4LPDfZZxzOVY62e4PHZ1e3BzENgA/ysO+rxmnBlqmYRUOqYKS8Fj/6EpuCA1js
nNY/NJWh8KBz/Kxfe03I1SuD/lHemzMahXLc9x58ecGU0q59VRNO+powUdPTyu0EPH8izh8GQ+
Ht33gicl4GMmXuwjmqoo4TZl6h4hPGf6al6piQ9gucsjgFpIoP9genqXqcDCmNkhLoLUMrk5t/
ZsA97AP7DHAGh5RcUT3Qktwz+D1I7Nd4oWUS6yo8ci39epiYSrG9B06OZ/kVo7k6to1Q7KVoCK
j3PsMJgCUVfOQ0HP1YnH599Cu4eyULaa3rIpafpycPvqh9g==",
```

```
  // iv + ciphertext
  "content":
"AAAADFaP8/FuTcoymbq2hswpgHER9BUM5itaTKWo2/UwtG4wWRsMgri9bSMZ6uLxjKgmGxKTi
R/5N8JuqgX1WJjoMJHp8wnLgSU4FFuGr5yNuGiLpW62fmf95GdF/ikInE5zkzT3hJRzTXbdTAL
eXWWPu/88WyZn1REGqSs6m2AdMY5pIhgia8bCHBxSIPBGuBtAujMII7Nlb6ocFwUL0QuaZqPF/
hNJWt/EI46YGHegKiRB6KP6hxg6K5I9Sz77uQfO60PV5833fnvU"
}
```

2.3.5 Encryption and Decryption Reference Implementation

- Data Encryption

```
private static final String AES_GCM_NOPADDING = "AES/GCM/NoPadding";
/**
 * AES GCM encryption
 * @param contentByte - Bytes of content to be encrypted
 * @param key - CEK (AES 256)
 * @return
 */
public static String encrypt(byte[] contentByte, byte[] key) {
     SecretKeySpec skeySpec = new SecretKeySpec(key, AES);
     byte[] encrypted = null;
     try {
          SecureRandom secureRandom = new SecureRandom();
          byte[] iv = new byte[12];
          // do not reuse iv with the same key
          secureRandom.nextBytes(iv);

          Cipher cipher = Cipher.getInstance(AES_GCM_NOPADDING);
          GCMParameterSpec parameterSpec = new GCMParameterSpec(128, iv);
          cipher.init(Cipher.ENCRYPT_MODE, skeySpec, parameterSpec);
          encrypted = cipher.doFinal(contentByte);

          ByteBuffer byteBuffer = ByteBuffer.allocate(
               4 + iv.length + encrypted.length);
          byteBuffer.putInt(iv.length);
          byteBuffer.put(iv);
          byteBuffer.put(encrypted);
          byte[] cipherMessage = byteBuffer.array();

          return Base64Util.base64Encode(cipherMessage);
     } catch (NoSuchAlgorithmException | NoSuchPaddingException
          | InvalidKeyException | InvalidAlgorithmParameterException
```

```
        | IllegalBlockSizeException | BadPaddingException e) {
        log.error("Encryption error: {}", e);
        throw new IAMSmartException(e);
    }
}
```

- Data Decryption

```
private static final String AES_GCM_NOPADDING = "AES/GCM/NoPadding";
/**
 * AES GCM decryption
 * @param content - Bytes of content to be decrypted
 * @param key - CEK (AES 256)
 * @return
 */
public static String decrypt(byte[] content, byte[] key) {
    SecretKeySpec skeySpec = new SecretKeySpec(key, AES);

    ByteBuffer byteBuffer = ByteBuffer.wrap(content);
    int ivLength = byteBuffer.getInt();
    if(ivLength != 12) {
        throw new IllegalArgumentException("invalid iv length");
    }
    try {
        byte[] iv = new byte[ivLength];
        byteBuffer.get(iv);
        byte[] cipherText = new byte[byteBuffer.remaining()];
        byteBuffer.get(cipherText);

        Cipher cipher = Cipher.getInstance(AES_GCM_NOPADDING);
        GCMParameterSpec parameterSpec = new GCMParameterSpec(128, iv);
        cipher.init(Cipher.DECRYPT_MODE, skeySpec, parameterSpec);
        byte[] original = cipher.doFinal(cipherText);
        return new String(original);
    } catch (NoSuchAlgorithmException | NoSuchPaddingException
        | InvalidKeyException | InvalidAlgorithmParameterException
        | IllegalBlockSizeException | BadPaddingException e) {
        throw new IAMSmartException(e);
    }
}
```

2.3.6 API Encryption Details

2.3.6.1 Request Symmetric Content Encryption Key

- **API Description**

Name	Description
Service Full Name	Request Symmetric Content Encryption Key
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/security/getKey`
Request Type	POST
Service Version	1.0.0
Description of Service	Each online service has to request a data encryption key to encrypt business data for **every POST** request when calling the "iAM Smart" API. This API allows online service to request symmetric Content Encryption Key for encrypting business data using AES-256 algorithm. Key expiration time will also be specified in the API response.
Parameters Encrypted or no	No

- **Request Parameters**

Common API Parameters described 2.4.1

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/security/getKey
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1562689594000
nonce: "e893647dc4204eb9b7b8eddd527b687c"
```

- **Response Parameters**

Parameter	Type	Presence	Description

secretKey	String	Required	Base64-encoded symmetric data encryption key encrypted by the certificate that online service submitted to “iAM Smart” System during the onboarding process (self service portal).
pubKey	String	Required	Base64-encoded public key of the certificate that is used to encrypt the symmetric data encryption key.
issueAt	Long	Required	secretKey issue time expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
expiresIn	Long	Required	Key expiration time, expressed in milliseconds.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "secretKey":
"X2WnRD+LhN5GZp5n1CKLqKr2pmoMtZHgpHs7umznCVQL5co1wPmR3nkqsDzp9JZ9HjQYxiHfW
z0MIlb7iShlWZZO/bfYkUXeazKWfXc/0VPQw1c0VzFJUIem0+ygtjxg6rL7zlQijrLNeWgrWRS
8RVtlQc8GpIZONAnHONzNvE/uJmmqAMElKpy88rHI2etzhXZIA05RvJb4eQigH+hgJxBjvOp5e
4JdBXZxde6dF56/JCwppeb/MhmFcsJQLKN+mG9JGK5/2D5NKQdzqrrRUD9LH8LmccZmgREdEUr
fegfbJlGGejDRmyiuwefvwtNwjeXUrfQ3MqzR/VrOxlFzmw==",
    "pubKey":
"MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAjAR2BAXJxP01Sla9AEmJiC+aVhTwu
sk3gzYUPKkE5y02YH7cp7o3C8wQYwA//f+tUGNXixyIdjfS1FZ3VLPXLXgZbDS/x2b+fPqCpQf
fZD+hmFaYfJHb1hrRSLxXzBUKgWAdkUjSTqUt/wuzUkIcntMNiP9S9/Tf0kIs97h+3u0EO2v/f
hP+kEbQdT4NTZGKQitxFlGdOEi9GumeG1gRcHRLRurs9LFifoYnNMpG4FxeoovCvDusprmBnDa
jGr9IavA0GczZ3asvsduwaKQPAmFp07/jGs1dYO2X4K5aZ7FLxFBbNco5GI7nFN22sHm/A5bju
rfr/AN3a9RmlkIy9QIDAQAB",
    " issueAt": 1557053922938,
```

```
    " expiresIn": 86400000
  }
}
```

- **Example Error Response**

```
//The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20006",
  "message": "signature verification failed",
}
```

2.3.6.2 Revoke Symmetric Content Encryption Key

- **API Description**

Name	Description
Service Full Name	Revoke Symmetric Content Encryption Key
URI (as in RESTFUL API)	https://<iAM_Smart_domain>/api/v1/security/revokeKey
Request Type	POST
Service Version	1.0.0
Description of Service	This API allows online service to revoke symmetric Content Encryption Key.
Parameters Encrypted or Not	No

- **Request Parameters**

Common API Parameters described 2.4.1

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/security/revokeKey
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
```

```
timestamp: 1562689594000
nonce: "e893647dc4204eb9b7b8eddd527b687c"
```

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS"
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20006",
  "message": "signature verification failed",
}
```

2.4 Common Parameters

2.4.1 Request Parameters

There are two kinds of parameters in each API request, the common parameters and the specific business parameters.

(1) Common Parameters

All of the required common parameters should be included in **every POST** request that online service sending to the "iAM Smart" System. The parameters should put in the POST request headers (see examples in the following sections for details). For GET requests or callbacks, most of these common parameters are also required. Which of them should be used and where to put them will be described in details in every GET requests or callback.

Parameter	Type	Presence	Description
`clientID`	String	Required	Online service client identifier. The clientID will be assigned to Online Service at the initial registration.
`signatureMethod`	String	Required	signature algorithm: HmacSHA256
`timestamp`	Long	Required	The timestamp is the request submit time expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT. It is used to prevent replay attack. The value MUST be a positive integer and equal or greater than the timestamp used in previous requests.
`nonce`	String	Required	A nonce is a random string, uniquely generated for each request by Online Service. It is used to prevent replay attack. A nonce can be an ASCII string of any length less than or equal to 36 (UUID string length) as long as the uniqueness requirement is met.
`rateLimitFactor`	String	Required (Conditional)	Base64-encoded rate limit factor required for specific functions. `{"key":"suaMethod","value":["FR","NFC"]}`.

			Online Service must submit this parameter while requesting the Step-up Authentication function.
signature	String	Required	It is a signature of the submitted data. Online service uses clientSecret to sign the concatenated string of clientID, signatureMethod, timestamp, nonce, and encrypted_request body to get the signature. The pseudo code to generate the signature is shown at below.

```
Pseudo code for signature generation
message = clientID + signatureMethod + timestamp + nonce +
encrypted_request_body
sha256_HMAC = Mac.getInstance("HmacSHA256");
secret_key = new SecretKeySpec(secret.getBytes(), "HmacSHA256");
sha256_HMAC.init(secret_key);
hash = Base64.encodeBase64String(sha256_HMAC.doFinal(message.getBytes()));
// If the hash needs to be URL encoded
signature = URLEncoder.encode(hash, "UTF-8")
```

(2) Specific Business Parameters

"iAM Smart" System returns results based on the business request parameters. Those parameters are normally put in the request body as json data and submitted as POST request.

2.4.2 Response and Callback Format

"iAM Smart" System returns data in json format. The format is:

```
{
  // The txID is a unique identifier of response message.
  // The identifier will be used during troubleshooting
  "txID": "xxxxxx",
  "code": "xxxxxx",      // return code
  "message": "xxxxxx",   //status message
  "content": {
```

```
      // the actual business data
    }
  }
```

When the code is D00000, the service execution is successful, and the `content` field returned by the HTTPS request contains the actual business data.

Other than D00000, it indicates that the service execution is abnormal, and the `content` field will be empty or only contains required parameters, e.g. `businessID`.

For callback, if the data passed are in the URL and error occurred before sending the callback, the URL will contain the error_code field to tell online service the error code. If the data is in the callback body, the above json format will be used.

2.4.3 Return Code

Code	Message/ Description
D00000	SUCCESS
Common Error Code	
D20000	unknown exception
D20001	parameter {%s} is missing
D20002	empty parameter {%s}
D20003	invalid parameter {%s}
D20004	duplicated request
D20005	unsupported signature method
D20006	signature verification failed
D20007	unsupported source
D20008	invalid online service URL The callback URL provided by online service is not registered in the ESP portal
D20009	accessToken not exist or expired
D20010	openID not exist
D20011	duplicated businessID
D20012	insufficient permission for online service / Forbidden Online service shall check the scope or API access control in the the ESP portal
D20015	scope mismatch

	Online service shall verify the scopes approved for the client id in ESP and the scopes configured in online service catalogue.
Encryption/Decryption Error Code	
D30001	key encryption key not exist or expired
D30002	content encryption key not exist or expired
D30003	encryption exception
D30004	decryption exception
Authentication Error Code	
D40000	user cancelled authentication request
D40001	user rejected authentication request
D40002	failed to authenticate
D40003	authentication request timeout
D40004	authCode not exist or expired
Profile Request Error Code	
D50001	user rejected profile request
D50002	failed to request profile
D50003	profile request timeout
Form Filling Request Error Code	
D60000	user cancelled Form Filling request
D60001	user rejected Form Filling request
D60002	failed to request Form Filling
D60003	Form Filling request timeout
Signing Request Error Code	
D70000	user cancelled signing request
D70001	user rejected signing request
D70002	failed to request signing
D70003	signing request timeout
D70004	user not allowed to sign
D70005	inconsistent HKIC number
D70006	failed to process signing acknowledgement
D77003	device not supported for NFC
D77004	user is still in waiting period
Bulk Digital Signing Request Error Code	
D71000	user cancelled bulk signing request
D71001	user rejected bulk signing request
D71002	failed to request signing
D71003	signing request timeout

D71004	user is not allowed to sign
D71005	inconsistent HKIC number
D71006	failed to process signing acknowledgement
D71007	the request to be cancelled is not Pending for Signing
D71008	BSQC Token not found
D71009	inconsistent department or service name
Re-authentication Error Code	
D80001	user rejected re-authentication request
D80002	failed to request re-authentication
D80003	re-authentication request timeout
Step-up Authentication Error Code	
D41001	user rejected step-up authentication request
D41002	step-up authentication request timeout
D41003	inconsistent HKIC number
D41500	failed to step-up authenticate
D41503	device not supported for NFC
D41006	user is still in waiting period
D41007	user is not allowed to step-up authentication
CDEG Consent Request Error Code	
D42001	user rejected CDEG consent request
D42002	CDEG consent request timeout
D42003	failed to CDEG consent request

2.4.4 HTTP Status Code

HTTP Status Code	Description
200 OK	Request successful
302 Found	The requested resource resides temporarily under a different URI.
400 Bad Request	The request could not be understood by the server due to malformed syntax.
401 Unauthorised	The request has not been applied because it lacks valid authentication credentials for the target resource.
403 Forbidden	The server understood the request but is refusing to fulfil it.
404 Not Found	The server has not found anything matching the Request-URI.

413 Payload Too Large	The uploaded files are larger than 5MB or the request entity is larger than 10MB.
429 Too Many Requests	The user has sent too many requests in a given amount of time.
500 Internal Server Error	The server encountered an unexpected condition which prevented it from fulfilling the request.
503 Service Unavailable	The server is currently unable to handle the request due to a temporary overloading or maintenance of the server.

2.5 Authorisation Scopes

“iAM Smart” System makes reference to the OAuth 2.0 authentication framework to enable Online Service to gain access to “iAM Smart” APIs, with “iAM Smart” user's authorisation using “iAM Smart” Mobile App.

Determine Authorisation Scope
To access “iAM Smart” APIs, Online Service should first determine the authorisation scope(s) required for the “iAM Smart” API(s) invoked for its business needs. Online service may refer the table below for the authorisation mapping:

API	Scope Value
Authentication	eidapi_auth
Form Filling with Service Login (aka Profiles)	eidapi_profiles
Form Filling without Service Login (Anonymous Form Filling)	eidapi_formFilling
Digital Signing with Service Login	eidapi_sign
Digital Signing without Service Login (Anonymous Digital Signing)	eidapi_sign
Re-authentication with Service Login	eidapi_fr
Bulk Digital Signing with Service Login	eidapi_bulksign
Bulk Digital Signing without Service Login (Anonymous Bulk Digital Signing)	eidapi_bulksign
Step-up Authentication with Service Login	eidapi_sua
Step-up Authentication without Service Login (Anonymous Step-up Authentication)	eidapi_sua

Online Service should combine all authorisation scope(s) required into a single request to get authorisation from “iAM Smart” user through “iAM Smart” System when “iAM Smart” user requests login to Online Service using “iAM Smart” Mobile App. For example, authorisation scope “eidapi_auth”, “eidapi_formFilling” and “eidapi_sign” are for “iAM Smart” authentication form filling and digital signing respectively.

3. AUTHENTICATION

3.1 Overview

Online service can make use of the Authentication API provide by “iAM Smart” to verify the identities of the users in a simple and secure way.

3.2 Scopes

Online service shall apply the following scope values to “iAM Smart” Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scope values in the Self-Service Portal (ESP).

Scope	Description
`eidapi_auth`	Scope for Authentication

3.3 Use Cases and Scenarios

3.3.1 Authentication (Online Service Website in Different Device)

The sequence diagram below shows how an online service website leverages the "iAM Smart" System to perform user authentication when the "iAM Smart" Mobile App is installed in a separated mobile device.

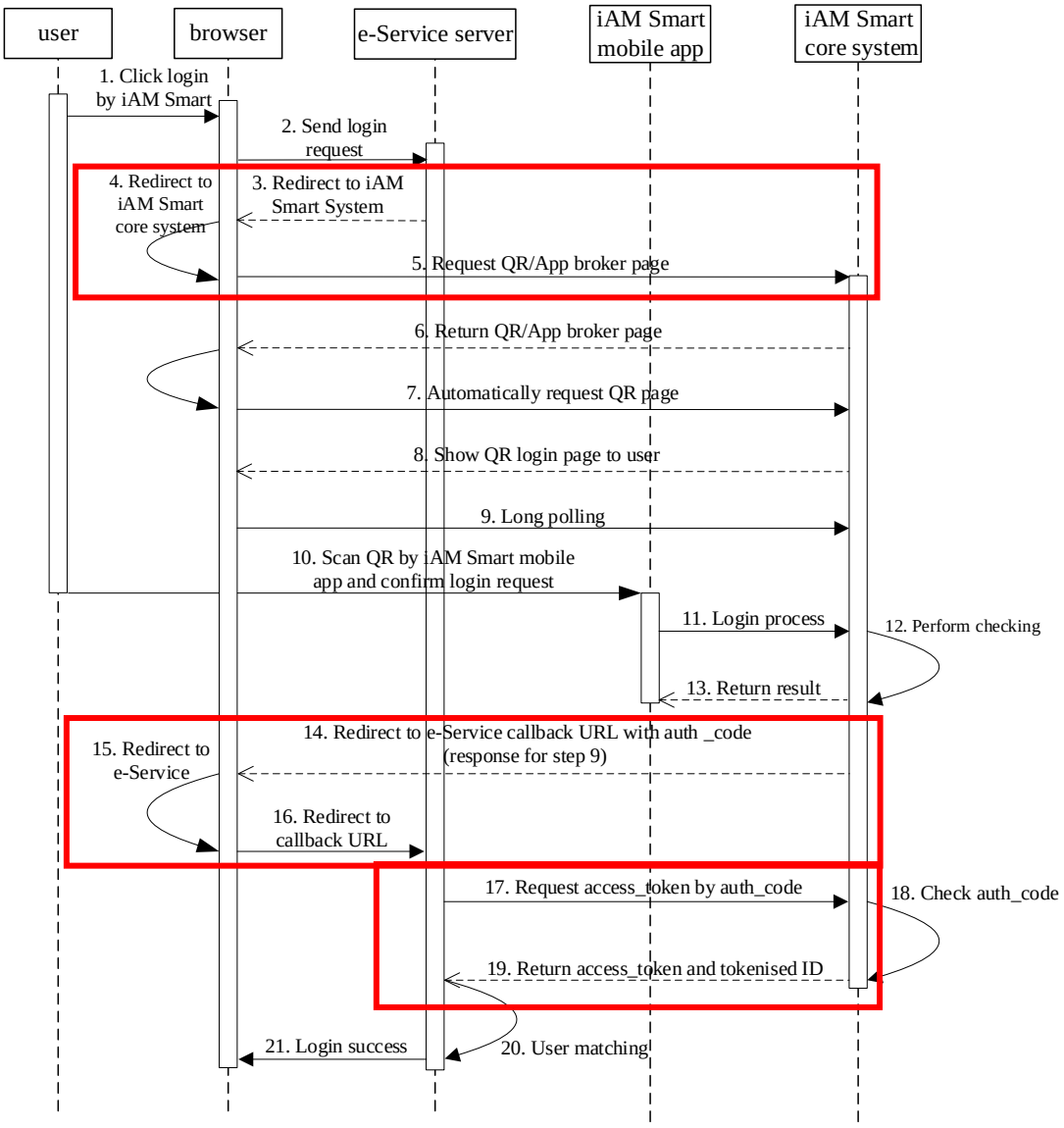

Figure-4 Authentication (Online Service Website in Different Device)

APIs interactions between online service servers and the “iAM Smart” System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Request QR Page	3.4.1
2	Callback with authCode to Online Service Server	3.4.4
3	Request accessToken & Tokenised ID	3.4.5

3.3.2 Authentication (Online Service Website in Same Device)

3.3.2.1 Initiated from Online Service Website

The sequence diagrams below shows how an online service website leverages the "iAM Smart" System to perform user authentication when the "iAM Smart" Mobile App is installed on the same mobile device.

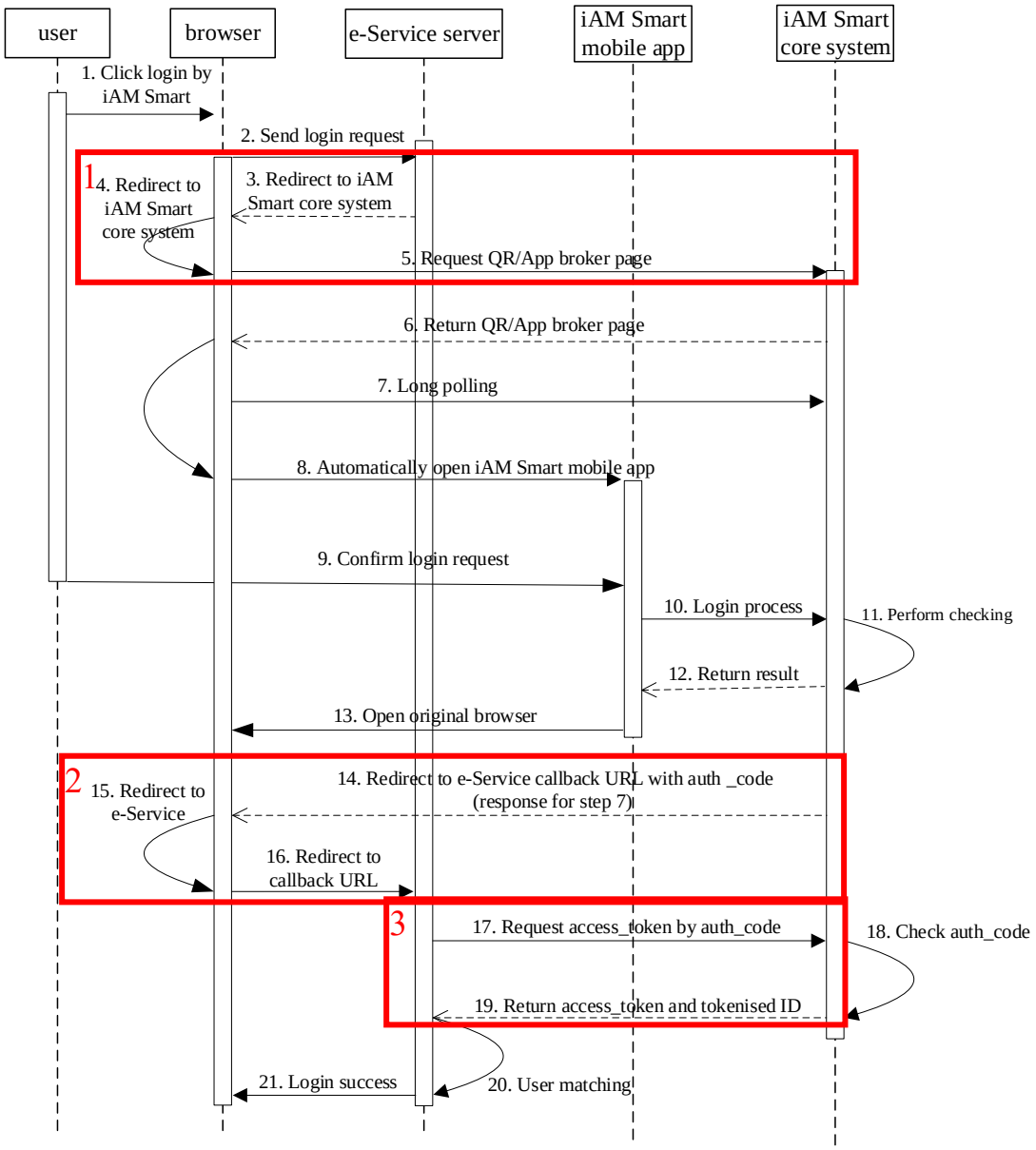

Figure-5 Authentication (Online Service Website in Same Device)

APIs interactions between online service servers and the “iAM Smart” System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Request QR Page	3.4.1
2	Callback with authCode to Online Service Server	3.4.4
3	Request accessToken & Tokenised ID	3.4.5

3.3.2.2 Initiated from "iAM Smart" mobile App (Direct Login v2)

The sequence diagram below shows how an Online Service perform Direct Login from "iAM Smart" mobile App.

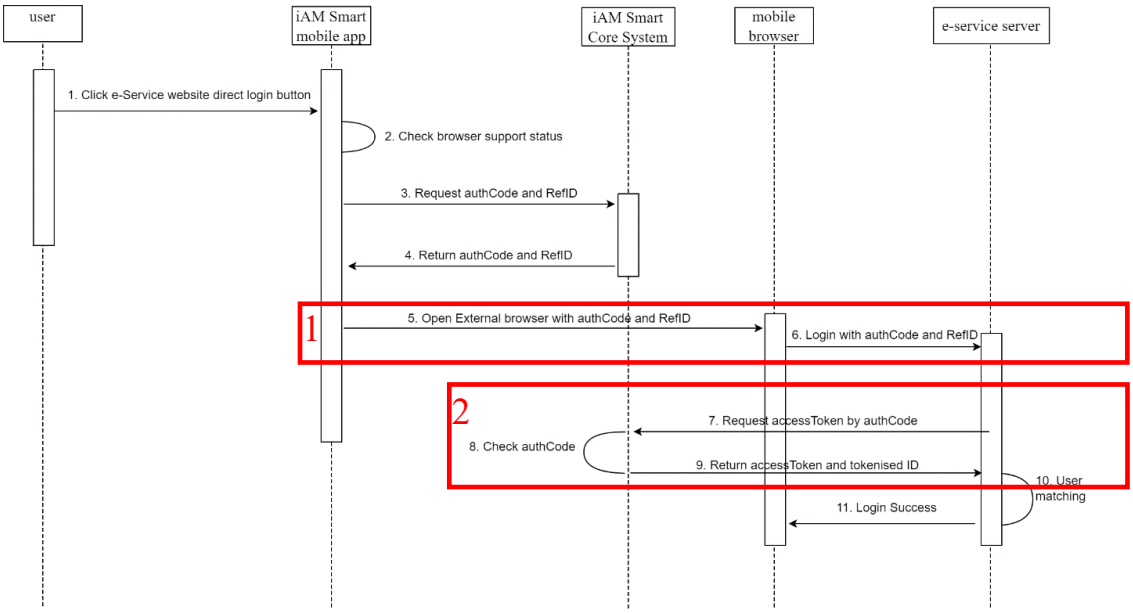

Figure-6 Authentication (Direct login v2 from "iAM Smart" to Online Service)

APIs interactions between online service servers and the "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Callback with AuthCode to Online Service Server (Direct Login v2)	3.4.6
2	Request accessToken & Tokenised ID	3.4.5

3.3.3 Authentication (Online Service App in Different Device)

The sequence diagram below shows how online service mobile application leverages the "iAM Smart" System to perform user authentication when the "iAM Smart" Mobile App is installed on a separated mobile device.

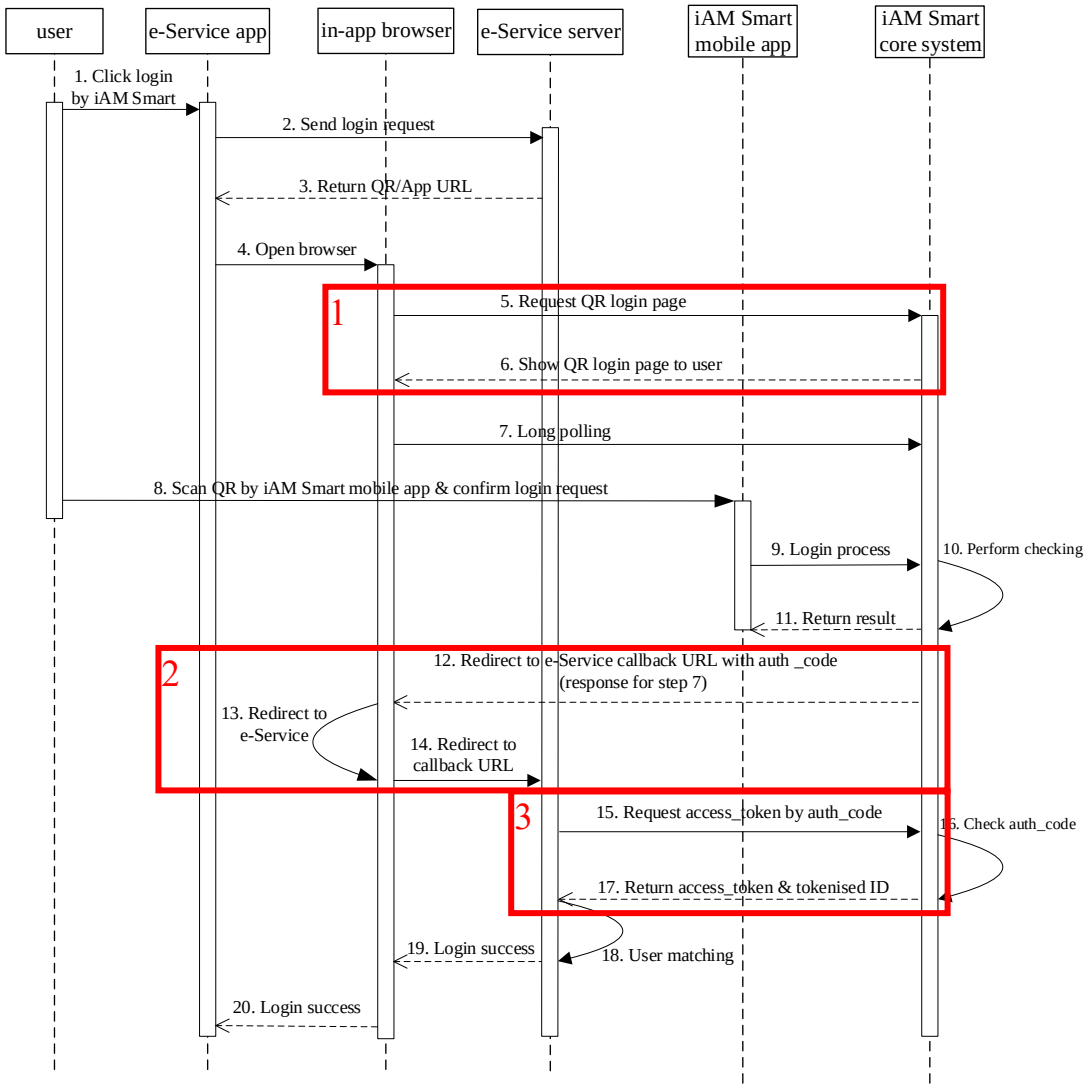

Figure-7 Authentication (Online Service App in Different Device)

APIs interactions between online service servers and the "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Request QR Page	3.4.1
2	Callback with authCode to Online Service Server	3.4.4

3	Request accessToken & Tokenised ID	3.4.5

3.3.4 Authentication (Online Service App in Same Device)

3.3.4.1 Initiated from Online Service App

The sequence diagram below shows how an online service mobile application leverages the “iAM Smart” System to perform user authentication when the “iAM Smart” Mobile App is installed in the same mobile device.

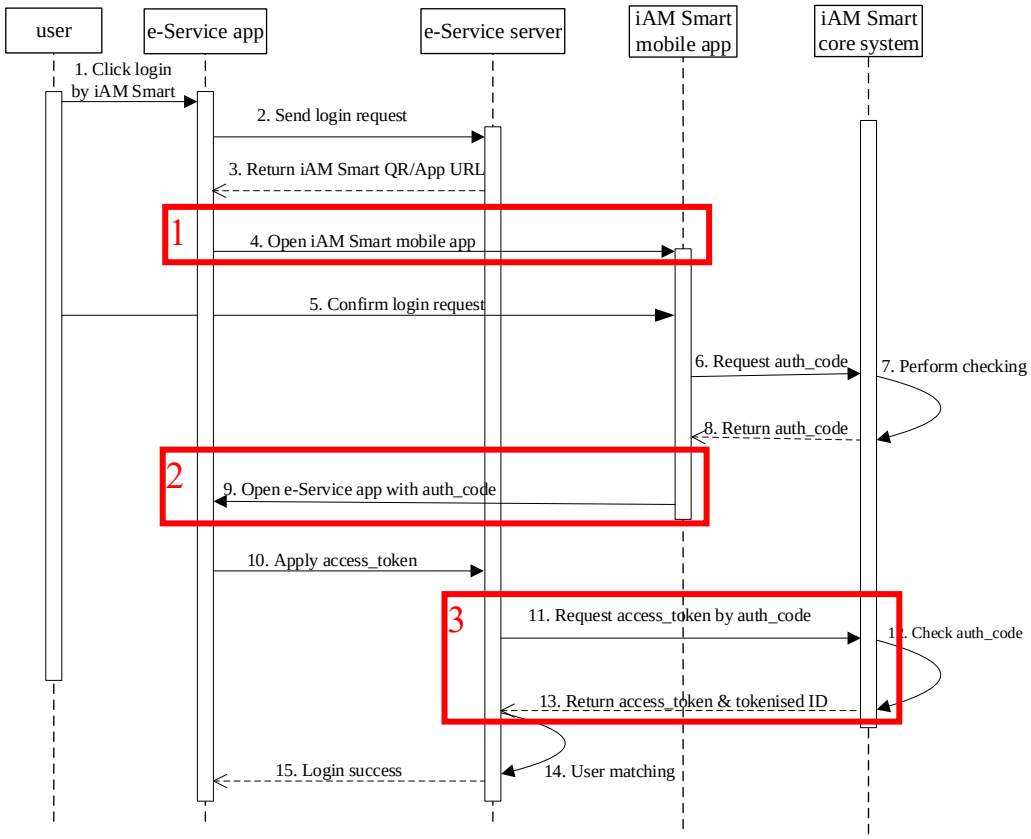

Figure-8 Authentication (Online Service App in Same Device)

APIs interactions between online service servers and the "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

There are two API versions (i.e. appv1[5] and appv2) available for Online Service App to open the "iAM Smart" Mobile App for Authentication (app-to-app scenario) .

For the new implementation of authentication, it is recommended to adopt the appv2 API in the Online Service App. For Android, this API enables "iAM Smart" to securely transmit the authentication code and launch the Online Service App using the Android package name. The feature of the new API is that it does not rely on App Link, making it compatible with both Android devices that have and do not have Google Mobile Services (GMS) enabled. For iOS, both appv1 and appv2 leverage Universal Link for "iAM Smart" to securely transmit the authentication code and launch the Online Service App.

For the existing implementation of authentication, the Online Service can either continue using the appv1, or choose to migrate to the appv2 API.

No.	API Name	API Reference (Section)
1	Open the "iAM Smart" Mobile App for Authentication (appv1)	3.4.2
	Open the "iAM Smart" Mobile App for Authentication (appv2)	3.4.8
2	Callback with authCode to Online Service App	3.4.3
3	Request accessToken & Tokenised ID	3.4.5

3.3.4.2 Initiated from "iAM Smart" mobile App (Direct Login v2 (App))

The sequence diagram below shows how an online service perform Direct Login from "iAM Smart" mobile App.

[5] appv1 is previously named Open the "iAM Smart" Mobile App for Authentication (appv1) which is the existing API that makes use of deep linking to redirect users to process authentication action in the "iAM Smart" Mobile App

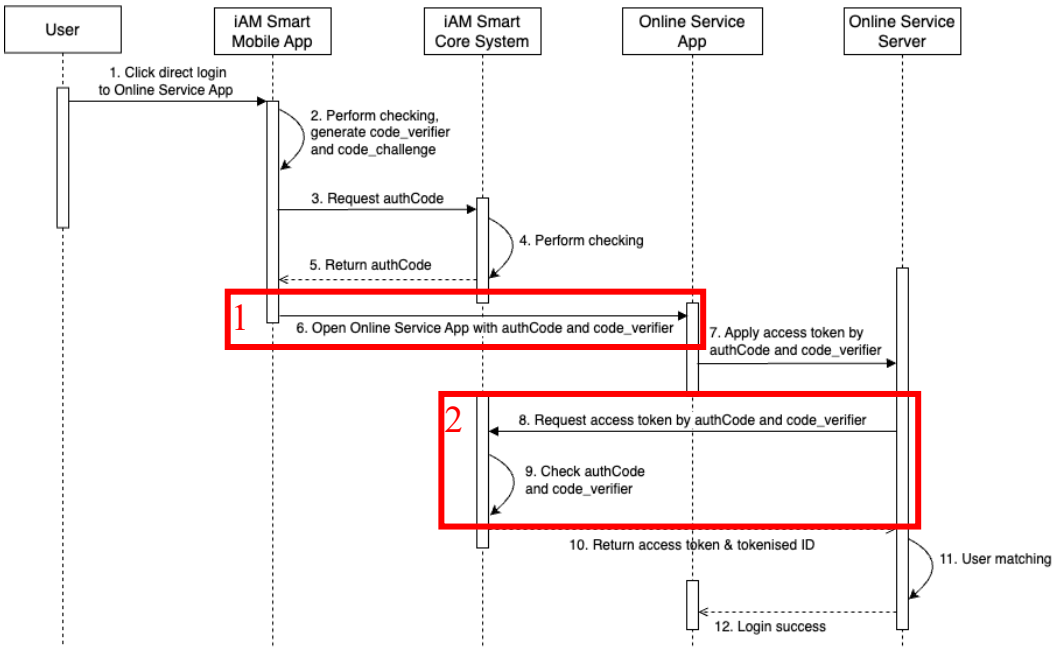

Figure-9 Authentication (Online Service App in Same Device)

APIs interactions between Online Service System and the "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Callback with authCode and code_verifier to Online *Service App*	3.4.9
2	Request accessToken & Tokenised ID	3.4.5

3.3.5 Verifying CCIC User

The sequence diagram below shows how Online Services verify whether the "iAM Smart" user is a holder of a Consular Corps Identity Card ("CCIC") after performing the authentication process.

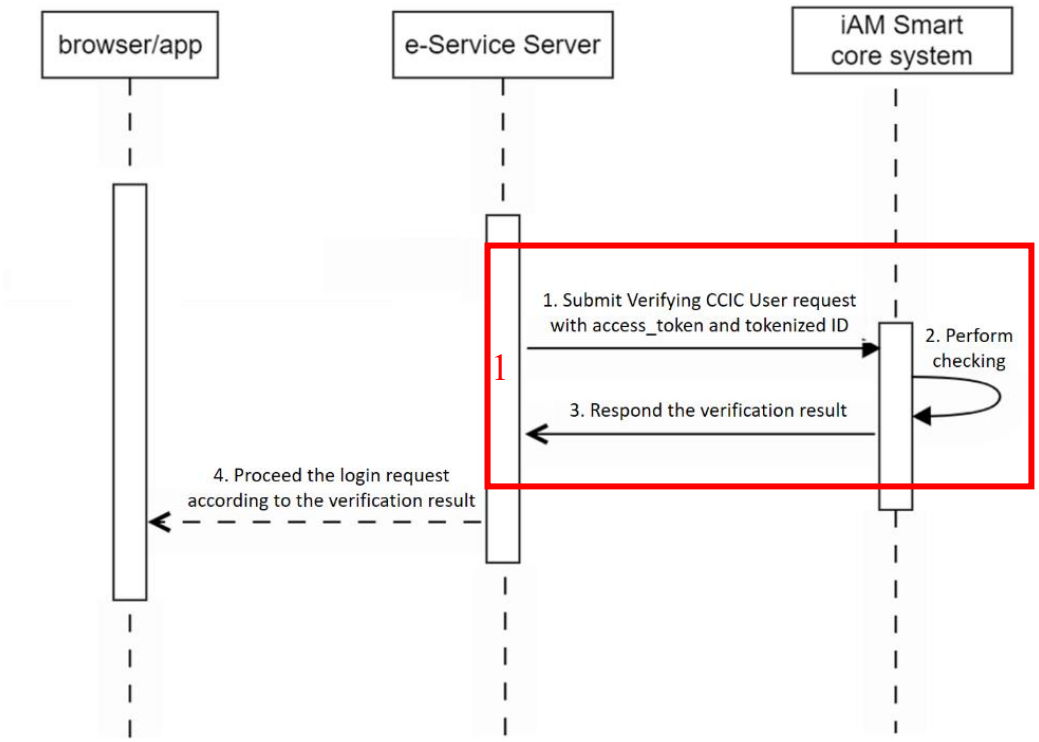

Figure-10 Verify whether the "iAM Smart" user is a CCIC holder

API interaction between online service server and the "iAM Smart" System is marked in red. Technical details of the respective API can be found in the following section.

No.	API Name	API Reference (Section)
1	Verify CCIC User	3.4.7

3.4 API Implementation Details

3.4.1 Request QR Page

- **API Description**

Name	Description
Service Full Name	Request QR Page
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/auth/getQR`
Request Type	GET
Service Version	1.0.0
Description of Service	Online service calls this API to get the QR/App broker page or QR page. After the user authorises login or anonymous request on the "iAM Smart" Mobile App, the page will be redirected to the `redirectURI` with `authCode` and `state` parameters. If the user denies it, only the `state` parameter will be redirected.

- **Request Parameters**

Parameter	Type	Presence	Description
`clientID`	String	Required	Online service client identifier. The clientID will be assigned to online service at the initial registration. Please refer to the self-service portal.
`responseType`	String	Required	The value MUST be set to `code`.
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification. Except (App_Link and App_Scheme).
`redirectURI`	String	Required	Callback redirect URI. The value should be URL encoded and registered in the self-service portal.
`scope`	String	Required	The value should be URL encoded. Specify the scope of "iAM Smart" functions: - `eidapi_auth` - `eidapi_profiles` - `eidapi_formFilling` - `eidapi_sign` - `eidapi_fr`

			- `eidapi_bulksign` - `eidapi_sua` The value of the scope parameter is expressed as a list of space-delimited, case-sensitive strings.
`lang`	String	Optional	Language to display: en-US, zh-HK, or zh-CN. If this parameter is not specified, zh-HK will be shown.
`state`	String	Optional	If the `state` parameter is presented in the request message, the same state value will be returned to online service during the callback. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.
`brokerPage`	Boolean	Optional	If `brokerPage` is set to `true`, Universal Link (iOS) / App Link (Android) will be leveraged to open the "iAM Smart" Mobile App. This feature is useful for online service supporting mobile web versions while triggering the "iAM Smart" Mobile App or showing a QR page automatically. (i.e. Show QR page without detecting whether the "iAM Smart" Mobile App is installed). The default value is `false`.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
GET
https://<iAM_Smart_domain>/api/v1/auth/getQR
?clientID=Online Service1
&responseType=code
&source=Android_Chrome
```

```
&redirectURI=https%3A%2F%2Frp_domain%2Frp_context%2Fcall_back_endpoint
&scope=eidapi_auth%20eidapi_formFilling%20eidapi_profiles
&lang=en-US
&state=eb9b7b8eddd5
```

- **Response Parameters**

N/A

3.4.2 Open the "iAM Smart" Mobile App for Authentication (appv1)

- **Using URL Scheme**

Name	Description
Service Full Name	Open "iAM Smart" Mobile App for authentication
URI (as in RESTFUL API)	<"iAM Smart" app URL scheme>://auth
Description of Service	The URL scheme makes use of deep linking to redirect users to process authentication action in the "iAM Smart" Mobile App. The URL schemes are supported on the iOS and Android versions of the "iAM Smart" Mobile App. *Remark: appv1 does not allow one client id to support multiple app.*

- **Request Parameters**

Parameter	Type	Presence	Description
clientID	String	Required	Online service client identifier. The clientID will be assigned to online service at the initial registration.
responseType	String	Required	the value MUST be set to code.
source	String	Required	App_Link/ App_Scheme Please use App_Link unless the support team approves.
redirectURI	String	Required	Callback redirect URI. The value should be URL encoded and registered in the self-service portal.
scope	String	Required	The value should be URL encoded. Specify the scope of "iAM Smart" functions: - eidapi_auth - eidapi_profiles

			- eidapi_formFilling - eidapi_sign - eidapi_fr - eidapi_bulksign The value of the scope parameter is expressed as a list of space-delimited, case-sensitive strings.
state	String	Optional	If state parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://auth
?clientID=Online Service1
&responseType=code
&source=Android_Chrome
&redirectURI=https%3A%2F%2Frp_domain%2Frp_context%2Fcall_back_endpoint
&scope=eidapi_auth%20eidapi_formFilling%20eidapi_profiles
&state=eb9b7b8eddd5
// Note: The values in scopes are space-delimited with URL Encoded.
```

3.4.3 Callback with authCode to Online Service App

- **URL Scheme and Package Name**

Name	Description
Service Full Name	Callback with authCode to Online Service App
URL Scheme (iOS and Android)	`<Universal / App Link>`. The online service custom app scheme or Universal / App Link should be registered with the "iAM Smart" System during onboarding.
Package name (Android only)	`<package name>` and `<activity class name>`. The online service package name must be registered in the self-service portal. *Remark: Direct Login v2 (App) and appv2 API adopts package name verification instead of App Link.*
Description of Service	For appv1 API, the Online Service App will be invoked and launched by Universal / App Link. It makes use of deep linking to redirect users to the Online Services app. App Link can only work for mobile devices with Google Mobile Services (GMS). For appv2 API, the Android Online Service App will be invoked and launched by the package name. It make use of intent to redirect users to the Online Service app. The iOS Online Service App will be invoked and launched via Universal link. The Universal / App Link with the landing location as well as the package name plus activity class name must be registered in the self-service portal and enabled by the support team.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **URL Scheme Parameters**

Parameter	Type	Presence	Description

businessID	String	Required (Conditional)	businessID will only be returned in anonymous workflows. businessID is a unique identifier for online service to differentiate different requests. Online service can use the businessID to relate the callback message with the original request. Maximum Length: 36
code	String	Required (Conditional)	The authorisation code generated by the "iAM Smart" System. The authorisation code will be expired in 60 seconds after issuance. Online service MUST NOT use the authorisation code more than once. An error message will be returned if an authorisation code is expired or re-used.
state	String	Optional	If the state parameter has been presented in the request message, the same state value will be returned. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random value.
error_code	String	Required (Conditional)	Return error code when exception.

- **Example URL Scheme**

Allow

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
?code=0ad186353c424c64897fcc00445c9ba1
&state=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
?error_code=D20001
&state=eddd527b6
```

- **Example Package Name**

Allow

```
Intent ii=new Intent(<package name>, <activity class name>);
ii.putExtra("code", "0ad186353c424c64897fcc00445c9ba1");
ii.putExtra("state", "eddd527b6");
startActivity(ii);
```

Deny

```
Intent ii=new Intent(<package name>, <activity class name>);
ii.putExtra("error_code", "D20001");
ii.putExtra("state", "eddd527b6");
startActivity(ii);
```

3.4.4 Callback with authCode to Online Service Server

- **API Description**

Name	Description
Service Full Name	Callback with authCode to Online Service Server
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	GET
Service Version	V1.0.0
Description of Service	This callback is used to pass `authCode` to online service Server. The URI must be registered in the self-service portal.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online service could treat the request as failed if it doesn't get callback within 18 minutes

- **Callback Parameters**

Parameter	Type	Presence	Description

businessID	String	Required (Conditional)	businessID will only be returned in anonymous workflows. businessID is a unique identifier for online service to differentiate different requests. Online service can use the businessID to relate the callback message with the original request. Maximum Length: 36
code	String	Required (Conditional)	The authorisation code is generated by the "iAM Smart" System. The authorisation code will be expired 60 seconds after issuance. Online service MUST NOT use the authorisation code more than once. An error message will be returned if an authorisation code is expired or re-used.
state	String	Optional	If the state parameter has been presented in the request, the same state value will be returned during the callback. It is used to prevent the CSRF attack. The value of the state is defined by online service and it should be a secure random value.
error_code	String	Required (Conditional)	Return error code when exception.

- **Example Callback**

Allow

```
// Line breaks are for legibility only.
GET
https://<call_back_endpoint>
?code=0ad186353c424c64897fcc00445c9ba1
&state=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
GET
https://<call_back_endpoint>
?error_code=D20001
&state=eddd527b6
```

3.4.5 Request accessToken & Tokenised ID

- **API Description**

Name	Description
Service Full Name	Request access token and tokenised ID with authCode
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/auth/getToken`
Request Type	POST
Service Version	1.0.0
Description of Service	Online service uses this API to retrieve the access token and Tokenised ID (openID). An authorisation code is necessary during the process. The `accessToken` and `openID` will be used to call corresponding "iAM Smart" services subsequently.

- **Request Parameters**

Parameter	Type	Presence	Description
`code`	String	Required	The authorisation code is received from the authorisation server. It can only be used once.
`code_verifier`	String	Required (Conditional)	This is required for Direct Login v2 (App). The PKCE code verifier received from "iAM Smart" App during login request.
`isDirectLoginV2`	Boolean	Required (Conditional)	Online service must configure separate callback endpoint (different from the one for normal login and Direct Login v1) to receive the authorisation code ("authCode") for Direct Login v2. Online service must submit the parameter "isDirectLoginV2" and set this value to *"true"* if the authCode is received by the callback endpoint for Direct Login v2. If the authCode for normal login or Direct Login v1 is received from other callback endpoints, online service can consider omitting this parameter or return a "`false`" value. The default value is "`false`".
`grantType`	String	Required	the value MUST be set to `authorization_code`.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/auth/getToken
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body (authCode from Direct Login v2)
{
  "code": "xxxa42e76bf4cb0846a68e6d83d6096",
  "isDirectLoginV2": "true"
  "grantType": "authorization_code"
}

// Unencrypted Request Body(authCode from other)
{
  "code": "xxxa42e76bf4cb0846a68e6d83d6096",
  "grantType": "authorization_code"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
accessToken	String	Required	accessToken value can be used multiple of times before expiry.
tokenType	String	Required	Token type, support "Bearer" only.
issueAt	Long	Required	The accessToken issue time is expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
expiresIn	Long	Required	The lifetime in milliseconds of the token. The value may vary for different Online Services.

`openID`	String	Required	Tokenised ID, uniquely generated for each user of each online service website or mobile application.
`lastModifiedDate`	Long	Required	The datetime of the user complete registration at "iAM Smart" System. The value will be updated when either of the following is valid:- (1) If any one of the following verified data are changed. • English name • Chinese name (* not applicable if it was marked as unverified during registration) • Gender • Date of birth (2) User re-register "iAM Smart" after "iAM Smart" de-registration. The modification time will be expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
`userType`	String	Required	`default` or `sign` `default`: "iAM Smart" user `sign`: "iAM Smart+" user (digital signing capability)
`scope`	String	Required	The scope of the token. Please refer to the corresponding section specified in each API function.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "accessToken": "0ad186353c424c64897fcc00445c9ba1",
    "tokenType": "Bearer",
    "issueAt": 1557053922938,
```

```
    "expiresIn": 14400000,
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
    "lastModifiedDate": 1560849218006,
    "userType": "sign",
    "scope": "eidapi_auth eidapi_formFilling"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D40004",
  "message": "authCode not exist or expired",
}
```

3.4.6 Callback with AuthCode to Online Service Server (Direct Login v2)

- **API Description**

Name	Description
Service Full Name	Callback with authCode to Online Service Server for direct login v2 of Service Catalogue
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	GET
Service Version	V2.0.0
Description of Service	This callback is used to pass `authCode` to online service Server. The online service MUST hold a separate callback endpoint to receive the AuthCode for Direct Login V2. The URI must be registered in the self-service portal (ESP) and approved by the support team, and online service can view the configuration in the ESP.

- **Callback Parameters**

Parameter	Type	Presence	Description
`code`	String	Required (Conditional)	The authorisation code is generated by the "iAM Smart" System. The authorisation code will be expired in **30** seconds after issuance. Online Service MUST NOT use the authorisation code more than once. An error message will be returned if an authorisation code is expired or re-used.
`RefID`	String	Required (Conditional)	A unique value for troubleshooting when necessary.
`error_code`	String	Required (Conditional)	Return error code to online service pre-defined fallback page when exception occurred.

- **Example Callback**

Allow

```
// Line breaks are for legibility only.
GET
https://<call_back_endpoint>
?code=0ad186353c424c64897fcc00445c9ba1
&RefID=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
GET
https://<online service fallback endpoint>
?error_code=D20008
```

- **Online Service onboarding**

The following configuration shall be configured and approved in the system.

Title	Description
Callback with authCode to Online Service Server for direct login v2 (en)	Receive authorisation code of direct login (web) with English language
Callback with authCode to Online Service Server for direct login v2 (tc)	Receive authorisation code of direct login (web) with Traditional Chinese language
Callback with authCode to Online Service Server for direct login v2 (sc)	Receive authorisation code of direct login (web) with Simplified Chinese language
Scopes for the direct login v2	The value of the scope parameter is expressed as a list of space-delimited, case-sensitive strings. D20012 is returned if requested scopes are not on the approved list. Online service can review their approved scope(s) in the self-service portal.
Fallback URL to Online Service Server for direct login v2 (en)	Provide a fallback for direct login (web) in the English language when the user fails to perform direct login via the "iAM Smart" service catalogue; or provide direct access to the online service in English without a login requirement
Fallback URL to Online Service Server for direct login v2 (tc)	Provide a fallback for direct login (web) in the Traditional Chinese language when the user fails to perform direct login via the "iAM Smart" service catalogue; or provide direct access to the online service in Traditional Chinese without a login requirement
Fallback URL to Online Service Server for direct login v2 (sc)	Provide a fallback for direct login (web) in the Simplified Chinese language when the user fails to perform direct login via the "iAM Smart" service catalogue; or provide direct access to the online service in Simplified Chinese without a login requirement

3.4.7 Verify CCIC User

- **API Description**

Name	Description
Service Full Name	Verify whether the "iAM Smart" user is a CCIC holder
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/ccic/verifyIsCcic`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to initiate a request for verifying CCIC user by sending `accessToken` and `openID` to "iAM Smart" System. Online service shall notify "iAM Smart" support team to grant the access right before using this API.

- **Request Parameters**

Parameter	Type	Presence	Description
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/ccic/verifyIsCcic
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
    "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
verifyResult	Boolean	Required	If the “iAM Smart” user is a CCIC holder, the "true" value will be returned to Online Service.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
        "verifyResult": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {accessToken}"
}
```

3.4.8 Open the “iAM Smart” Mobile App for Authentication (appv2)

- **Using URL Scheme**

Name	Description
Service Full Name	Open “iAM Smart” Mobile App for authentication
URI (as in RESTFUL API)	<“iAM Smart” app URL scheme>://v2_auth
Service Version	V2.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to process authentication action in the “iAM Smart” Mobile App. The URL schemes are supported on the iOS and Android versions of the “iAM Smart” Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description

clientID	String	Required	Online service client identifier. The clientID will be assigned to online service at the initial registration.
responseType	String	Required	the value MUST be set to code.
source	String	Required	App_Link (iOS) or App_Package (Android)
clientRedirectURI	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (“,”) are not accepted.
packageName	String	Required (Android only)	Package name of the Online Service App for callback use.
activityClass	String	Required (Android only)	Activity class name of the Online Service App for callback use.
activityParams	String	Optional (Android only)	Optional params used during callback.
scope	String	Required	The value should be URL encoded. Specify the scope of “iAM Smart” functions: - eidapi_auth - eidapi_profiles - eidapi_formFilling - eidapi_sign - eidapi_fr - eidapi_bulksign The value of the scope parameter is expressed as a list of space-delimited, case-sensitive strings.
state	String	Optional	If state parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and

			hyphens are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://v2_auth
?clientID=Online Service1
&responseType=code
&source=App_Package
&packageName=com.onlineservice.myapp
&activityClass=callback_activity
&activityParams=callback_param
&scope=eidapi_auth%20eidapi_formFilling%20eidapi_profiles
&state=eb9b7b8eddd5
// Note: The values in scopes are space-delimited with URL Encoded.
```

3.4.9 Callback with authCode and code_verifier to Online Service App (Direct Login v2)

- **URL Scheme and package name**

Name	Description
Service Full Name	Callback with authCode and code_verifier to Online Service App
URL Scheme (iOS only)	`<Universal Link>`. The online service Universal Link should be registered with the "iAM Smart" System during onboarding.
Package name (Android)	`<package name>` and `<activity class name>`. The online service package name and class name should be registered with the "iAM Smart" System during onboarding.
Description of Service	This callback is used to pass `authCode` and `code_verifier` to online service App.

- **URL Scheme / activity Parameters**

Parameter	Type	Presence	Description
`code`	String	Required	The authorisation code generated by the "iAM Smart" System. The authorisation code will be expired in 30 seconds after issuance. Online service MUST NOT use the authorisation code more than once. An

			error message will be returned if an authorisation code is expired or re-used.
code_verifier	String	Required	The PKCE code verifier generated by the "iAM Smart" System.

- **Example Universal Link**

```
<Universal Link>://<landing location>?code=0ad186353c424c64897fcc00445c9ba1
&code_verifier =eddd527b6
```

- **Example Package Name**

```
Intent ii=new Intent(<package name>, <activity class name>);
ii.putExtra("code", "0ad186353c424c64897fcc00445c9ba1");
ii.putExtra("code_verifier", "DxjdFp0vKYCY4F0DE6M3eEadLxcJBTb6k4LZ9J5z");
startActivity(ii);
```

3.4.10 Get last login status

- **API Description**

Name	Description
Service Full Name	A unique API for eService to get the last login status for user
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/auth/account/getLastLoginStatus`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to query the login method used by users for the last time they logged on (PIN, Biometric) by sending `accessToken` and `openID` to "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/auth/account/getLastLoginStatus
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
    "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description

loginType	String	Required	If you login back to “PIN” using pin code, and “BIO” if login with biometrics.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
        "loginType ":"PIN"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {accessToken}"
}
```

3.5 API Deprecation

The following section lists the deprecation period of the respective APIs. The Deprecation Date is the start date of the deprecation period, while Shutdown Date is the end date of the deprecation period. After the Shutdown Date, the API request will no longer be available.

3.5.1 Getting Profile

Version	Deprecation Date	Shutdown Date	Details
V1.0.0	1 January 2024	31 December 2025	On or after 31 December 2025, existing online service will not be able to reach the endpoint (api/v1/account/auth/profile/initiateRequest). Online service has to update the testing and production application form for the migration process. Online service shall migrate to Profiles API (https://<iAM_Smart_domain>/api/v1/profiles) by requesting the parameter in profile fields for user identity verification purpose. Details of the Profiles API can be found in Section 4.1.4.1.

3.5.1.1 Request Profile

- **API Description**

Name	Description
Service Full Name	Request profile
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/account/auth/profile/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API making request for profile by sending `accessToken` and `openID` to "iAM Smart" System.
Deprecation Date	1 January 2024
Shutdown Date	31 December 2025

- **Request Parameters**

<table>
<tr><th>Parameter</th><th>Type</th><th>Presence</th><th>Description</th></tr>
<tr><td>businessID</td><td>String</td><td>Required</td><td>businessID is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.</td></tr>
<tr><td>accessToken</td><td>String</td><td>Required</td><td>accessToken value</td></tr>
<tr><td>openID</td><td>String</td><td>Required</td><td>Tokenised ID value</td></tr>
<tr><td>source</td><td>String</td><td>Required</td><td>Request initiator. The supported source values can be found in Appendix B of this specification.</td></tr>
<tr><td>redirectURI</td><td>String</td><td>Required</td><td>Callback URI.</td></tr>
<tr><td>state</td><td>String</td><td>Optional</td><td>If state parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.</td></tr>
<tr><td>profileFields</td><td>Array</td><td>Required</td><td>Specify the profile fields to be requested. The available profileFields are as follows:
<table>
<tr><th>profileFields</th><th>Description</th></tr>
<tr><td>idNo</td><td>ID number</td></tr>
<tr><td>enName</td><td>English name</td></tr>
<tr><td>chName</td><td>Chinese name</td></tr>
<tr><td>birthDate</td><td>Date of birth</td></tr>
<tr><td>gender</td><td>gender</td></tr>
</table>
</td></tr>
</table>

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
```

```
https://<iAM_Smart_domain>/api/v1/auth/profile/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
 "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
 "source": "Android_Chrome",
 "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
 "state": "eddd527b6",
 " profileFields": ["idNo", "enName"]
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If user completes authentication by scanning QR code, then "true" will be returned to online service.
`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger “iAM Smart” Mobile App when authByQR is `false`.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {profileFields}"
}
```

3.5.1.2 Open "iAM Smart" Mobile App for Getting Profile

- **Using URL Scheme**

Name	Description
Service Full Name	Open "iAM Smart" Mobile App for getting profile
URI (as in RESTFUL API)	<"iAM Smart" app URL scheme>://profile
Service Version	V1.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific profile request in "iAM Smart" Mobile App. The URL schemes are supported on iOS and Android versions of "iAM Smart" Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
ticketID	String	Required	TicketID is a unique identifier provided by "iAM Smart" System, online service retrieves ticketID while requesting the get profile function. It is an ASCII string with a length of less than or equal to 36 chars.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://profile
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

3.5.1.3 Callback to Receive "iAM Smart" Profile

- **API Description**

Name	Description
Service Full Name	Callback to Receive "iAM Smart" Profile
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	V1.0.0
Description of Service	This callback used to return user profile information to online service upon consent.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. Online service can use the `businessID` to relate the callback message with the original request.
`state`	String	Optional	If the `state` parameter has been present in the request message, the exact value of `state` will be returned. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random value.
`idNo`	JSON Object	Optional	ID number
`enName`	JSON Object	Optional	English name
`chName`	JSON Object	Optional	Chinese name
`birthDate`	String	Optional	Date of birth
`gender`	String	Optional	gender
`chNameVerified`	String	Optional	ImmD characters code point will be returned to online service if Chinese name verified.

- **Example Callback**

The maximum length of idNo, prefix, enName, chName and chNameVerified can be found in Appendix A of this specification. Most of the fields are defined in DPO common schema. More information about common schema can be found at https://www.digitalpolicy.gov.hk/en/our_work/data_governance/policies_standards/interoperability_framework/common_schemas/.

```
// Line breaks are for legibility only.
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>
// Callback Body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "businessID": "3e47be25-66a6-43fb-89f6-7e2dd138aff8",
    "state": "unesidkd",
    "idNo": {
        "Identification": "A123456",
        "CheckDigit": "A"
    },
    "enName": {
        "UnstructuredName": "SAN, Chi Nan"
    },
    "chName": {
        "ChineseName": "申智能"
    },
    "birthDate": "19960000",
    "gender": "M",
    "chNameVerified": "申智能"
  }
}
```

4. FORM FILLING WITH SERVICE LOGIN

4.1 Profiles API

4.1.1 Overview

Online Services can request “Profiles” API to obtain “iAM Smart” user’s personal data, namely English Name, Chinese Name, HKID no., gender and date of birth for identity verificaton. This API also facilitates “iAM Smart” user to perform form filling by passing their personal information stored in “iAM Smart” to online service application swiftly. The profiles API enables online services to retrieve user profiles without an authorisation page like Form Filling(v2) API. This Profiles API is designed for GRO online service to realise a “single portal for online government services ".

The typical use cases cover account link-up, application form system, registration form, remote account onboarding, identity verification and address verification.

4.1.2 Prerequisite

- Applicable to Government and Related Organisations (GRO).
- Online service must undergo the authentication process (i.e. Request accessToken & Tokenised ID) in Section 3.4.5 in order to obtain the accessToken and openID as the input of Profiles API.
- “Profile fields” in Profiles API is for user identity verification purpose, while “e-ME fields” in Profiles API is for form filling usage.
- Online service shall refer to “Reference System Flow and User Interface Specifications for “iAM Smart” Adoption” for the detail UI/UX requirements.

4.1.3 Scope

Online service shall submit the application for using the following scope values to “iAM Smart” Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scope values in the Self-Service Portal (ESP).

Scope	Description
`eidapi_profiles`	Scope for Profiles

4.1.4 Use Cases and Scenarios

4.1.4.1 Obtain Profiles Information

The sequence diagram below shows how Online Services obtain user profile information right after authentication process.

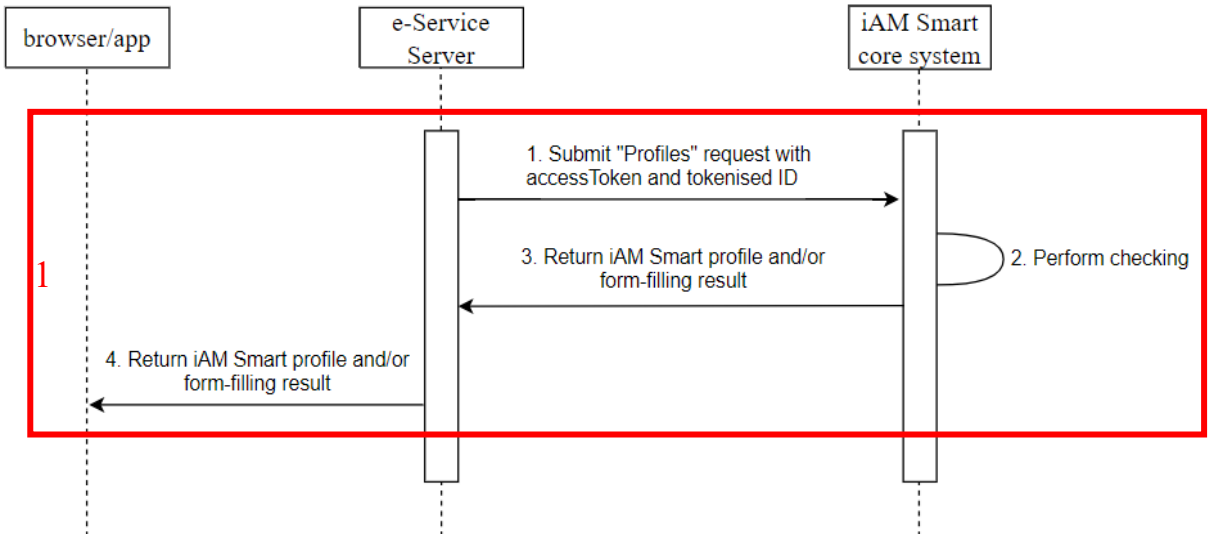

Figure-11 Profiles

APIs interactions between online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Obtain Profiles Information	4.1.5.1

4.1.5 API Implementation Details

4.1.5.1 Obtain Profiles Information

- **API Description**

Name	Description
Service Full Name	Request the personal information of the "iAM Smart" user
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/profiles`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to request the "iAM Smart" user account information (profileFields) for user identity verification purpose and request "e-ME" profile (include the "iAM Smart" user account information) (eMEFields) for form filling purpose.
Remark	Online service that would like to use this API to request the data fields MUST submit the application to "iAM Smart" Support Team in advance.

- **Request Parameters**

Parameter	Type	Presence	Description
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value
`eMEFields`	Array	Required (Conditional)	It is for user form-filling purposes. Specify the "e-ME" fields to be requested. If eMEFields and the following profileFields are not provided or are empty arrays, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned. The available eMEFields are as follows:

eMEFields	Description
idNo	ID number
prefix	prefix
enName	English name
chName	Chinese name
birthDate	Date of birth
gender	gender
maritalStatus	marital status
homeTelNumber	home telephone

<table>
<tr><td></td><td></td><td></td><td><table>
<tr><td></td><td>number</td></tr>
<tr><td>officeTelNumber</td><td>office telephone number</td></tr>
<tr><td>mobileNumber</td><td>mobile number</td></tr>
<tr><td>emailAddress</td><td>email address</td></tr>
<tr><td>residentialAddress</td><td>residential address</td></tr>
<tr><td>postalAddress</td><td>postal address</td></tr>
<tr><td>educationLevel</td><td>education level</td></tr>
<tr><td>addressDocInfo</td><td>provider name, retrieval date, owner name and address information related to an e-bill</td></tr>
<tr><td>addressDocFile</td><td>e-bill from an address data provider</td></tr>
</table></td></tr>
<tr><td>profileFields</td><td>Array</td><td>Required (Conditional)</td><td>It is for user identity verification purposes. Specify the profile fields to be requested. If profileFields and the above eMEFields are not provided or are empty arrays, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned. The available profileFields are as follows:<table>
<tr><th>profileFields</th><th>Description</th></tr>
<tr><td>idNo</td><td>ID number</td></tr>
<tr><td>enName</td><td>English name</td></tr>
<tr><td>chName</td><td>Chinese name</td></tr>
<tr><td>birthDate</td><td>Date of birth</td></tr>
<tr><td>gender</td><td>gender</td></tr>
</table></td></tr>
</table>

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/profiles
```

```
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
   "profileFields": ["idNo", "enName", "gender", "chName", "birthDate"],
   "eMEFields": ["mobileNumber", " emailAddress", "addressDocInfo"]
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
idNo	JSON Object	Optional	HKIC number
prefix	String	Optional	prefix
enName	JSON Object	Optional	English name
chName	JSON Object	Optional	Chinese name
birthDate	String	Optional	Date of birth
gender	String	Optional	gender
maritalStatus	String	Optional	marital status
homeTelNumber	JSON Object	Optional	home telephone number
officeTelNumber	JSON Object	Optional	office telephone number
mobileNumber	JSON Object	Optional	mobile number
emailAddress	String	Optional	email address
residentialAddress	JSON Object	Optional	residential address If the data is retrieved from CDEG, “addressProvider” field will also be included. The “providerCode” and “lastRetrievalDate” fields indicate where and when the data was updated.
postalAddress	JSON Object	Optional	postal address If the data is retrieved from CDEG, “addressProvider” field will also be included. The “providerCode” and

			"lastRetrievalDate" fields indicate where and when the data was updated.
`educationLevel`	String	Optional	education level
`chNameVerified`	String	Optional	ImmD characters code point will be returned to online service if Chinese name verified.
`addressDocInfo`	JSON Object	Optional	the data related to the obtained e-bill, including the provider name, retrieval date, owner name, service address and postal address. The sub-field `lastRetrievalDate` refers to the date and time at which "iAM Smart" obtained the e-bill. The schema of sub-fields is given in **Appendix A** of API Specification.
`addressDocFile`	JSON Object	Optional	the e-bill obtained from an address data provider. Sub-fields include the PDF e-bill file (`docFile`), the SHA256 hash (`docHash`) and the bill date of the e-bill (`billDate`). The schema of sub-fields is given in **Appendix A** of API Specification.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
   "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
   "code": "D00000",
   "message": "SUCCESS",
   "content": {
      "idNo": {
         "Identification": "A123456",
         "CheckDigit": "A"
      },
      "prefix": "Mr",
      "enName": {
```

```
        "UnstructuredName": "SAN, Chi Nan"
    },
    "chName": {
        "ChineseName": "申智能"
    },
    "chNameVerified": "申智能",
    // May be 19960000 or 19960100
    "birthDate": "19960128",
    "gender": "M",
    "maritalStatus": "S",
    "homeTelNumber": {
        "CountryCode": "852",
        "SubscriberNumber": "98765432"
    },
    "officeTelNumber": {
        "CountryCode": "1",
        "SubscriberNumber": "123456"
    },
    "mobileNumber": {
        "CountryCode": "1",
        "SubscriberNumber": "98765432"
    },
    "emailAddress": "scn@digitalpolicy.gov.hk",

     // Three different tags exist in residentialAddress
     //  and the returned value  can be either one of the following
     //  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress

     // Both ChiPremisesAddress and EngPremisesAddress contain either
     // 1): Standard/Village Address
     // 2): Lot Address

    //   Example of ChiPremisesAddress with Standard/Village Address
    "residentialAddress": {
        "ChiPremisesAddress": {
            "Region": "香港",
            "ChiDistrict": {
                "DcDistrict": "WC",
```

```
                "Sub-district": "灣仔"
            },
            "BuildingName": "灣仔政府大樓",
            "ChiEstate": {
                "EstateName": "華富",
                "ChiPhase": {
                    "PhaseName": "華清"
                }
            },
            "ChiStreet": {
                "StreetName": "港灣道",
                "BuildingNoFrom": "12"
            },
            "ChiBlock": {
                "BlockDescriptor": "座",
                "BlockNo": "東"
            },
            "Chi3dAddress": {
                "ChiFloor": {
                    "FloorNum": "15"
                },
                "ChiUnit": {
                    "UnitDescriptor": "室",
                    "UnitNo": "A1"
                }
            }
        },

        //   If the data is retrieved from CDEG, "addressProvider" field
        //   will also be included.
        "addressProvider": {
            "providerCode": "wsd",
            "lastRetrievalDate": 1560853718006
        }
    },

    //   Example of EngPremisesAddress with Lot address
    "residentialAddress": {
```

```
        "EngPremisesAddress": {
            "EngLot": {
                "EngStructuredLot": {
                    "DdType": "DD",
                    "DdNo": "110",
                    "LotType": "LOT",
                    "LotNo": "157",
                    "LotSection1": "A",
                    "LotSubsection1": "1",
                    "LotSection2": "B",
                    "LotSubsection2": "1",
                    "LotSection3": "AA",
                    "LotSubsection3": "1",
                    "LotExtendPortionCode": "3"
                }
            }
        }
    },

    //   Example of FreeFormatAddress
    "residentialAddress": {
        "FreeFormatAddress": {
            "LanguageCode": "en",
            "AddressLine1": "Unit 2000, 200/F",
            "AddressLine2": "5033 Yitian Road, Futian CBD",
            "AddressLine3": "Futian district, Shenzhen"
        }
    },

     // Four different tags exist in postalAddress
    //  The returned value of postalAddress can be either one of the
    //  following
    //  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress,
    //  PostBoxAddress

    //  Apart from PostBoxAddress, the other three tags are equivalent to
    //  ResidentialAddress
```

```
//  PostBoxAddress contains either
     // 1): EngPostBox
     // 2): ChiPostBox

    // Example of postalAddress with EngPostBox
  "postalAddress": {
     "PostBoxAddress": {
          "EngPostBox": {
              "PoBoxNo": 24700,
              "PostOffice": "ABERDEEN POST OFFICE",
              "PostOfficeRegion": "HONG KONG"
         }
     }
  },
  "educationLevel": "T",
  // all the sub-fields come from the address data provider
  "addressDocInfo": {
     // name of the address data provider, possible values are:
     // The Hong Kong and China Gas Company Limited
     // CLP Power Hong Kong Ltd.
     // HK Electric
     // Water Supplies Department
     "enProviderName": "HK Electric",
     "tcProviderName": "港燈",
     "scProviderName": "港灯",
     // unstructured name of the e-bill owner
     "enUserName": "SAN, Chi Nan",
     "tcUserName": "申智能",
     // service address of the e-bill, at most 4 lines
     "enServiceAddress": {
        "addressLine1": "19/F, 213 Queen's Road East",
        "addressLine2": "Wanchai, Hong Kong"
     },
     "tcServiceAddress": {
        "addressLine1": "香港灣仔皇后大道東 231 號",
        "addressLine2": "胡忠大廈 19 樓"
     },
     // postal address of the e-bill, at most 4 lines
```

```
            "enPostalAddress": {
                "addressLine1": "19/F, 213 Queen's Road East",
                "addressLine2": "Wanchai, Hong Kong"
            },
            "tcPostalAddress": {
                "addressLine1": "香港灣仔皇后大道東 231 號",
                "addressLine2": "胡忠大廈 19 樓"
            },
            // last retrieval date of the e-bill
            // expressed in the number of milliseconds since 1970/1/1 GMT
            "lastRetrievalDate": 1560853718006
        }
        "addressDocFile": {
            // Base64 encoded string of the PDF e-bill file
            "docFile": "JVBERi0xLjUKJcK1wrXCtcK1CjEgMCBv…iag==",
            // SHA256 hash of the PDF e-bill file
            "docHash": "69e3…c4e5",
            // bill date of the e-bill
            // expressed in the number of milliseconds since 1970/1/1 GMT
            "billDate": 1560849218006
        }
    }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in  Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {accessToken}"
}
```

- **API Specific Error Code**

  Special Error Code for Profiles API

Code	Description
D20003	Invalid parameter – {accessToken | openID | eMEFields | profileFields}.

	For this error, please check the ESP for the approved datafields in eMEFields / profileFields in Profiles API. The error occurred may also due to the incorrect accessToken or openID provided.

- **API Specific Timeout**

Condition	Timeout value
Normal request	10 seconds
For request include “addressDocFile”	30 seconds

4.1.6 API Deprecation

The following section lists the deprecation period of the respective APIs. The Deprecation Date is the start date of the deprecation period, while Shutdown Date is the end date of the deprecation period. After the Shutdown Date, the API request will no longer be available.

4.1.6.1 Scope

Scope	Description
`eidapi_eMe`	Scope for Form Filling (v1)

4.1.6.2 Form Filling

Version	Deprecation Date	Shutdown Date	Details
V1.0.0	1 October 2022	31 December 2025	On or after 31 December 2025, existing online service will not be able to reach the endpoint (/api/v1/account/eme/initiateRequest). Online service has to update the testing and production application form for the migration process.

4.1.6.2.1 Request Form Filling (v1)

- **API Description**

Name	Description
Service Full Name	Request Form Filling
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/account/e`

	me/initiateRequest
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to request for form filling
Depreciation Date	1 October 2022
Shutdown Date	31 December 2025

- **Request Parameters**

Parameter	Type	Presence	Description
businessID	String	Required	businessID is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
accessToken	String	Required	accessToken value
openID	String	Required	Tokenised ID value
source	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.
redirectURI	String	Required	Callback URI.
state	String	Optional	If state parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
formName	String	Optional	The name of the form should be encoded in Unicode. Maximum Length: 255
formNum	String	Optional	The number of the form should be encoded in Unicode. Maximum Length: 20
formDesc	String	Optional	The description of the form should be encoded in Unicode.

<table>
<tr><td></td><td></td><td></td><td>Maximum Length: 255</td></tr>
<tr><td>eMEFields</td><td>Array</td><td>Required</td><td>Specify the “e-ME” fields to be requested. If eMEFields is not provided or is an empty array, the HTTP code 200 with error code D20002 (empty parameter {eMEFields}) will be returned.
The available eMEFields are as follows:<table><tr><th>eMEFields</th><th>Description</th></tr><tr><td>idNo</td><td>ID number</td></tr><tr><td>prefix</td><td>prefix</td></tr><tr><td>enName</td><td>English name</td></tr><tr><td>chName</td><td>Chinese name</td></tr><tr><td>birthDate</td><td>Date of birth</td></tr><tr><td>gender</td><td>gender</td></tr><tr><td>maritalStatus</td><td>marital status</td></tr><tr><td>homeTelNumber</td><td>home telephone number</td></tr><tr><td>officeTelNumber</td><td>office telephone number</td></tr><tr><td>mobileNumber</td><td>mobile number</td></tr><tr><td>emailAddress</td><td>email address</td></tr><tr><td>residentialAddress</td><td>residential address</td></tr><tr><td>postalAddress</td><td>postal address</td></tr><tr><td>educationLevel</td><td>education level</td></tr><tr><td>addressDocInfo</td><td>provider name, retrieval date, owner name and address information related to an e-bill</td></tr><tr><td>addressDocFile</td><td>e-bill from an address data provider</td></tr></table></td></tr>
</table>

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
```

```
POST
https://<iAM_Smart_domain>/api/v1/account/eme/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
 "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
 "source": "Android_Chrome",
 "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
 "state": "eddd527b6",
 "formName": "Online Service form Name",
 "formNum": "FormNO_2019001",
 "formDesc": "form description",
 "eMEFields": ["idNo", "enName", "birthDate", "gender", "chName",
 "addressDocInfo", "addressDocFile"]
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If user completes authentication by scanning QR code (i.e. different device), then "true" will be returned to Online Service.
`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger “iAM Smart” Mobile App when authByQR is `false`.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
```

```
  "content": {
    "authByQR": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {eMEFields}"
}
```

4.1.6.3 Open “iAM Smart” Mobile App for Form Filling

- **Using URL Scheme**

Name	Description
Service Full Name	Open “iAM Smart” Mobile App for form filling
URI (as in RESTFUL API)	For Form Filling v1: <“iAM Smart” app URL scheme>://eme
Service Version	V1.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific profile request in “iAM Smart” Mobile App. The URL schemes are supported on iOS and Android versions of “iAM Smart” Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
ticketID	String	Required	TicketID is a unique identifier provided by “iAM Smart” System, online service retrieves ticketID while requesting the Form Filling function. It is an ASCII string with a length of less than or equal to 36 chars.

- **Example Scheme**

```
// Line breaks are for legibility only.
<“iAM Smart” app URL scheme>://form-filling
```

```
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

4.2 Form Filling (v2) API

4.2.1 Overview

Similar to Profiles API, online service can initiate an authorisation request after service login with “iAM Smart” authentication API. The online service will receive the authorised data after the user consents to the authorisation page.

The typical use cases cover account link-up, application form system, registration form, remote account onboarding, identity verification and address verification.

4.2.2 Prerequisite

- Applicable to non GRO organsiations
- “Profile fields” in Form Filling API is for user identity verification purpose, while “e-ME” fields in Form Filling API is for form filling usage.
- Online service shall refer to “Reference System Flow and User Interface Specifications for “iAM Smart” Adoption” for the detail UI/UX requirements.

4.2.3 Scope

Online service shall submit the application for using the following scope values to “iAM Smart” Support Team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_formFilling`	Scope for Form Filling(v2)

4.2.4 Use Cases and Scenarios

4.2.4.1 Form Filling (Online Service Website/App in Different Device)

The sequence diagram below shows how an authenticated user performs form filling when Online Service and the "iAM Smart" Mobile App are running in different devices.

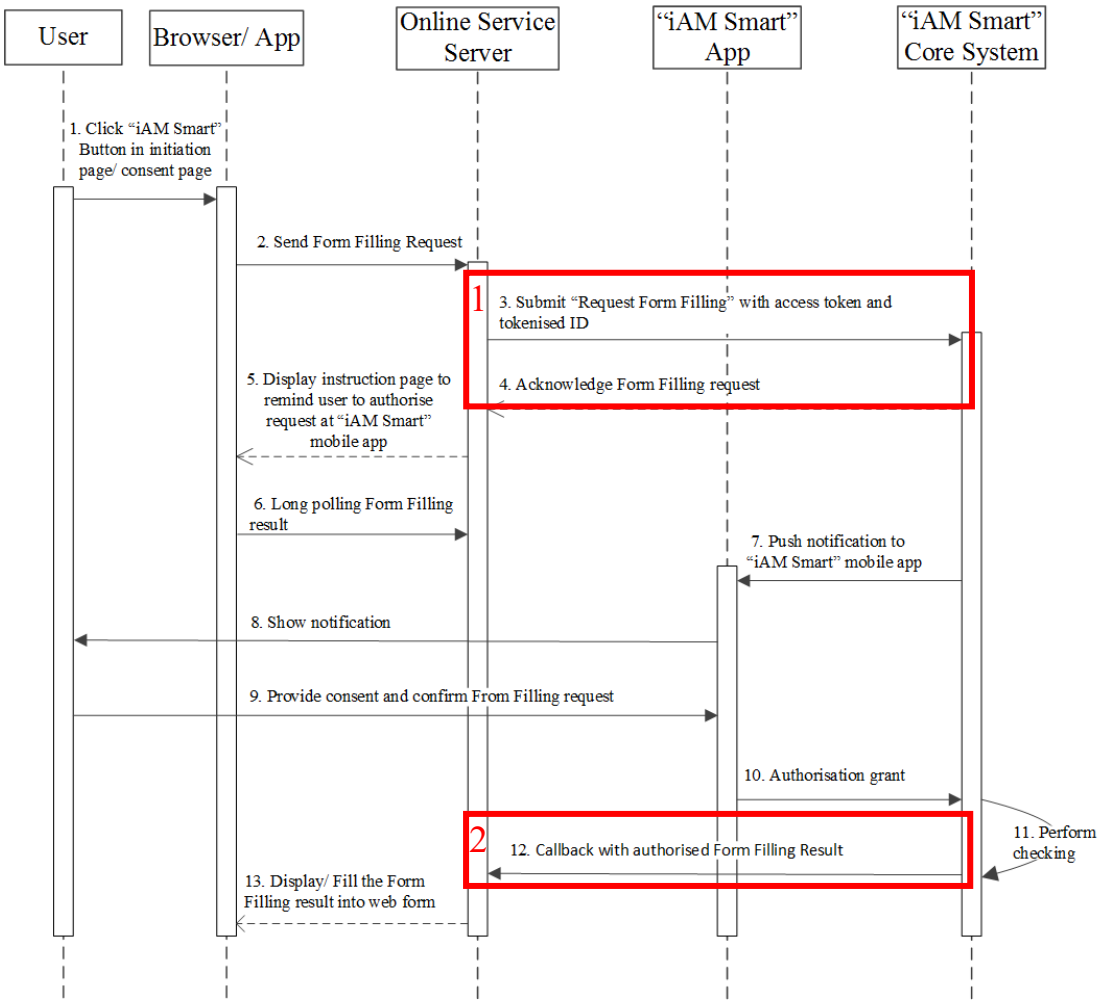

Figure-12 Form Filling (Online Service Website/App in Different Device)

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Form Filling	4.2.5.1
2	Callback to Receive Form Filling Information	4.2.5.3

4.2.4.2 Form Filling (Online Service Website/App in same Device)

The sequence diagram below shows how an authenticated user performs form filling when Online Service website and the "iAM Smart" Mobile App are running in the same device.

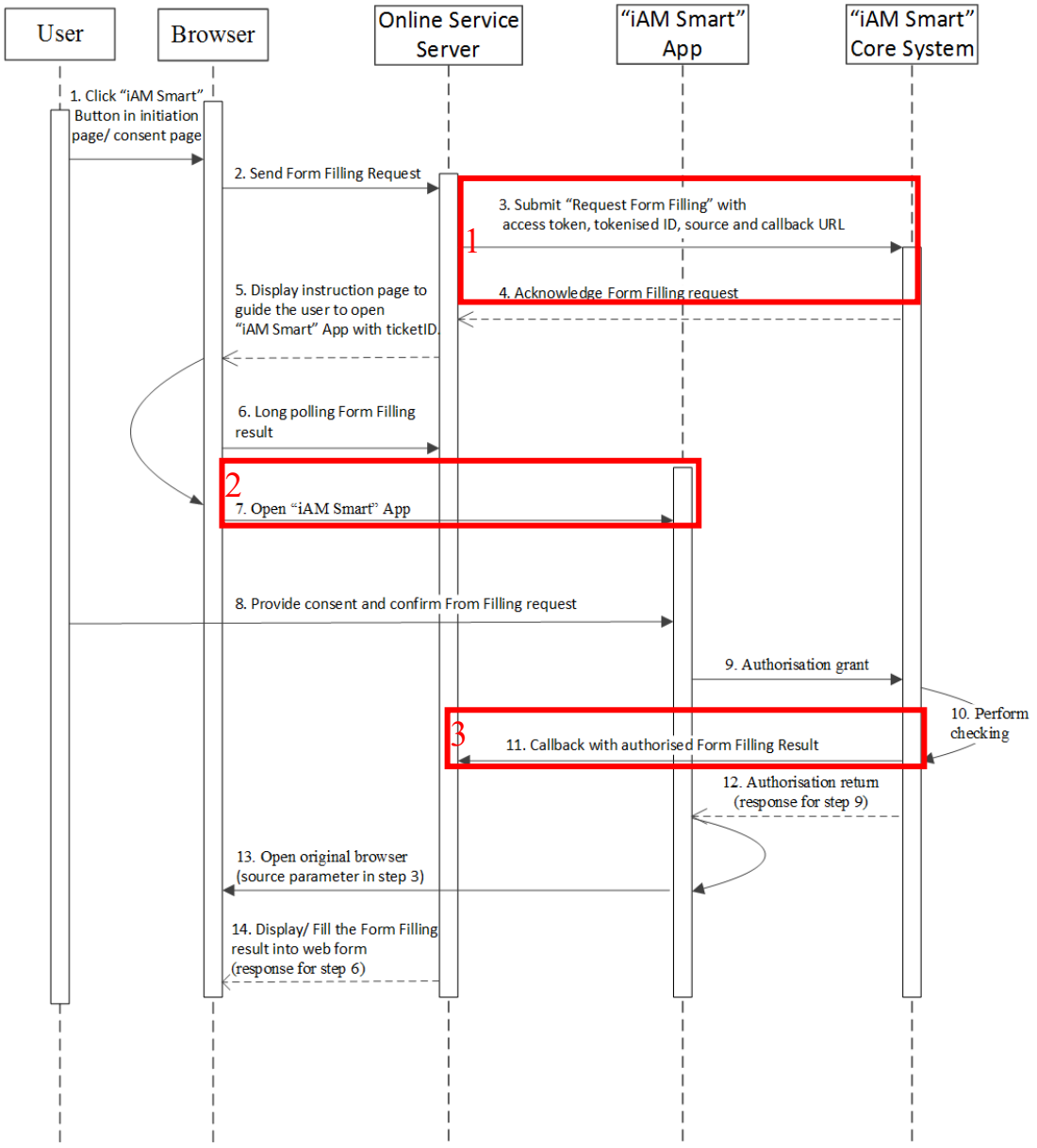

Figure-13 Form Filling (Online Service Website/App in Same Device)

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Form Filling (v2)	4.2.5.1

2	Open “iAM Smart” Mobile App for Form Filling	4.2.5.2
3	Callback to Receive Form Filling Information	4.2.5.3

4.2.4.3 Form Filling (Online Service App in same Device)

The sequence diagram below shows how an authenticated user performs form filling when Online Service App and the “iAM Smart” Mobile App are running in the same device.

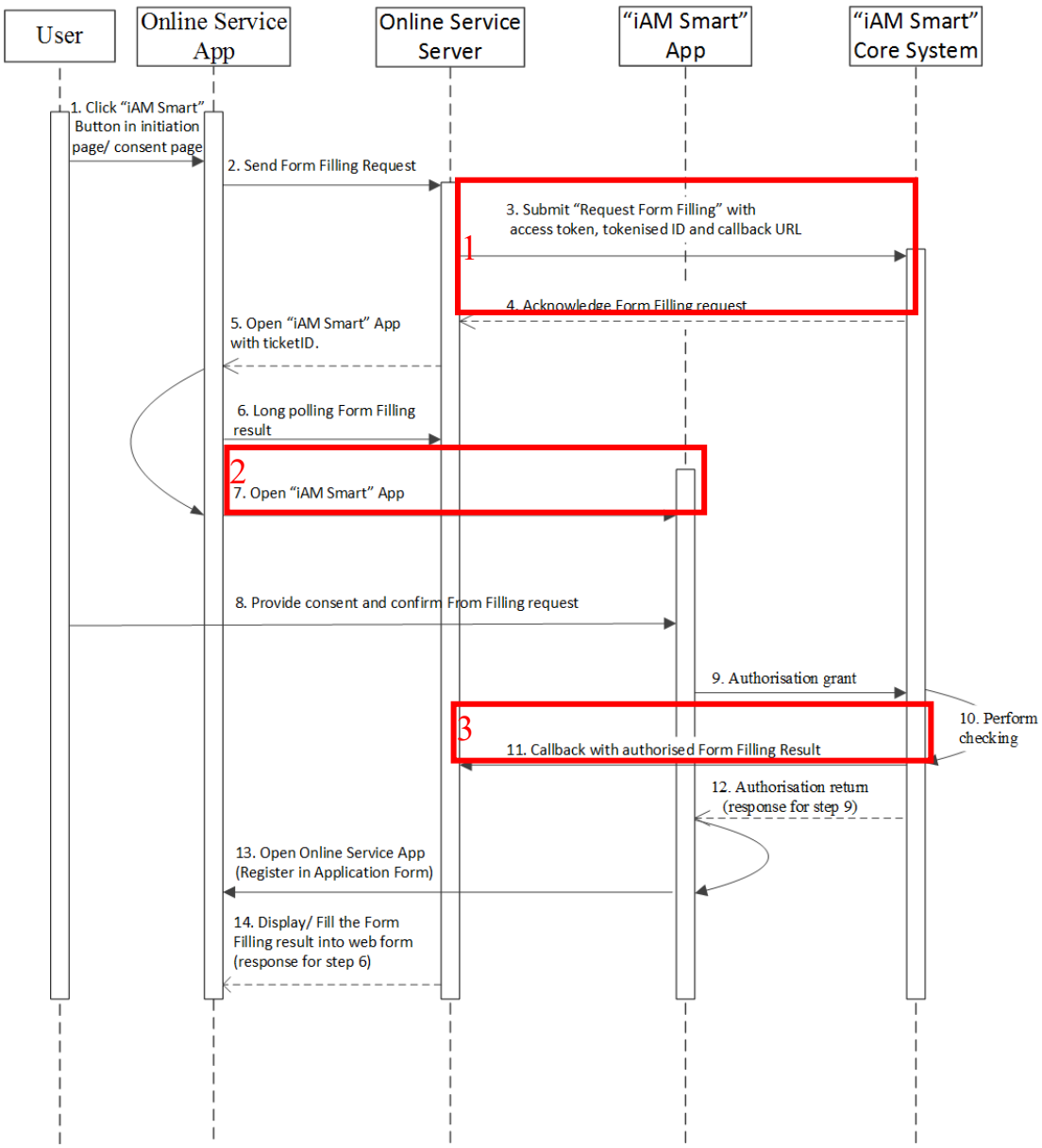

Figure-14 Form Filling (Online Service App in Same Device)

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Form Filling (v2)	4.2.5.1
	Request Form Filling (appv2)	4.2.5.4
2	Open "iAM Smart" Mobile App for Form Filling	4.2.5.2
3	Callback to Receive Form Filling Information	4.2.5.3

4.2.5 API Implementation Details

4.2.5.1 Request Form Filling (v2)

- **API Description**

Name	Description
Service Full Name	Request Form Filling
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v2/account/formFilling/initiateRequest`
Request Type	POST
Service Version	V2.0.0
Description of Service	Online Service can use this API to request for form filling
Remark	If the following eMEFields and profileFields are not provided or are empty array, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned. If same data field exists in both eMEFields and profileFields, it will be shown as profile field. Please refer to "Reference System Flow and User Interface Specifications for "iAM Smart" Adoption" for the user interface requirements.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for Online Service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.

accessToken	String	Required	accessToken value
openID	String	Required	Tokenised ID value
source	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.
redirectURI	String	Required	Callback URI.
state	String	Optional	If state parameter is presented in the request message, the same state value will be returned to Online Service during callback. It is used to prevent the CSRF attack. The value of state is defined by Online Service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
formName	String	Optional	The name of the form should be encoded in Unicode. Maximum Length: 255
formNum	String	Optional	The number of the form should be encoded in Unicode. Maximum Length: 20
formDesc	String	Optional	The description of the form should be encoded in Unicode. Maximum Length: 255
callbackContentType	String	Optional	Specify the content type of the callback request that an Online Service expects to receive in Section 4.2.5.3. The callbackContentType supports "application/json" and "multipart/form-data" values. The default value is "application/json". If the Online Service endpoint has a request size limit (Section 4.2.5.3 callback), Online Service may consider using "multipart/form-data" type.
eMEFields	Array	Required	Specify the "e-ME" fields to be requested.

<table>
<tr><td></td><td></td><td>(Conditional)</td><td>If eMEFields and the following profileFields are not provided or are empty arrays, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned.
The available eMEFields are as follows:<table><tr><th>eMEFields</th><th>Description</th></tr><tr><td>idNo</td><td>ID number</td></tr><tr><td>prefix</td><td>prefix</td></tr><tr><td>enName</td><td>English name</td></tr><tr><td>chName</td><td>Chinese name</td></tr><tr><td>birthDate</td><td>Date of birth</td></tr><tr><td>gender</td><td>gender</td></tr><tr><td>maritalStatus</td><td>marital status</td></tr><tr><td>homeTelNumber</td><td>home telephone number</td></tr><tr><td>officeTelNumber</td><td>office telephone number</td></tr><tr><td>mobileNumber</td><td>mobile number</td></tr><tr><td>emailAddress</td><td>email address</td></tr><tr><td>residentialAddress</td><td>residential address</td></tr><tr><td>postalAddress</td><td>postal address</td></tr><tr><td>educationLevel</td><td>education level</td></tr><tr><td>addressDocInfo</td><td>provider name, retrieval date, owner name and address information related to an e-bill</td></tr><tr><td>addressDocFile</td><td>e-bill from an address data provider</td></tr></table></td></tr>
<tr><td>profileFields</td><td>Array</td><td>Required (Conditional)</td><td>Specify the profile fields to be requested. If profileFields and the above eMEFields are not provided or are empty arrays, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned.</td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td>The available profileFields are as follows:<table><tr><th>profileFields</th><th>Description</th></tr><tr><td>idNo</td><td>ID number</td></tr><tr><td>enName</td><td>English name</td></tr><tr><td>chName</td><td>Chinese name</td></tr><tr><td>birthDate</td><td>Date of birth</td></tr><tr><td>gender</td><td>gender</td></tr></table></td></tr>
</table>

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v2/account/formFilling/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
 "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
 "source": "Android_Chrome",
 "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
 "state": "eddd527b6",
 "formName": "Example Account Registration Form",
 "formNum": "APP0001",
 "formDesc": "Example form description",
 "callbackContentType": "multipart/form-data",
 "profileFields": ["idNo", "enName", "gender", "chName", "birthDate"],
 "eMEFields": ["mobileNumber", " emailAddress", "addressDocInfo"]
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If user completes authentication by scanning QR code (i.e. different device), then "true" will be returned to Online Service.
`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger “iAM Smart” Mobile App when authByQR is `false`

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter { redirectURI }"
}
```

4.2.5.2 Open "iAM Smart" Mobile App for Form Filling

- **Using URL Scheme**

Name	Description
Service Full Name	Open "iAM Smart" Mobile App for Form Filling
URI (as in RESTFUL API)	For Form Filling appv1: <"iAM Smart" app URL scheme>://form-filling For Form Filling appv2: <"iAM Smart" app URL scheme>://v2_form-filling
Service Version	V1.0.0 and V2.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to form filling action in "iAM Smart" Mobile App. The URL schemes supported on iOS and Android versions of "iAM Smart" Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description

`ticketID`	String	Required	`ticketID` is a unique identifier provided by "iAM Smart" System, online service retrieve `ticketID` while requesting the digital signing function. It is ASCII string with length less than or equal 36 chars.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://v2_form-filling
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

4.2.5.3 Callback to Receive Form Filling Information

4.2.5.3.1 Callback with application/json to Receive Form Filling Information

If the optional parameter `callbackContentType` of the "Request Form Filling" API in section *4.2.5.1* is empty or if its value is not set as "multipart/form-data", this callback content type will be in "application/json".

- **API Description**

Name	Description
Service Full Name	Callback with "application/json" content type to Receive Form Filling Information
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	V1.0.0
Description of Service	This callback allow "iAM Smart" System to pass Form Filling information with "application/json" content type to Online Service upon consent.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for Online Service to differentiate different request. Online Service can use the `businessID` to relate the callback message with the original request.

state	String	Optional	If the state parameter has been present in the request message, the exact value of state will be returned. It is used to prevent the CSRF attack. The value of state is defined by Online Service and it should be a secure random value.
idNo	JSON Object	Optional	HKIC number
prefix	String	Optional	prefix
enName	JSON Object	Optional	English name
chName	JSON Object	Optional	Chinese name
birthDate	String	Optional	Date of birth
gender	String	Optional	gender
maritalStatus	String	Optional	marital status
homeTelNumber	JSON Object	Optional	home telephone number
officeTelNumber	JSON Object	Optional	office telephone number
mobileNumber	JSON Object	Optional	mobile number
emailAddress	String	Optional	email address
residentialAddress	JSON Object	Optional	residential address If the data is retrieved from CDEG, “addressProvider” field will also be included. The “providerCode” and “lastRetrievalDate” fields indicate where and when the data was updated.
postalAddress	JSON Object	Optional	postal address If the data is retrieved from CDEG, “addressProvider” field will also be included. The “providerCode” and “lastRetrievalDate” fields indicate where and when the data was updated.
educationLevel	String	Optional	education level
chNameVerified	String	Optional	ImmD characters code point will be returned to Online Service if Chinese name verified.
addressDocInfo	JSON Object	Optional	the data related to the obtained e-bill, including the provider name,

			retrieval date, owner name, service address and postal address. The sub-field `lastRetrievalDate` refers to the date and time at which "iAM Smart" obtained the e-bill. The schema of sub-fields is given in **Appendix A**.
`addressDocFile`	JSON Object	Optional	the e-bill obtained from an address data provider. Sub-fields include the PDF e-bill file (`docFile`), the SHA256 hash (`docHash`) and the bill date of the e-bill (`billDate`). The schema of sub-fields is given in **Appendix A**.

- **Example Callback**

The maximum length of idNo, prefix, enName, chName and chNameVerified can be found in Appendix A of this specification. Most of the fields are defined in DPO common schema. More information about common schema can be found at https://www.digitalpolicy.gov.hk/en/our_work/data_governance/policies_standards/interoperability_framework/common_schemas/.

```
// The descriptions of txID, code, and message are in Section 2.4.2
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>
// Callback Body
{
   "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
   "code": "D00000",
   "message": "SUCCESS",
   "content": {
      "businessID": "2YotnFZFEjr1zCsicMWpAA",
      "idNo": {
         "Identification": "A123456",
         "CheckDigit": "A"
      },
      "prefix": "Mr",
      "enName": {
         "UnstructuredName": "SAN, Chi Nan"
      },
      "chName": {
         "ChineseName": "申智能"
      },
```

```
"chNameVerified": "申智能",
// May be 19960000 or 19960100
"birthDate": "19960128",
"gender": "M",
"maritalStatus": "S",
"homeTelNumber": {
   "CountryCode": "852",
   "SubscriberNumber": "98765432"
},
"officeTelNumber": {
   "CountryCode": "1",
   "SubscriberNumber": "123456"
},
"mobileNumber": {
   "CountryCode": "1",
   "SubscriberNumber": "98765432"
},
"emailAddress": "scn@digitalpolicy.gov.hk",

 // Three different tags exist in residentialAddress
//  and the returned value  can be either one of the following
//  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress

// Both ChiPremisesAddress and EngPremisesAddress contain either
// 1): Standard/Village Address
// 2): Lot Address

//   Example of ChiPremisesAddress with Standard/Village Address
"residentialAddress": {
   "ChiPremisesAddress": {
      "Region": "香港",
      "ChiDistrict": {
         "DcDistrict": "WC",
         "Sub-district": "灣仔"
      },
      "BuildingName": "灣仔政府大樓",
      "ChiEstate": {
         "EstateName": "華富",
         "ChiPhase": {
            "PhaseName": "華清"
         }
      },
      "ChiStreet": {
         "StreetName": "港灣道",
         "BuildingNoFrom": "12"
      },
      "ChiBlock": {
         "BlockDescriptor": "座",
         "BlockNo": "東"
```

```
            },
            "Chi3dAddress": {
                "ChiFloor": {
                    "FloorNum": "15"
                },
                "ChiUnit": {
                    "UnitDescriptor": "室",
                    "UnitNo": "A1"
                }
            }
        },

        //  If the data is retrieved from CDEG, "addressProvider" field
        //  will also be included.
        "addressProvider": {
            "providerCode": "wsd",
            "lastRetrievalDate": 1560853718006
        }
    },

    //  Example of EngPremisesAddress with Lot address
    "residentialAddress": {
        "EngPremisesAddress": {
            "EngLot": {
                "EngStructuredLot": {
                    "DdType": "DD",
                    "DdNo": "110",
                    "LotType": "LOT",
                    "LotNo": "157",
                    "LotSection1": "A",
                    "LotSubsection1": "1",
                    "LotSection2": "B",
                    "LotSubsection2": "1",
                    "LotSection3": "AA",
                    "LotSubsection3": "1",
                    "LotExtendPortionCode": "3"
                }
            }
        }
    },

    //  Example of FreeFormatAddress
    "residentialAddress": {
        "FreeFormatAddress": {
            "LanguageCode": "en",
            "AddressLine1": "Unit 2000, 200/F",
            "AddressLine2": "5033 Yitian Road, Futian CBD",
            "AddressLine3": "Futian district, Shenzhen"
        }
    },
```

```
 // Four different tags exist in postalAddress
//  The returned value of postalAddress can be either one of the
//  following
//  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress,
//  PostBoxAddress

//  Apart from PostBoxAddress, the other three tags are equivalent to
//  ResidentialAddress

//  PostBoxAddress contains either
   // 1): EngPostBox
   // 2): ChiPostBox

  // Example of postalAddress with EngPostBox
"postalAddress": {
   "PostBoxAddress": {
        "EngPostBox": {
           "PoBoxNo": 24700,
           "PostOffice": "ABERDEEN POST OFFICE",
           "PostOfficeRegion": "HONG KONG"
       }
   }
},
"educationLevel": "T",
// all the sub-fields come from the address data provider
"addressDocInfo": {
   // name of the address data provider, possible values are:
   // The Hong Kong and China Gas Company Limited
   // CLP Power Hong Kong Ltd.
   // HK Electric
   // Water Supplies Department
   "enProviderName": "HK Electric",
   "tcProviderName": "港燈",
   "scProviderName": "港灯",
   // unstructured name of the e-bill owner
   "enUserName": "SAN, Chi Nan",
   "tcUserName": "申智能",
   // service address of the e-bill, at most 4 lines
   "enServiceAddress": {
      "addressLine1": "19/F, 213 Queen's Road East",
      "addressLine2": "Wanchai, Hong Kong"
   },
   "tcServiceAddress": {
      "addressLine1": "香港灣仔皇后大道東231號",
      "addressLine2": "胡忠大廈19樓"
   },
   // postal address of the e-bill, at most 4 lines
   "enPostalAddress": {
      "addressLine1": "19/F, 213 Queen's Road East",
```

```
            "addressLine2": "Wanchai, Hong Kong"
         },
         "tcPostalAddress": {
            "addressLine1": "香港灣仔皇后大道東231號",
            "addressLine2": "胡忠大廈19樓"
         },
         // last retrieval date of the e-bill
         // expressed in the number of milliseconds since 1970/1/1 GMT
         "lastRetrievalDate": 1560853718006
      }
      "addressDocFile": {
         // Base64 encoded string of the PDF e-bill file
         "docFile": "JVBERi0xLjUKJcK1wrXCtcK1CjEgMCBv…iag==",
         // SHA256 hash of PDF e-bill file
         "docHash": "69e3…c4e5",
         // bill date of the e-bill
         // expressed in the number of milliseconds since 1970/1/1 GMT
         "billDate": 1560849218006
      }
   }
}
```

4.2.5.3.2 Callback with multipart/form-data to Receive Form Filling Information

If the optional parameter `callbackContentType` of the "Request Form Filling" API in section *4.2.5.1* is set as "multipart/form-data", this callback content type will be "multipart/form-data". The callback body consists of two parts. One part is the "docFile" which contains an e-bill with a binary pdf format. Another part is the "profile" data which includes the eMEFields and profileFields that returned in JSON format. Both "docFile" and "profile" parts are encrypted with the same CEK.

- **API Description**

Name	Description
Service Full Name	Callback with multipart/form-data to Receive Form Filling Information
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	V1.0.0
Description of Service	This callback allows "iAM Smart" System to pass Form Filling information in the multipart/form-data content type to Online Service upon consent.

- **Callback Parameters**

Parameter	Type	Presence	Description
businessID	String	Required	businessID is a unique identifier for Online Service to differentiate different request. Online Service can use the businessID to relate the callback message with the original request.
state	String	Optional	If the state parameter has been present in the request message, the exact value of state will be returned. It is used to prevent the CSRF attack. The value of state is defined by Online Service and it should be a secure random value.
idNo	JSON Object	Optional	HKIC number
prefix	String	Optional	prefix
enName	JSON Object	Optional	English name
chName	JSON Object	Optional	Chinese name
birthDate	String	Optional	Date of birth
gender	String	Optional	gender
maritalStatus	String	Optional	marital status
homeTelNumber	JSON Object	Optional	home telephone number
officeTelNumber	JSON Object	Optional	office telephone number
mobileNumber	JSON Object	Optional	mobile number
emailAddress	String	Optional	email address
residentialAddress	JSON Object	Optional	residential address If the data is retrieved from CDEG, “addressProvider” field will also be included. The “providerCode” and “lastRetrievalDate” fields indicate where and when the data was updated.
postalAddress	JSON Object	Optional	postal address If the data is retrieved from CDEG, “addressProvider” field will also be included. The “providerCode” and “lastRetrievalDate” fields indicate

			where and when the data was updated.
`educationLevel`	String	Optional	education level
`chNameVerified`	String	Optional	ImmD characters code point will be returned to Online Service if Chinese name verified.
`addressDocInfo`	JSON Object	Optional	the data related to the obtained e-bill, including the provider name, retrieval date, owner name, service address and postal address. The sub-field `lastRetrievalDate` refers to the date and time at which "iAM Smart" obtained the e-bill. The schema of sub-fields is given in **Appendix A**.
`addressDocFile`	JSON Object	Optional	the e-bill metadata obtained from an address data provider. Sub-fields include the SHA256 hash (`docHash`) and the bill date of the e-bill (`billDate`). The schema of sub-fields is given in **Appendix A**. These metadata correspond to the docFile that's returned in another part of the multipart/form-data callback.
`docFile`	Binary	Optional	the PDF e-bill binary file obtained from an address data provider. It's returned in an individual part of the multipart/form-data callback.

- **Example Callback**

The maximum length of idNo, prefix, enName, chName and chNameVerified can be found in Appendix A of this specification. Most of the fields are defined in DPO common schema. More information about common schema can be found at https://www.digitalpolicy.gov.hk/en/our_work/data_governance/policies_standards/interoperability_framework/common_schemas/.

```
// The descriptions of txID, code, and message are in Section 2.4.2
```

```
// Line breaks are for legibility only.
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>

// Callback Header
// The $<boundary> looks like --ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9
// It is automatically generated and is different in every callback
Content-Type: multipart/form-data;boundary=$<boundary>

// Callback Body
$<boundary>
Content-Disposition: form-data; name="docFile"; filename="docfile.pdf"
Content-Type: application/pdf
Content-Length: 62892
$<encrypted PDF e-bill binary file>

$<boundary>
Content-Disposition: form-data; name="profile"
Content-Type: application/json;charset=UTF-8
{
   "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
   "code": "D00000",
   "message": "SUCCESS",
   "content": {
      "businessID": "2YotnFZFEjr1zCsicMWpAA",
      "idNo": {
         "Identification": "A123456",
         "CheckDigit": "A"
      },
      "prefix": "Mr",
      "enName": {
         "UnstructuredName": "SAN, Chi Nan"
      },
      "chName": {
         "ChineseName": "申智能"
      },
      "chNameVerified": "申智能",
      // May be 19960000 or 19960100
      "birthDate": "19960128",
      "gender": "M",
      "maritalStatus": "S",
      "homeTelNumber": {
         "CountryCode": "852",
         "SubscriberNumber": "98765432"
      },
      "officeTelNumber": {
         "CountryCode": "1",
         "SubscriberNumber": "123456"
      },
```

```
        "mobileNumber": {
            "CountryCode": "1",
            "SubscriberNumber": "98765432"
        },
        "emailAddress": "scn@digitalpolicy.gov.hk",

         // Three different tags exist in residentialAddress
        //  and the returned value  can be either one of the following
        //  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress

        // Both ChiPremisesAddress and EngPremisesAddress contain either
        // 1): Standard/Village Address
        // 2): Lot Address

       //   Example of ChiPremisesAddress with Standard/Village Address
        "residentialAddress": {
            "ChiPremisesAddress": {
                "Region": "香港",
                "ChiDistrict": {
                    "DcDistrict": "WC",
                    "Sub-district": "灣仔"
                },
                "BuildingName": "灣仔政府大樓",
                "ChiEstate": {
                    "EstateName": "華富",
                    "ChiPhase": {
                        "PhaseName": "華清"
                    }
                },
                "ChiStreet": {
                    "StreetName": "港灣道",
                    "BuildingNoFrom": "12"
                },
                "ChiBlock": {
                    "BlockDescriptor": "座",
                    "BlockNo": "東"
                },
                "Chi3dAddress": {
                    "ChiFloor": {
                        "FloorNum": "15"
                    },
                    "ChiUnit": {
                        "UnitDescriptor": "室",
                        "UnitNo": "A1"
                    }
                }
            } ,

            //   If the data is retrieved from CDEG, "addressProvider" field
```

```
            //   will also be included.
            "addressProvider": {
                "providerCode": "wsd",
                "lastRetrievalDate": 1560853718006
            }
        },

        //   Example of EngPremisesAddress with Lot address
        "residentialAddress": {
            "EngPremisesAddress": {
                "EngLot": {
                    "EngStructuredLot": {
                        "DdType": "DD",
                        "DdNo": "110",
                        "LotType": "LOT",
                        "LotNo": "157",
                        "LotSection1": "A",
                        "LotSubsection1": "1",
                        "LotSection2": "B",
                        "LotSubsection2": "1",
                        "LotSection3": "AA",
                        "LotSubsection3": "1",
                        "LotExtendPortionCode": "3"
                    }
                }
            }
        },

        //   Example of FreeFormatAddress
        "residentialAddress": {
            "FreeFormatAddress": {
                "LanguageCode": "en",
                "AddressLine1": "Unit 2000, 200/F",
                "AddressLine2": "5033 Yitian Road, Futian CBD",
                "AddressLine3": "Futian district, Shenzhen"
            }
        },

         // Four different tags exist in postalAddress
        //  The returned value of postalAddress can be either one of the
        //  following
        //  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress,
        //  PostBoxAddress

        //  Apart from PostBoxAddress, the other three tags are equivalent to
        //  ResidentialAddress

        //  PostBoxAddress contains either
            // 1): EngPostBox
            // 2): ChiPostBox
```

```
  // Example of postalAddress with EngPostBox
"postalAddress": {
   "PostBoxAddress": {
        "EngPostBox": {
           "PoBoxNo": 24700,
           "PostOffice": "ABERDEEN POST OFFICE",
           "PostOfficeRegion": "HONG KONG"
       }
   }
},
"educationLevel": "T",
// all the sub-fields come from the address data provider
"addressDocInfo": {
   // name of the address data provider, possible values are:
   // The Hong Kong and China Gas Company Limited
   // CLP Power Hong Kong Ltd.
   // HK Electric
   // Water Supplies Department
   "enProviderName": "HK Electric",
   "tcProviderName": "港燈",
   "scProviderName": "港灯",
   // unstructured name of the e-bill owner
   "enUserName": "SAN, Chi Nan",
   "tcUserName": "申智能",
   // service address of the e-bill, at most 4 lines
   "enServiceAddress": {
      "addressLine1": "19/F, 213 Queen's Road East",
      "addressLine2": "Wanchai, Hong Kong"
   },
   "tcServiceAddress": {
      "addressLine1": "香港灣仔皇后大道東231號",
      "addressLine2": "胡忠大廈19樓"
   },
   // postal address of the e-bill, at most 4 lines
   "enPostalAddress": {
      "addressLine1": "19/F, 213 Queen's Road East",
      "addressLine2": "Wanchai, Hong Kong"
   },
   "tcPostalAddress": {
      "addressLine1": "香港灣仔皇后大道東231號",
      "addressLine2": "胡忠大廈19樓"
   },
   // last retrieval date of the e-bill
   // expressed in the number of milliseconds since 1970/1/1 GMT
   "lastRetrievalDate": 1560853718006
}
"addressDocFile": {
   // SHA256 hash of PDF e-bill file
```

```
            "docHash": "69e3…c4e5",
            // bill date of the e-bill
            // expressed in the number of milliseconds since 1970/1/1 GMT
            "billDate": 1560849218006
        }
    }
}
$<boundary>
```

4.2.5.4 Request Form Filling (appv2)

- **API Description**

Name	Description
Service Full Name	Request Form Filling
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/appv2/account/formFilling/initiateRequest`
Request Type	POST
Service Version	V2.0.0
Description of Service	Online service can use this API to request for form filling
Remark	If the following eMEFields and profileFields are not provided or are empty array, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned. If same data field exists in both eMEFields and profileFields, it will be shown as profile field. Please refer to “Reference System Flow and User Interface Specifications for “iAM Smart” Adoption” for the user interface requirements.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value
`source`	String	Required	App_Link (iOS) or App_Package (Android)

clientRedirectURI	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (“,”) are not accepted.
packageName	String	Required (Android only)	Package name of the Online Service App for callback use.
activityClass	String	Required (Android only)	Activity class name of the Online Service App for callback use.
activityParams	String	Optional (Android only)	Optional params used during callback.
serverRedirectURI	String	Required	Callback URI.
state	String	Optional	If state parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
formName	String	Optional	The name of the form should be encoded in Unicode. Maximum Length: 255
formNum	String	Optional	The number of the form should be encoded in Unicode. Maximum Length: 20
formDesc	String	Optional	The description of the form should be encoded in Unicode. Maximum Length: 255
callbackContentType	String	Optional	Specify the content type of the callback request that an online service expects to receive in 4.2.5.3. The callbackContentType supports

<table>
<tr><td></td><td></td><td></td><td>“application/json” and “multipart/form-data” values. The default value is “application/json”.
If the online service endpoint has a request size limit (Section 4.2.5.3 callback), online service may consider using “multipart/form-data” type.</td></tr>
<tr><td>eMEFields</td><td>Array</td><td>Required (Conditional)</td><td>Specify the “e-ME” fields to be requested. If eMEFields and the following profileFields are not provided or are empty arrays, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned.
The available eMEFields are as follows:
<table>
<tr><th>eMEFields</th><th>Description</th></tr>
<tr><td>idNo</td><td>ID number</td></tr>
<tr><td>prefix</td><td>prefix</td></tr>
<tr><td>enName</td><td>English name</td></tr>
<tr><td>chName</td><td>Chinese name</td></tr>
<tr><td>birthDate</td><td>Date of birth</td></tr>
<tr><td>gender</td><td>gender</td></tr>
<tr><td>maritalStatus</td><td>marital status</td></tr>
<tr><td>homeTelNumber</td><td>home telephone number</td></tr>
<tr><td>officeTelNumber</td><td>office telephone number</td></tr>
<tr><td>mobileNumber</td><td>mobile number</td></tr>
<tr><td>emailAddress</td><td>email address</td></tr>
<tr><td>residentialAddress</td><td>residential address</td></tr>
<tr><td>postalAddress</td><td>postal address</td></tr>
<tr><td>educationLevel</td><td>education level</td></tr>
<tr><td>addressDocInfo</td><td>provider name, retrieval date, owner name and address information related to an e-bill</td></tr>
<tr><td>addressDocFile</td><td>e-bill from an</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td><table><tr><td></td><td>address data provider</td></tr></table></td></tr>
<tr><td>profileFields</td><td>Array</td><td>Required (Conditional)</td><td>Specify the profile fields to be requested. If profileFields and the above eMEFields are not provided or are empty arrays, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned.
The available profileFields are as follows:<table><tr><th>profileFields</th><th>Description</th></tr><tr><td>idNo</td><td>ID number</td></tr><tr><td>enName</td><td>English name</td></tr><tr><td>chName</td><td>Chinese name</td></tr><tr><td>birthDate</td><td>Date of birth</td></tr><tr><td>gender</td><td>gender</td></tr></table></td></tr>
</table>

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/appv2/account/formFilling/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
 "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
 "source": "App_Package",
 "packageName": "com.onlineservice.myapp",
 "activityClass": "callback_activity",
 "activityParams": "callback_param",
 "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
```

```
 "state": "eddd527b6",
 "formName": "Example Account Registration Form",
 "formNum": "APP0001",
 "formDesc": "Example form description",
 "callbackContentType": "multipart/form-data",
 "profileFields": ["idNo", "enName", "gender", "chName", "birthDate"],
 "eMEFields": ["mobileNumber", " emailAddress", "addressDocInfo"]
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If user completes authentication by scanning QR code (i.e. different device), then "true" will be returned to Online Service. `authByQR` should always be `false` in appv2 API.
`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger “iAM Smart” Mobile App when authByQR is `false`.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": false,
    "ticketID": "bbb8aae57c104cda40c93843ad5e6db8"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
```

```
  "message": "empty parameter { redirectURI }"
}
```

5. FORM FILLING WITHOUT SERVICE LOGIN (AKA ANONYMOUS FORM FILLING)

5.1 Overview

Similar to Form Filling API, online service can request "Anonymous Form Filling" API to obtain "iAM Smart" user's personal data upon user consent. Unlike the accessToken obtained from Authentication API, the accessToken received in this API can only be used once.

The typical use cases cover account link-up, application form system, registration form, remote account onboarding, identity verification and address verification.

5.2 Prerequisite

- "Profile fields" in Anonymous Form Filling API is for user identity verification purpose, while "e-ME" fields in Anonymous Form Filling API is for form filling usage.
- Online service shall refer to "Reference System Flow and User Interface Specifications for "iAM Smart" Adoption" for the detail UI/UX requirements.

5.3 Scope

Online service shall submit the application for using the following scope values to "iAM Smart" Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_formFilling`	Scope for Form Filling(v2)

5.4 Use Cases and Scenarios

5.4.1 Anonymous Form Filling (Online Service Website in Different Device)

The sequence diagram below shows how an anonymous user performs form filling when online service website and the "iAM Smart" Mobile App are running in different devices.

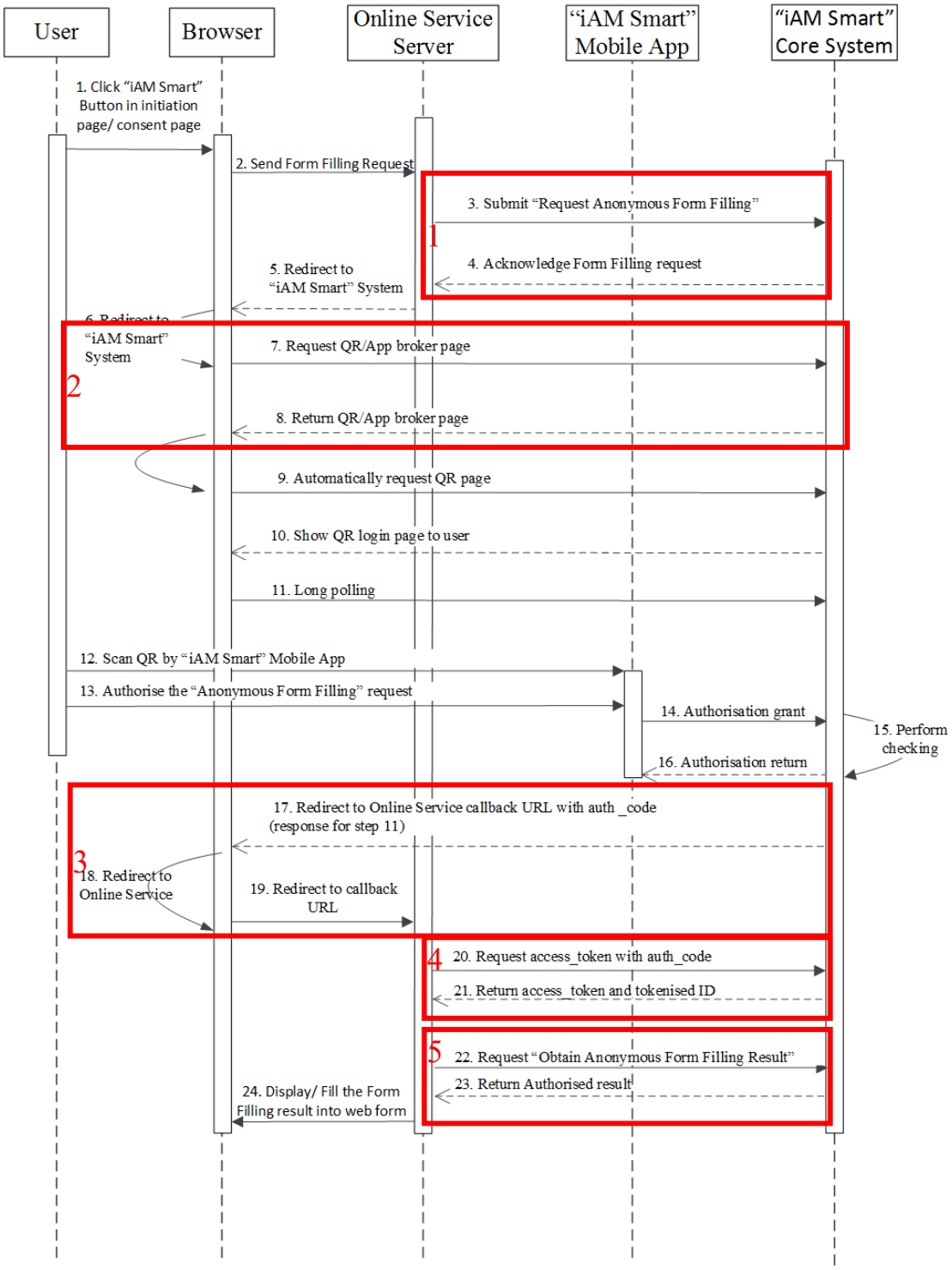

Figure-15 Anonymous Form Filling (Online Service Website in Different Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Form Filling	5.5.1
2	Request QR Page	5.5.2
3	Callback with authCode to Online Service Server	5.5.5
4	Request accessToken & Tokenised ID	5.5.6
5	Obtain Anonymous Form Filling Result	5.5.7

5.4.2 Anonymous Form Filling (Online Service Website in Same Device)

The sequence diagram below shows how an anonymous user performs form filling when online service website and the "iAM Smart" Mobile App are running in the same device.

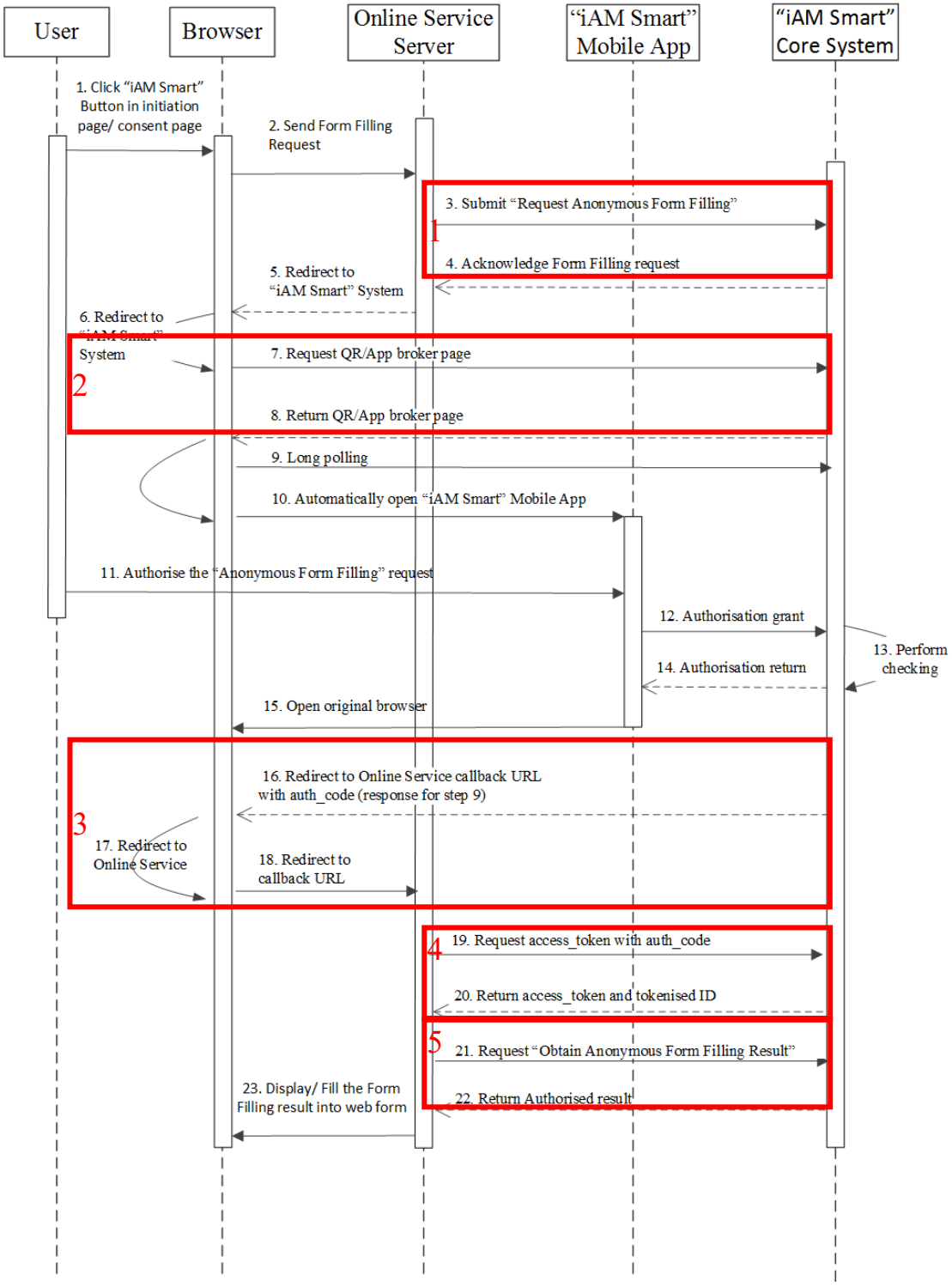

Figure-16 Anonymous Form Filling (Online Service Website in Same Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Form Filling	5.5.1
2	Request QR Page	5.5.2
3	Callback with authCode to Online Service Server	5.5.5
4	Request accessToken & Tokenised ID	5.5.6
5	Obtain Anonymous Form Filling Result	5.5.7

5.4.3 Anonymous Form Filling (Online Service App in Different Device)

The sequence diagram below shows how an anonymous user performs form filling when online service App and the "iAM Smart" Mobile App are running in different devices.

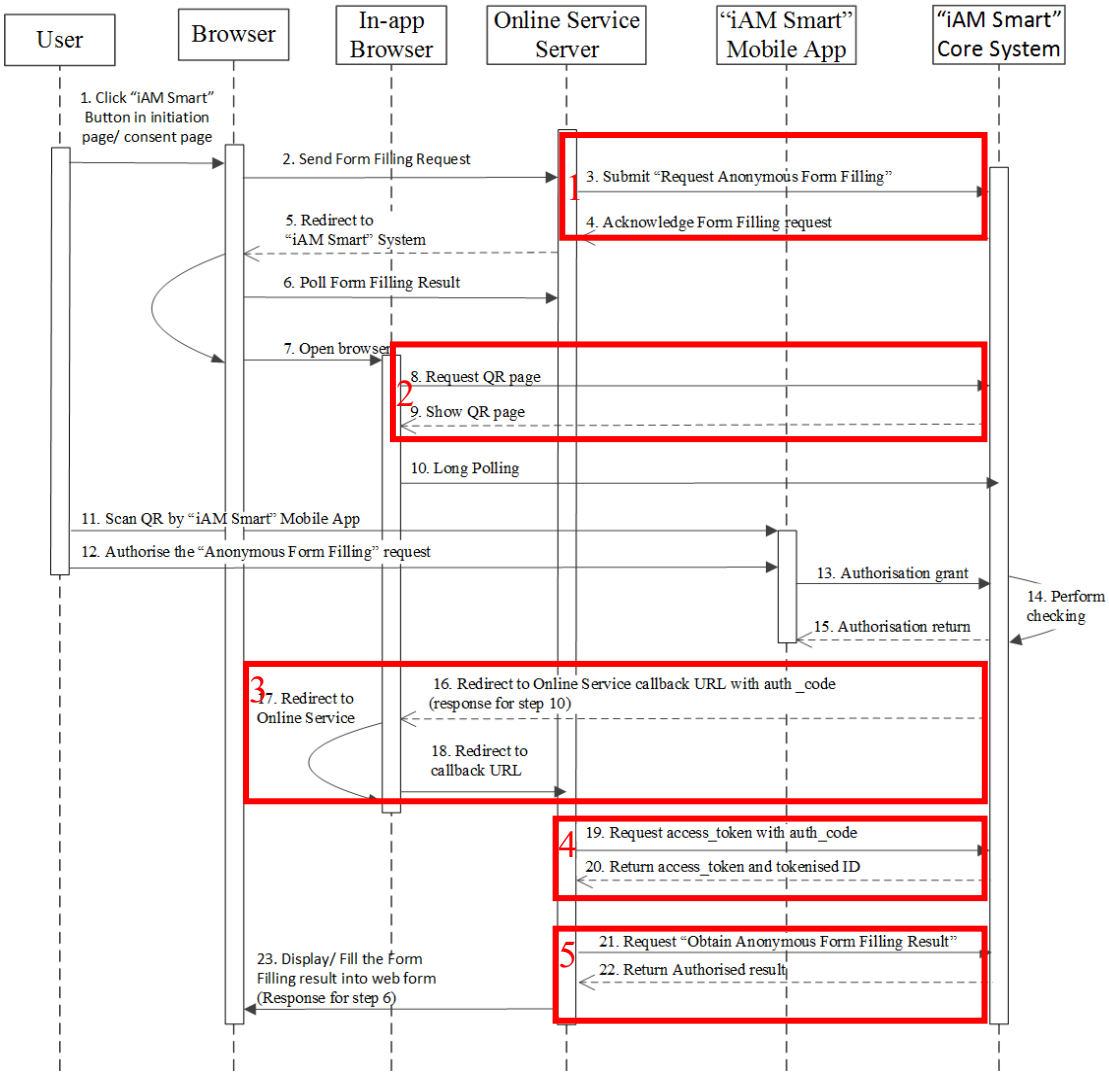

Figure-17 Anonymous Form Filling (Online Service App in Different Device)

APIs interactions between online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Form Filling	5.5.1
2	Request QR Page	5.5.2
3	Callback with authCode to Online Service Server	5.5.5

4	Request accessToken & Tokenised ID	5.5.6
5	Obtain Anonymous Form Filling Result	5.5.7

5.4.4 Anonymous Form Filling (Online Service App in Same Device)

The sequence diagram below shows how an anonymous user performs form filling when online service App and the "iAM Smart" Mobile App are running in the same device.

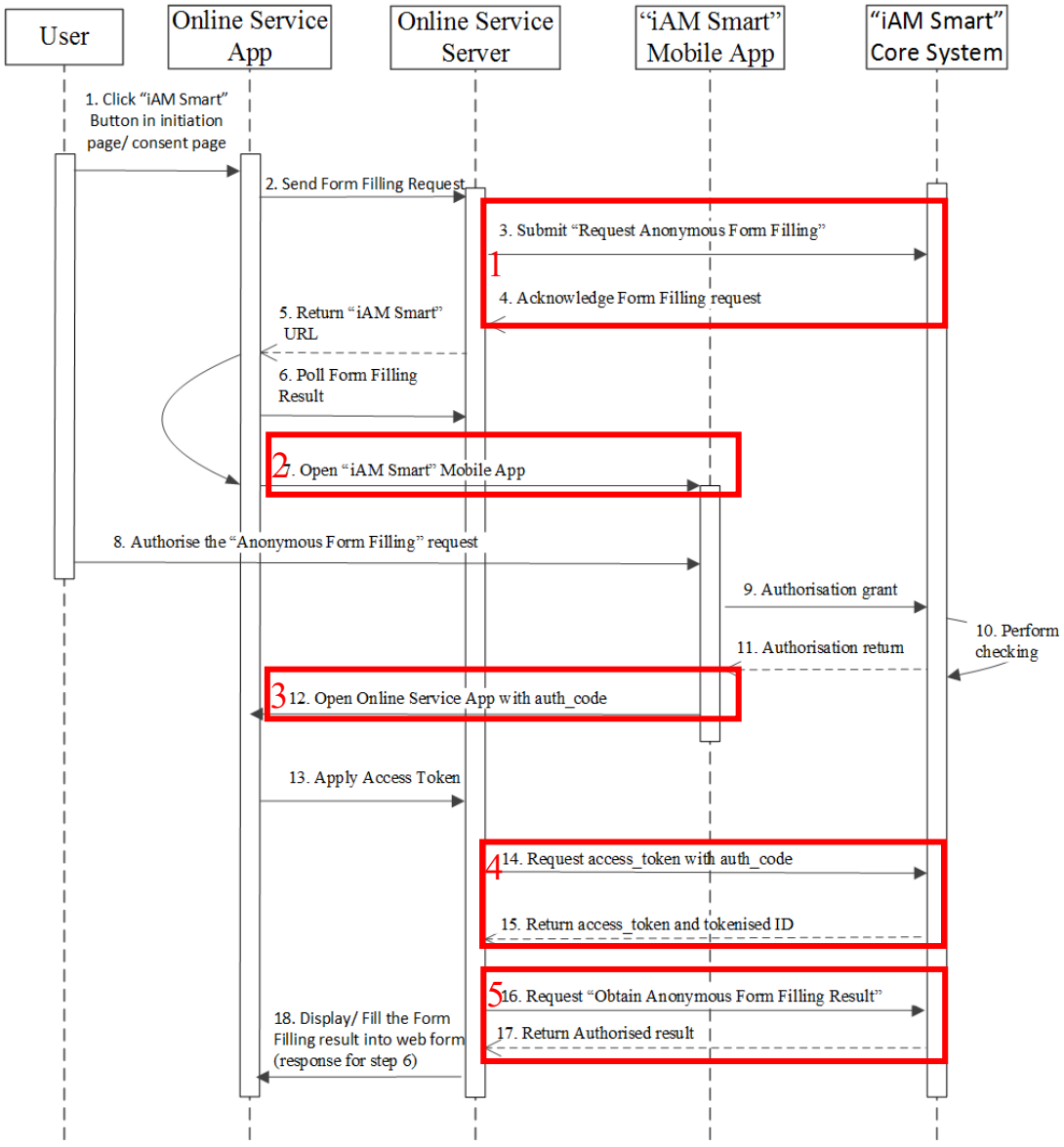

Figure-18 Anonymous Form Filling (Online Service App in Same Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Form Filling	5.5.1
2	Open “iAM Smart” Mobile App for Anonymous Form Filling (appv1)	5.5.3
3	Open “iAM Smart” Mobile App for Anonymous Form Filling (appv2)	5.5.8
4	Callback with authCode to Online Service App	5.5.4
5	Request accessToken & Tokenised ID	5.5.6
6	Obtain Anonymous Form Filling Result	5.5.7

5.5 API Implementation Details

5.5.1 Request Anonymous Form Filling

- **API Description**

Name	Description
Service Full Name	Request Anonymous Form Filling
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v2/anonymous/formFilling/initiateRequest`
Request Type	POST
Service Version	V2.0.0
Description of Service	Online service can use this API to initiate anonymous form filling the request.
Remark	If the following eMEFields and profileFields are not provided or are empty arrays, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned. If the same data field exists in both eMEFields and profileFields, it will be shown as a profile field. Please refer to "Reference System Flow and User Interface Specifications for "iAM Smart" Adoption" for the user interface requirements.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`formName`	String	Optional	The name of the form should be encoded in Unicode. Maximum Length: 255
`formNum`	String	Optional	The number of the form should be encoded in Unicode. Maximum Length: 20
`formDesc`	String	Optional	The description of the form should be encoded in Unicode. Maximum Length: 255

<table>
<tr><td>eMEFields</td><td>Array</td><td>Required
(Conditional)</td><td>It is for user form-filling purposes.
Specify the “e-ME” fields to be requested.
If eMEFields and the following profileFields are not provided or are empty arrays, the HTTP code 200 with error code D20002 (empty parameter {eMEFields，profileFields}) will be returned.
The available eMEFields are as follows:<table><tr><th>eMEFields</th><th>Description</th></tr><tr><td>idNo</td><td>ID number</td></tr><tr><td>prefix</td><td>prefix</td></tr><tr><td>enName</td><td>English name</td></tr><tr><td>chName</td><td>Chinese name</td></tr><tr><td>birthDate</td><td>Date of birth</td></tr><tr><td>gender</td><td>gender</td></tr><tr><td>maritalStatus</td><td>marital status</td></tr><tr><td>homeTelNumber</td><td>home telephone number</td></tr><tr><td>officeTelNumber</td><td>office telephone number</td></tr><tr><td>mobileNumber</td><td>mobile number</td></tr><tr><td>emailAddress</td><td>email address</td></tr><tr><td>residentialAddress</td><td>residential address</td></tr><tr><td>postalAddress</td><td>postal address</td></tr><tr><td>educationLevel</td><td>education level</td></tr><tr><td>addressDocInfo</td><td>provider name, retrieval date, owner name and address information related to an e-bill</td></tr><tr><td>addressDocFile</td><td>e-bill from an address data provider</td></tr></table></td></tr>
<tr><td>profileFields</td><td>Array</td><td>Required
(Conditional)</td><td>It is for user identity verification purposes.
Specify the profile fields to be requested. If profileFields and the above eMEFields are</td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td>not provided or are empty arrays, the HTTP code 200 with error code D20002 (empty parameter {eMEFields , profileFields}) will be returned.
The available profileFields are as follows:<table><tr><th>profileFields</th><th>Description</th></tr><tr><td>idNo</td><td>ID number</td></tr><tr><td>enName</td><td>English name</td></tr><tr><td>chName</td><td>Chinese name</td></tr><tr><td>birthDate</td><td>Date of birth</td></tr><tr><td>gender</td><td>gender</td></tr></table></td></tr>
</table>

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v2/anonymous/formFilling/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "formName": "Example Account Registration Form",
 "formNum": "APP0001",
 "formDesc": "Example Form Description",
 "profileFields": ["idNo", "enName", "gender", "chName", "birthDate"],
 "eMEFields": ["mobileNumber", " emailAddress", "addressDocInfo"]
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier that will be used to identify this request in

			the following steps of the Anonymous Form Filling workflow. The ticketID will be expired 12 minutes after issuance.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "ticketID": "561b6f1ce5a043ba8aae33fb5c50386d"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in  Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {businessID}"
}
```

5.5.2 Request QR Page

- **API Description**

Name	Description
Service Full Name	Request QR Page
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/auth/getQR`
Request Type	GET
Service Version	1.0.0
Description of Service	Online service calls this API to get the QR/App broker page or QR page. After the user authorises login or anonymous request on the “iAM Smart” Mobile App, the page will be redirected to the `redirectURI` with `authCode` and `state` parameters. If the user denies it, only the `state` parameter will be redirected.

- **Request Parameters**

Parameter	Type	Presence	Description
`clientID`	String	Required	Online service client identifier. The clientID will be assigned to online service at the initial registration.
`responseType`	String	Required	The value MUST be set to `code`.
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.
`redirectURI`	String	Required	Callback redirectURI. The value should be URL encoded and registered in the self-service portal.
`scope`	String	Required	The value should be URL encoded. Specify the scope of "iAM Smart" functions: - `eidapi_formFilling` The value of the scope parameter is expressed as a list of space-delimited, case-sensitive strings.
`ticketID`	String	Required	Required in anonymous form filling workflows for getting QR page.
`lang`	String	Optional	Language to display: en-US, zh-HK, or zh-CN. If this parameter is not specified, zh-HK will be shown.
`state`	String	Optional	If the `state` parameter is presented in the request message, the same state value will be returned to online service during the callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.
`brokerPage`	Boolean	Optional	If `brokerPage` is set to `true`, Universal Link (iOS) / App Link (Android) will be leveraged to open "iAM Smart" Mobile App. This feature is useful for Online Services that support mobile web versions while triggering

			the “iAM Smart” Mobile App or showing a QR page automatically. (i.e. Show QR page without detecting whether “iAM Smart” Mobile App is installed). The default value is `false`.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
GET
https://<iAM_Smart_domain>/api/v1/auth/getQR
?clientID=Online Service1
&responseType=code
&source=Android_Chrome
&redirectURI=https%3A%2F%2Frp_domain%2Frp_context%2Fcall_back_endpoint
&scope=eidapi_formFilling
&lang=en-US
&state=eb9b7b8eddd5
```

- **Response Parameters**

N/A

5.5.3 Open “iAM Smart” Mobile App for Anonymous Form Filling (appv1)

- **Using URL Scheme**

Name	Description
Service Full Name	Open “iAM Smart” Mobile App for Anonymous Form Filling
URI (as in RESTFUL API)	`<"iAM Smart" app URL scheme>://anon_form-filling`
Service Version	V1.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific Form Filling in “iAM Smart” Mobile App. The URL schemes are supported on iOS and Android versions of “iAM Smart” Mobile App. *Remark: appv1 does not allow one client id to support multiple app.*

- **Request Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required	`TicketID` is a unique identifier provided by "iAM Smart" System. Online service retrieves `ticketID` while requesting the Anonymous Form Filling function. It is an ASCII string with a length of less than or equal to 36 chars.
`source`	String	Required	App_Link/ App_Scheme Please use App_Link unless the support team approves.
`redirectURI`	String	Required	Callback redirect URI. The value should be **URL encoded** and **registered** in the self-service portal.
`state`	String	Required (Conditional)	If the `state` parameter is presented in the request message, the same state value will be returned to Online Service during the callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://anon_form-filling
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

5.5.4 Callback with authCode to Online Service App

- **URL Scheme and Package Name**

Name	Description
Service Full Name	Callback with authCode to Online Service App
URL Scheme (iOS and Android)	`<Universal / App Link>`. The online service custom app scheme or Universal / App Link should be registered with the "iAM Smart" System during onboarding.

Package name (Android only)	`<package name>` and `<activity name>`. The online service package name must be registered in the self-service portal. *Remark: Direct Login v2 (App) and appv2 API adopts package name verification instead of App Link.*
Description of Service	For appv1 API, the Online Service App will be invoked and launched by Universal / App Link. It makes use of deep linking to redirect users to the Online Services app. App Link can only work for mobile devices with Google Mobile Services (GMS). For appv2 API, the Android Online Service App will be invoked and launched by the package name. It make use of intent to redirect users to the Online Service app. The iOS Online Service App will be invoked and launched via Universal link. The Universal / App Link with the landing location as well as the package name plus activity name must be registered in the self-service portal and enabled by the support team.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	12 minutes	Online Service could treat the request as failed if it doesn't get callback within 12 minutes

- **URL Scheme Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required (Conditional)	`businessID` will only be returned in anonymous workflows. `businessID` is a unique identifier for Online Service to differentiate different requests. Online service can use the `businessID` to relate the callback message with the original request.
`code`	String	Required (Conditional)	The authorisation code is generated by the "iAM Smart" System. The authorisation code

			will be expired 1 minute after issuance. Online Service MUST NOT use the authorisation code more than once. An error message will be returned if an authorisation code is expired or re-used.
`state`	String	Optional	If the `state` parameter has been presented in the request message, the same state value will be returned. It is used to prevent the CSRF attack. The value of the state is defined by Online Service, and it should be a secure random value.
`error_code`	String	Required (Conditional)	Return error code when exception.

- **Example URL Scheme**

Allow

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
?businessID=b2c99aa83b0049e9ba370c5341681225
&code=0ad186353c424c64897fcc00445c9ba1
&state=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
?error_code=D20001
&state=eddd527b6
```

- **Example Package Name**

Allow

```
Intent ii=new Intent(<package name>, <activity class name>);
ii.putExtra("code", "0ad186353c424c64897fcc00445c9ba1");
ii.putExtra("state", "eddd527b6");
ii.putExtra("businessID", "b2c99aa83b0049e9ba370c5341681225");
startActivity(ii);
```

Deny

```
Intent ii=new Intent(<package name>, <activity name>);
ii.putExtra("error_code", "D20001");
```

```
ii.putExtra("state", "eddd527b6");
startActivity(ii);
```

5.5.5 Callback with authCode to Online Service Server

- **API Description**

Name	Description
Service Full Name	Callback with authCode to Online Service Server
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	GET
Service Version	V1.0.0
Description of Service	This callback is used to pass `authCode` to online service Server. The URI must be registered in the self-service portal.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	12 minutes	Online service could treat the request as failed if it doesn't get callback within 12 minutes.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required (Conditional)	`businessID` will only be returned in anonymous workflows. `businessID` is a unique identifier for online service to differentiate different requests. Online service can use the `businessID` to relate the callback message with the original request.
`code`	String	Required (Conditional)	The authorisation code is generated by the "iAM Smart" System. The authorisation code will be expired 1 minute after issuance. Online service MUST NOT use the authorisation code more than once. An error message will be returned if an authorisation code is expired or re-used.

state	String	Optional	If the state parameter has been presented in the request, the same state value will be returned during the callback. It is used to prevent the CSRF attack. The value of the state is defined by Online Service, and it should be a secure random value.
error_code	String	Required (Conditional)	Return error code when exception.

- **Example Callback**

Allow

```
// Line breaks are for legibility only.
GET
https://<call_back_endpoint>
?businessID=b2c99aa83b0049e9ba370c5341681225
&code=0ad186353c424c64897fcc00445c9ba1
&state=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
GET
https://<call_back_endpoint>
?error_code=D20001
&state=eddd527b6
```

5.5.6 Request accessToken & Tokenised ID

- **API Description**

Name	Description
Service Full Name	Request access token and tokenised ID with authCode
URI (as in RESTFUL API)	https://<iAM_Smart_domain>/api/v1/auth/getToken
Request Type	POST
Service Version	1.0.0
Description of Service	Online service uses this API to retrieve the access token and Tokenised ID (openID). An authorisation code is necessary during the process. The accessToken and openID will be used to call

	corresponding “iAM Smart” services subsequently.

- **Request Parameters**

Parameter	Type	Presence	Description
code	String	Required	The authorisation code is received from the authorisation server. One time use only and will be expired in 1 minute.
grantType	String	Required	the value MUST be set to authorization_code.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/auth/getToken
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
  "code": "xxxa42e76bf4cb0846a68e6d83d6096",
  "grantType": "authorization_code"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
accessToken	String	Required	accessToken value can only be used **once**.
tokenType	String	Required	Token type, support "Bearer" only
issueAt	Long	Required	The accessToken issue time is expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.

`expiresIn`	Long	Required	The lifetime in milliseconds of the token. The value may vary for different Online Services.
`openID`	String	Required	Tokenised ID, uniquely generated for each user of each online service website or mobile application.
`lastModifiedDate`	Long	Required	The datetime of the user complete registration at "iAM Smart" System. The value will be updated when either of the following is valid:- (1) If any one of the following verified data are changed. English name Chinese name (* not applicable if it was marked as unverified during registration) Gender Date of birth (2) User re-register "iAM Smart" after "iAM Smart" de-registration. The modification time will be expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
`userType`	String	Required	`default` or `sign` `default`: "iAM Smart" user `sign`: "iAM Smart+" user (digital signing capability)
`scope`	String	Required	The scope of the token. Please refer to the corresponding section specified in each API function.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
```

```
    "accessToken": "0ad186353c424c64897fcc00445c9ba1",
    "tokenType": "Bearer",
    "issueAt": 1557053922938,
    "expiresIn": 14400000,
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
    "lastModifiedDate": 1560849218006,
    "userType": "sign",
    "scope": "eidapi_formFilling"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D40004",
  "message": "authCode not exist or expired",
}
```

5.5.7 Obtain Anonymous Form Filling Result

- **API Description**

Name	Description
Service Full Name	Obtain Anonymous Form Filling Result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v2/anonymous/formFilling/getResult`
Request Type	POST
Service Version	V2.0.0
Description of Service	Online service can use this API to retrieve anonymous form filling result.

- **Request Parameters**

Parameter	Type	Presence	Description
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v2/anonymous/formFilling/getResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
    "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
idNo	JSON Object	Optional	HKIC number
prefix	String	Optional	prefix
enName	JSON Object	Optional	English name
chName	JSON Object	Optional	Chinese name
birthDate	String	Optional	Date of birth
gender	String	Optional	gender
maritalStatus	String	Optional	marital status
homeTelNumber	JSON Object	Optional	home telephone number
officeTelNumber	JSON Object	Optional	office telephone number
mobileNumber	JSON Object	Optional	mobile number
emailAddress	String	Optional	email address
residentialAddress	JSON Object	Optional	residential address If the data is retrieved from CDEG, “addressProvider” field will also be included. The “providerCode” and “lastRetrievalDate” fields indicate where and when the data was updated.

`postalAddress`	JSON Object	Optional	postal address If the data is retrieved from CDEG, "addressProvider" field will also be included. The "providerCode" and "lastRetrievalDate" fields indicate where and when the data was updated.
`educationLevel`	String	Optional	education level
`chNameVerified`	String	Optional	ImmD characters code point will be returned to online service if the Chinese name is verified.
`addressDocInfo`	JSON Object	Optional	the data related to the obtained e-bill, including the provider name, retrieval date, owner name, service address and postal address. The sub-field `lastRetrievalDate` refers to the date and time at which "iAM Smart" obtained the e-bill. The schema of sub-fields is given in **Appendix A**.
`addressDocFile`	JSON Object	Optional	the e-bill obtained from an address data provider. Sub-fields include the PDF e-bill file (`docFile`), the SHA256 hash (`docHash`) and the bill date of the e-bill (`billDate`). The schema of sub-fields is given in **Appendix A**.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// Decrypted Body
{
   "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
   "code": "D00000",
   "message": "SUCCESS",
   "content": {
      "idNo": {
         "Identification": "A123456",
         "CheckDigit": "A"
      },
      "prefix": "Mr",
```

```
        "enName": {
            "UnstructuredName": "SAN, Chi Nan"
        },
        "chName": {
            "ChineseName": "申智能"
        },
        "chNameVerified": "申智能",
        // May be 19960000 or 19960100
        "birthDate": "19960128",
        "gender": "M",
        "maritalStatus": "S",
        "homeTelNumber": {
            "CountryCode": "852",
            "SubscriberNumber": "98765432"
        },
        "officeTelNumber": {
            "CountryCode": "1",
            "SubscriberNumber": "123456"
        },
        "mobileNumber": {
            "CountryCode": "1",
            "SubscriberNumber": "98765432"
        },
        "emailAddress": "scn@digitalpolicy.gov.hk",

         // Three different tags exist in residentialAddress
        //  and the returned value  can be either one of the following
        //  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress

        // Both ChiPremisesAddress and EngPremisesAddress contain either
        // 1): Standard/Village Address
        // 2): Lot Address

       //   Example of ChiPremisesAddress with Standard/Village Address
        "residentialAddress": {
            "ChiPremisesAddress": {
                "Region": "香港",
                "ChiDistrict": {
                    "DcDistrict": "WC",
                    "Sub-district": "灣仔"
                },
                "BuildingName": "灣仔政府大樓",
                "ChiEstate": {
                    "EstateName": "華富",
                    "ChiPhase": {
                        "PhaseName": "華清"
                    }
                },
                "ChiStreet": {
```

```
                "StreetName": "港灣道",
                "BuildingNoFrom": "12"
            },
            "ChiBlock": {
                "BlockDescriptor": "座",
                "BlockNo": "東"
            },
            "Chi3dAddress": {
                "ChiFloor": {
                    "FloorNum": "15"
                },
                "ChiUnit": {
                    "UnitDescriptor": "室",
                    "UnitNo": "A1"
                }
            }
        } ,

        //  If the data is retrieved from CDEG, "addressProvider" field
        //  will also be included.
        "addressProvider": {
            "providerCode": "wsd",
            "lastRetrievalDate": 1560853718006
        }
    },

    //  Example of EngPremisesAddress with Lot address
    "residentialAddress": {
        "EngPremisesAddress": {
            "EngLot": {
                "EngStructuredLot": {
                    "DdType": "DD",
                    "DdNo": "110",
                    "LotType": "LOT",
                    "LotNo": "157",
                    "LotSection1": "A",
                    "LotSubsection1": "1",
                    "LotSection2": "B",
                    "LotSubsection2": "1",
                    "LotSection3": "AA",
                    "LotSubsection3": "1",
                    "LotExtendPortionCode": "3"
                }
            }
        }
    },

    //  Example of FreeFormatAddress
    "residentialAddress": {
        "FreeFormatAddress": {
```

```
            "LanguageCode": "en",
            "AddressLine1": "Unit 2000, 200/F",
            "AddressLine2": "5033 Yitian Road, Futian CBD",
            "AddressLine3": "Futian district, Shenzhen"
        }
    },

     // Four different tags exist in postalAddress
    //  The returned value of postalAddress can be either one of the
    //  following
    //  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress,
    //  PostBoxAddress

    //  Apart from PostBoxAddress, the other three tags are equivalent to
    //  ResidentialAddress

    //  PostBoxAddress contains either
        // 1): EngPostBox
        // 2): ChiPostBox

      // Example of postalAddress with EngPostBox
    "postalAddress": {
        "PostBoxAddress": {
             "EngPostBox": {
                "PoBoxNo": 24700,
                "PostOffice": "ABERDEEN POST OFFICE",
                "PostOfficeRegion": "HONG KONG"
            }
        }
    },
    "educationLevel": "T",
    // all the sub-fields come from the address data provider
    "addressDocInfo": {
        // name of the address data provider, possible values are:
        // The Hong Kong and China Gas Company Limited
        // CLP Power Hong Kong Ltd.
        // HK Electric
        // Water Supplies Department
        "enProviderName": "HK Electric",
        "tcProviderName": "港燈",
        "scProviderName": "港灯",
        // unstructured name of the e-bill owner
        "enUserName": "SAN, Chi Nan",
        "tcUserName": "申智能",
        // service address of the e-bill, at most 4 lines
        "enServiceAddress": {
            "addressLine1": "19/F, 213 Queen's Road East",
            "addressLine2": "Wanchai, Hong Kong"
        },
        "tcServiceAddress": {
```

```
                "addressLine1": "香港灣仔皇后大道東231號",
                "addressLine2": "胡忠大廈19樓"
            },
            // postal address of the e-bill, at most 4 lines
            "enPostalAddress": {
                "addressLine1": "19/F, 213 Queen's Road East",
                "addressLine2": "Wanchai, Hong Kong"
            },
            "tcPostalAddress": {
                "addressLine1": "香港灣仔皇后大道東231號",
                "addressLine2": "胡忠大廈19樓"
            },
            // last retrieval date of the e-bill
            // expressed in the number of milliseconds since 1970/1/1 GMT
            "lastRetrievalDate": 1560853718006
        }
        "addressDocFile": {
            // Base64 encoded string of the PDF e-bill file
            "docFile": "JVBERi0xLjUKJcK1wrXCtcK1CjEgMCBv…iag==",
            // SHA256 hash of the PDF e-bill file
            "docHash": "69e3…c4e5",
            // bill date of the e-bill
            // expressed in the number of milliseconds since 1970/1/1 GMT
            "billDate": 1560849218006
        }
    }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {accessToken}"
}
```

5.5.8 Open “iAM Smart” Mobile App for Anonymous Form Filling (appv2)

- **Using URL Scheme**

Name	Description
Service Full Name	Open “iAM Smart” Mobile App for Anonymous Form Filling
URI (as in RESTFUL API)	<“iAM Smart” app URL scheme>://v2_anon_form-filling
Service Version	V2.0.0

Description of Service	The URL scheme makes use of deep linking to redirect users to specific Form Filling in "iAM Smart" Mobile App. The URL schemes are supported on iOS and Android versions of "iAM Smart" Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required	`TicketID` is a unique identifier provided by "iAM Smart" System. Online service retrieves `ticketID` while requesting the Anonymous Form Filling function. It is an ASCII string with a length of less than or equal to 36 chars.
`source`	String	Required	App_Link (iOS) or App_Package (Android)
`clientRedirectURI`	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (",") are not accepted.
`packageName`	String	Required (Android only)	Package name of the Online Service App for callback use.
`activityClass`	String	Required (Android only)	Activity class name of the Online Service App for callback use.
`activityParams`	String	Optional (Android only)	Optional params used during callback.
`state`	String	Required (Conditional)	If the `state` parameter is presented in the request message, the same state value will be returned to Online Service during the callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://anon_form-filling
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
&source=App_Package
&packageName=com.onlineservice.myapp
&activityClass=callback_activity
&activityParams=callback_param
```

5.6 API Deprecation

The following section lists the deprecation period of the respective APIs. The Deprecation Date is the start date of the deprecation period, while Shutdown Date is the end date of the deprecation period. After the Shutdown Date, the API request will no longer be available.

5.6.1 Scope

Scope	Description
`eidapi_eMe`	Scope for Form Filling (v1)

5.6.2 Anonymous Form Filling (V1)

Version	Deprecation Date	Shutdown Date	Details
V1.0.0	1 October 2022	31 December 2025	On or after 31 December 2025, the existing online service will not be able to reach the endpoint (/api/v1/anonymous/eme/initiateRequest) and endpoint (/api/v1/anonymous/eme/getResult). Online service has to update the testing and production application form for the migration process.

5.6.2.1 Request Anonymous Form Filling

- **API Description**

Name	Description
Service Full Name	Request Anonymous Form Filling
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/anonymous/eme/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to initiate anonymous form filling request.
Deprecation Date	1 October 2022

Shutdown Date	31 December 2025

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`formName`	String	Optional	The name of the form should be encoded in Unicode. Maximum Length: 255
`formNum`	String	Optional	The number of the form should be encoded in Unicode. Maximum Length: 20
`formDesc`	String	Optional	The description of the form should be encoded in Unicode. Maximum Length: 255
`eMEFields`	Array	Required	Specify the "e-ME" fields to be requested. If eMEFields is not provided or is an empty array, the HTTP code 200 with error code D20002 (empty parameter {eMEFields}) will be returned. The available eMEFields are as follows:

eMEFields	Description
idNo	ID number
prefix	prefix
enName	English name
chName	Chinese name
birthDate	Date of birth
gender	gender
maritalStatus	marital status
homeTelNumber	home telephone number
officeTelNumber	office telephone number
mobileNumber	mobile number
emailAddress	email address
residentialAddress	residential address

			postalAddress	postal address
			educationLevel	education level
			addressDocInfo	provider name, retrieval date, owner name and address information related to an e-bill
			addressDocFile	e-bill obtained from an address data provider

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/anonymous/eme/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "formName": "Online Service form Name",
 "formNum": "FormNO_2019001",
 "formDesc": "form description",
 "eMEFields": ["idNo", "enName", "birthDate", "gender", "chName",
"addressDocInfo", "addressDocFile"]
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier that will be used to identify this

			request in the following steps of the form filling workflow.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "ticketID": "561b6f1ce5a043ba8aae33fb5c50386d"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {eMEFields}"
}
```

5.6.2.2 Obtain Anonymous Form Filling Result

- **API Description**

Name	Description
Service Full Name	Obtain Anonymous Form Filling Result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/anonymous/eme/getResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to retrieve anonymous form filling.
Deprecation Date	1 October 2022
Shutdown Date	31 December 2025

- **Request Parameters**

Parameter	Type	Presence	Description
`accessToken`	String	Required	`accessToken` value

openID	String	Required	Tokenised ID value

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/anonymous/eme/getResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
    "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
idNo	JSON Object	Optional	HKIC number
prefix	String	Optional	prefix
enName	JSON Object	Optional	English name
chName	JSON Object	Optional	Chinese name
birthDate	String	Optional	Date of birth
gender	String	Optional	gender
maritalStatus	String	Optional	marital status
homeTelNumber	JSON Object	Optional	home telephone number
officeTelNumber	JSON Object	Optional	office telephone number
mobileNumber	JSON Object	Optional	mobile number
emailAddress	String	Optional	email address
residentialAddress	JSON Object	Optional	residential address
posalAddress	JSON Object	Optional	postal address
educationLevel	String	Optional	education level
chNameVerified	String	Optional	ImmD characters code point will be returned to online service if the

			Chinese name is verified.
addressDocInfo	JSON Object	Optional	the data related to the obtained e-bill, including the provider name, retrieval date, owner name, service address and postal address. The sub-field `lastRetrievalDate` refers to the date and time at which "iAM Smart" obtained the e-bill. The schema of sub-fields is given in **Appendix A**.
addressDocFile	JSON Object	Optional	the e-bill obtained from an address data provider. Sub-fields include the PDF e-bill file (`docFile`), the SHA256 hash (`docHash`) and the bill date of the e-bill (`billDate`). The schema of sub-fields is given in **Appendix A**.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
   "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
   "code": "D00000",
   "message": "SUCCESS",
   "content": {
      "idNo": {
         "Identification": "A123456",
         "CheckDigit": "A"
      },
      "prefix": "Mr",
      "enName": {
         "UnstructuredName": "SAN, Chi Nan"
      },
      "chName": {
         "ChineseName": "申智能"
      },
      "chNameVerified": "申智能",
      // May be 19960000 or 19960100
      "birthDate": "19960128",
      "gender": "M",
      "maritalStatus": "S",
      "homeTelNumber": {
         "CountryCode": "852",
         "SubscriberNumber": "98765432"
```

```
    },
    "officeTelNumber": {
       "CountryCode": "1",
       "SubscriberNumber": "123456"
    },
    "mobileNumber": {
       "CountryCode": "1",
       "SubscriberNumber": "98765432"
    },
    "emailAddress": "scn@digitalpolicy.gov.hk",

     // Three different tags exist in residentialAddress
    //  and the returned value  can be either one of the following
    //  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress

    // Both ChiPremisesAddress and EngPremisesAddress contain either
    // 1): Standard/Village Address
    // 2): Lot Address

   //   Example of ChiPremisesAddress with Standard/Village Address
    "residentialAddress": {
       "ChiPremisesAddress": {
          "Region": "香港",
          "ChiDistrict": {
             "DcDistrict": "WC",
             "Sub-district": "灣仔"
          },
          "BuildingName": "灣仔政府大樓",
          "ChiEstate": {
             "EstateName": "華富",
             "ChiPhase": {
                "PhaseName": "華清"
             }
          },
          "ChiStreet": {
             "StreetName": "港灣道",
             "BuildingNoFrom": "12"
          },
          "ChiBlock": {
             "BlockDescriptor": "座",
             "BlockNo": "東"
          },
          "Chi3dAddress": {
             "ChiFloor": {
                "FloorNum": "15"
             },
             "ChiUnit": {
                "UnitDescriptor": "室",
                "UnitNo": "A1"
```

```
            }
        }
    }
},

//  Example of EngPremisesAddress with Lot address
"residentialAddress": {
    "EngPremisesAddress": {
        "EngLot": {
            "EngStructuredLot": {
                "DdType": "DD",
                "DdNo": "110",
                "LotType": "LOT",
                "LotNo": "157",
                "LotSection1": "A",
                "LotSubsection1": "1",
                "LotSection2": "B",
                "LotSubsection2": "1",
                "LotSection3": "AA",
                "LotSubsection3": "1",
                "LotExtendPortionCode": "3"
            }
        }
    }
},

//  Example of FreeFormatAddress
"residentialAddress": {
    "FreeFormatAddress": {
        "LanguageCode": "en",
        "AddressLine1": "Unit 2000, 200/F",
        "AddressLine2": "5033 Yitian Road, Futian CBD",
        "AddressLine3": "Futian district, Shenzhen"
    }
},

 // Four different tags exist in postalAddress
//  The returned value of postalAddress can be either one of the
//  following
//  ChiPremisesAddress, EngPremisesAddress, FreeFormatAddress,
//  PostBoxAddress

//  Apart from PostBoxAddress, the other three tags are equivalent to
//  ResidentialAddress

//  PostBoxAddress contains either
    // 1): EngPostBox
    // 2): ChiPostBox

  // Example of postalAddress with EngPostBox
```

```
        "postalAddress": {
            "PostBoxAddress": {
                 "EngPostBox": {
                    "PoBoxNo": 24700,
                    "PostOffice": "ABERDEEN POST OFFICE",
                    "PostOfficeRegion": "HONG KONG"
                }
            }
        },
        "educationLevel": "T",
        // all the sub-fields come from the address data provider
        "addressDocInfo": {
            // name of the address data provider, possible values are:
            // The Hong Kong and China Gas Company Limited
            // CLP Power Hong Kong Ltd.
            // HK Electric
            // Water Supplies Department
            "enProviderName": "HK Electric",
            "tcProviderName": "港燈",
            "scProviderName": "港灯",
            // unstructured name of the e-bill owner
            "enUserName": "SAN, Chi Nan",
            "tcUserName": "申智能",
            // service address of the e-bill, at most 4 lines
            "enServiceAddress": {
                "addressLine1": "19/F, 213 Queen's Road East",
                "addressLine2": "Wanchai, Hong Kong"
            },
            "tcServiceAddress": {
                "addressLine1": "香港灣仔皇后大道東231號",
                "addressLine2": "胡忠大廈19樓"
            },
            // postal address of the e-bill, at most 4 lines
            "enPostalAddress": {
                "addressLine1": "19/F, 213 Queen's Road East",
                "addressLine2": "Wanchai, Hong Kong"
            },
            "tcPostalAddress": {
                "addressLine1": "香港灣仔皇后大道東231號",
                "addressLine2": "胡忠大廈19樓"
            },
            // last retrieval date of the e-bill
            // expressed in the number of milliseconds since 1970/1/1 GMT
            "lastRetrievalDate": 1560853718006
        }
        "addressDocFile": {
            // Base64 encoded string of the PDF e-bill file
            "docFile": "JVBERi0xLjUKJcK1wrXCtcK1CjEgMCBv…iag==",
            // SHA256 hash of the PDF e-bill file
```

```
            "docHash": "69e3…c4e5",
            // bill date of the e-bill
            // expressed in the number of milliseconds since 1970/1/1 GMT
            "billDate": 1560849218006
        }
    }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {accessToken}"
}
```

6. DIGITAL SIGNING WITH SERVICE LOGIN

6.1 Overview

“iAM Smart+” version supports digital signing with legal backing under the Electronic Transactions Ordinance (Cap. 553) for handling statutory documents and procedures. Online Services can make use of Digital Signing API to enable “iAM Smart” users complete digital signing online upon user consent. It can be used in many cases, such as digital signing online application form and digital signing contract and agreement.

6.2 Prerequisite

- Online service must undergo the authentication process (i.e. Request accessToken & Tokenised ID) in Section 3.4.5 in order to obtain the accessToken and openID as the input of Digital Signing API.
- To request the Digital Signing API, the “iAM Smart” user shall be a “iAM Smart+” user. Online service shall prompt the message in their website / mobile app and guide the user to upgrade his/her user account if he/she is not the “iAM Smart+” user.
- Online service shall refer to “Reference System Flow and User Interface Specifications for “iAM Smart” Adoption” for the detail UI/UX requirements.
- Online Service must submit the common parameter of "rateLimitFactor" while requesting the Step-up Authentication function.

6.3 Scope

Online service shall submit the application for using the following scope values to “iAM Smart” Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_sign`	Scope for Digital Signing

Online service may demand user to confirm the signing request by verifying his/her personal identity using NFC and/or FR. In this case, online service shall submit the application for using NFC and/or FR to “iAM Smart” Support team in advance and provide the required values based on the business requirement specified in the API request.

6.4 Use Cases and Scenarios

6.4.1 Digital Signing (Online Service Website/App in Different Device)

The sequence diagram below shows how an authenticated user authorises and signs the document hash when Online Service website/App and the "iAM Smart" Mobile App are running in different devices.

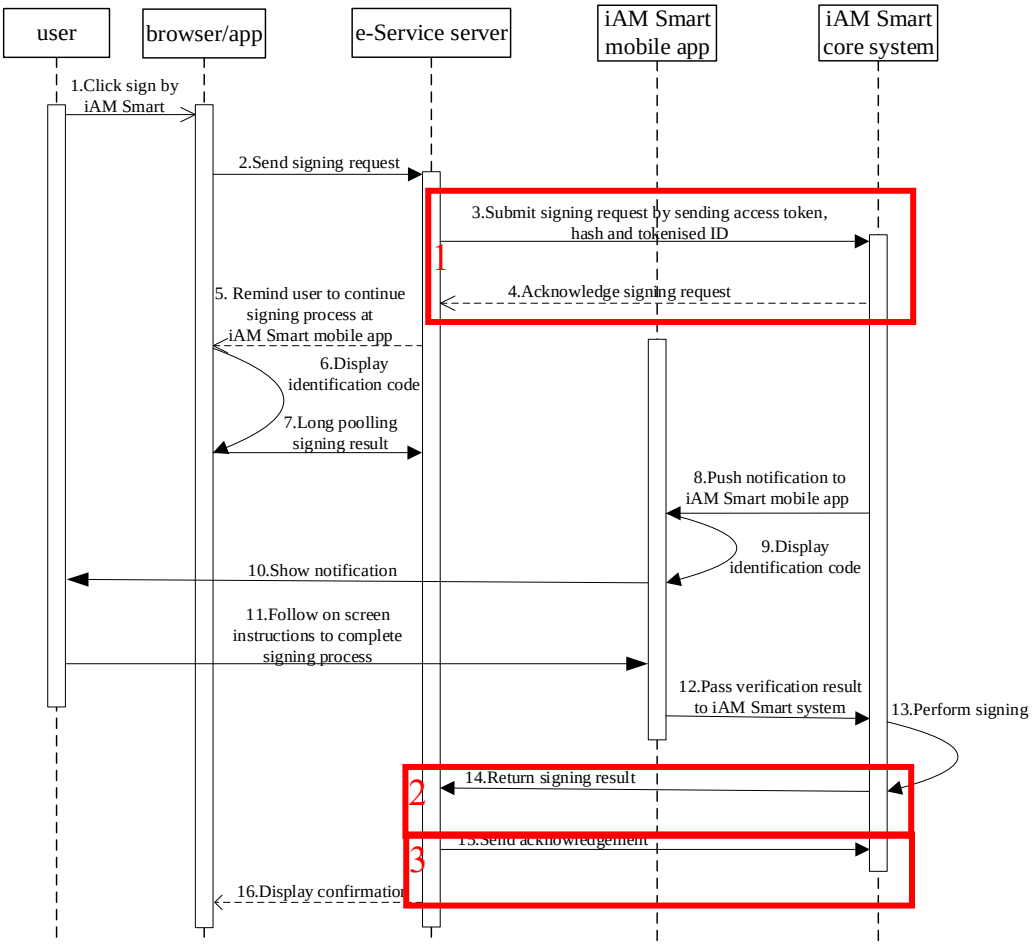

Figure-19 Digital Signing (Online Service Website/App in Different Device)

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1.	Request Digital Signing (appv1)	6.5.1

	Request PDF Digital Signing (appv1)	6.5.2
2.	Callback to Receive Digital Signing Result	6.5.4
	Callback to Receive PDF Digital Signing Result	6.5.5
3.	Online Service Acknowledges Digital Signing Result	6.5.6

6.4.2 Digital Signing (Online Service Website in Same Device)

The sequence diagram below shows how an authenticated user authorises and signs the document hash when online service website and the "iAM Smart" Mobile App are running in the same device.

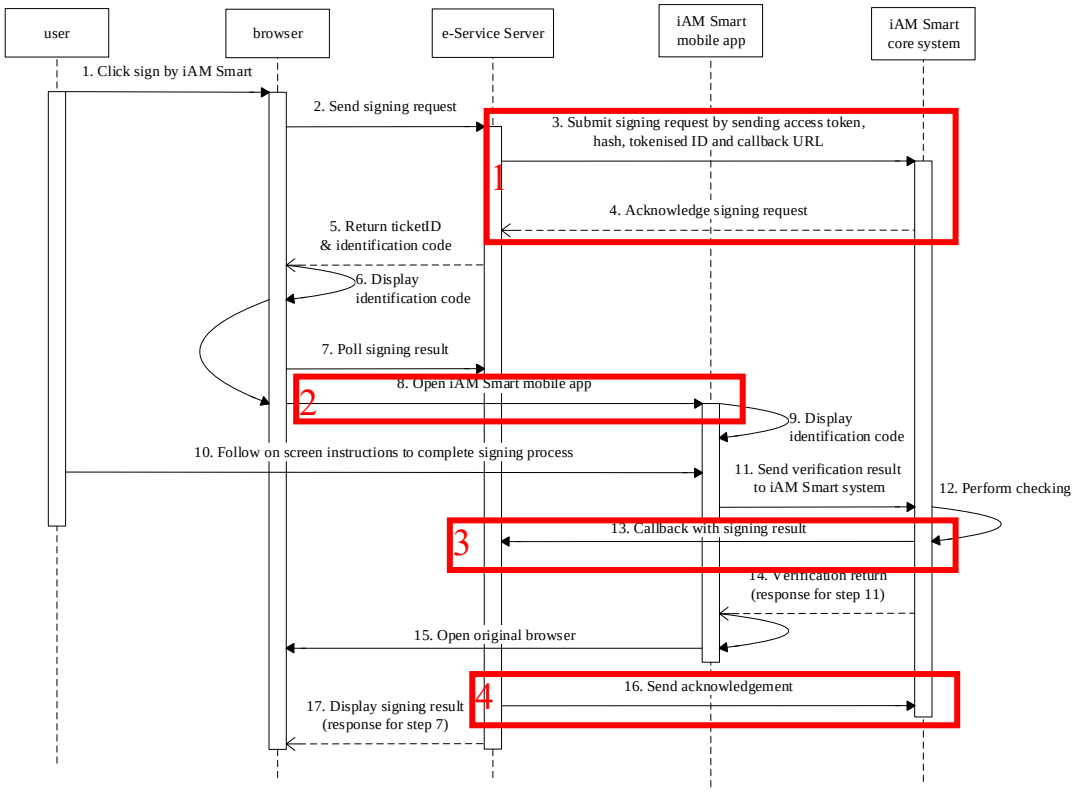

Figure-20 Digital Signing (Online Service Website in Same Device)

APIs interactions between online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Digital Signing (appv1)	6.5.1
	Request PDF Digital Signing (appv1)	6.5.2
2	Open "iAM Smart" Mobile App for Digital Signing	6.5.3
3	Callback to Receive Digital Signing Result	6.5.4
	Callback to Receive PDF Digital Signing Result	6.5.5
4	Online Service Acknowledges Digital Signing Result	6.5.6

6.4.3 Digital Signing (Online Service App in Same Device)

The sequence diagram below shows how an authenticated user authorises and signs the document hash when online service App and the "iAM Smart" Mobile App are running in the same device.

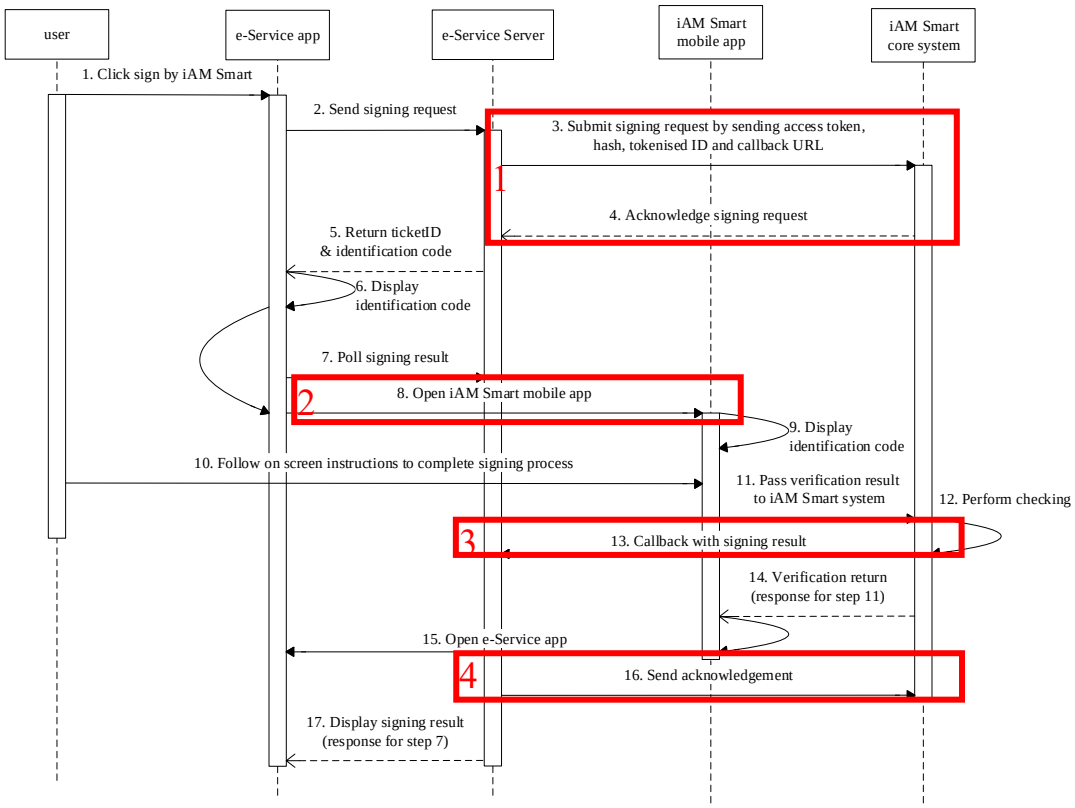

Figure-21 Digital Signing (Online Service App in Same Device)

APIs interactions between online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Digital Signing (appv1)	6.5.1
	Request Digital Signing (appv2)	6.5.7
	Request PDF Digital Signing (appv1)	6.5.2
	Request PDF Digital Signing (appv2)	6.5.8
2	Open "iAM Smart" Mobile App for Digital Signing	6.5.3
3	Callback to Receive Digital Signing Result	6.5.4

	Callback to Receive PDF Digital Signing Result	6.5.5
4	Online Service Acknowledges Digital Signing Result	6.5.6

6.5 API Implementation Details

6.5.1 Request Digital Signing (appv1)

- **API Description**

Name	Description
Service Full Name	Request digital signing
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/account/signing/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API making request for digital signing by sending `accessToken` and `openID` to "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value Maximum Length: 64
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.
`redirectURI`	String	Required	Callback URI.
`state`	String	Optional	If `state` parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII

			letters, numbers, underscore and hyphen are accepted.
`hashCode`	String	Required	the document hash to be signed. Online service should compute the hash and send this hash to "iAM Smart" System for digital signing. The value should be Base64 encoded.
`sigAlgo`	String	Optional	signature algorithm to be used. Online service can specify either `SHA256withRSA` or `NONEwithRSA`. The default value is `SHA256withRSA`. While using `NONEwithRSA`, `hashCode` provided must be hashed with SHA256.
`HKICHash`	String	Optional	Signing request will only be processed when the hash of the HKIC number of the "iAM Smart" user is matched with the HKICHash provided by Online Service. Online service should convert the HKIC number into hash value using SHA256 before sending to "iAM Smart" System. Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
`department`	String	Optional	The department that initiates the digital signing request. Maximum Length: 100
`serviceName`	String	Required	The online service name. Maximum Length: 255
`documentName`	String	Required	The document name that the user is going to sign. Maximum Length: 255
`suaMethod`	String (JSON)	Optional	`{"unary": ["FR"]}, {"unary": ["NFC"]},{"and": ["FR", "NFC"]}`. Online service should have been granted the requested step-up authentication method(s) in the e-Service Management Portal to use the `"NFC", "FR"` methods to verify the user. The API returns unsuccessful

			immediately if the first method fails. No matter if this parameter is specified, FIDO will be used to verify the user at the beginning as before.
suaWaitPeriod	Integer	Optional	If suaMethod is provided, this waiting period defines the minimum number of hours since the user account was created. If system defined waiting period is greater than suaWaitPeriod, greater value of system defined waiting period would be used. If suaWaitPeriod is greater than system defined waiting period, greater value of suaWaitPeriod would be used. System default value is zero. Maximum Value: 720

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/account/signing/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
   "source": "Android_Chrome",
   "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
   "state": "eddd527b6",
```

```
    "hashCode": " 965e638fda4c70f667efc2d68c40c6111e5965bfc82356d",
    "HKICHash": " 6ea37e641260714009a880a8057032a414009a80f667efc",
    "department": " Digital Policy Office",
    "serviceName": "Online Service1",
    "documentName": "Doc0001",
    "suaMethod": "{\"unary\": [\"FR\"]}"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If the user completes authentication by scanning the QR code (i.e. different device), then "true" will be returned to online service.
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier to trigger “iAM Smart” Mobile App when authByQR is `false` The value valid for 18 minutes.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {hashCode}"
}
```

6.5.2 Request PDF Digital Signing (appv1)

- **API Description**

Name	Description
Service Full Name	Request PDF digital signing
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/account/pdfsigning/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to request for PDF digital signing by sending `accessToken` and `openID` to "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value Maximum Length: 64
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.
`redirectURI`	String	Required	Callback URI.
`state`	String	Optional	If `state` parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
`docDigest`	String	Required	the PDF document digest to be signed. Online service should compute the digest using the Adobe.PPKLite filter and the

			adbe.pkcs7.detached subfilter. The value should be Base64 encoded.
HKICHash	String	Optional	Signing request will only be processed when the hash of the HKIC number of the "iAM Smart" user is matched with the HKICHash provided by Online Service. Online service should convert the HKIC number into hash value using SHA256 before sending to "iAM Smart" System. Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
department	String	Optional	The department that initiates the digital signing request. Maximum Length: 100
serviceName	String	Required	The online service name. Maximum Length: 255
documentName	String	Required	The document name that the user is going to sign. Maximum Length: 255
suaMethod	String (JSON)	Optional	`{"unary": ["FR"]}, {"unary": ["NFC"]},{"and": ["FR", "NFC"]}`. Online service should have been granted the requested step-up authentication method(s) in the e-Service Management Portal to use the `"NFC"`, `"FR"` methods to verify the user. The API returns unsuccessful immediately if the first method fails. No matter if this parameter is specified, FIDO will be used to verify the user at the beginning as before.
suaWaitPeriod	Integer	Optional	If `suaMethod` is provided, this waiting period defines the minimum number of hours since the user account was created. If system defined waiting period is greater than `suaWaitPeriod`, greater value of system defined waiting period would be

			used. If `suaWaitPeriod` is greater than system defined waiting period, greater value of `suaWaitPeriod` would be used. System default value is zero. Maximum Value: 720

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/account/pdfsigning/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
   "source": "Android_Chrome",
   "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
   "state": "eddd527b6",
   "docDigest": " 965e638fda4c70f667efc2d68c40c6111e5965bfc82356d",
   "HKICHash": " 6ea37e641260714009a880a8057032a414009a80f667efc",
   "department": " Digital Policy Office",
   "serviceName": "Online Service1",
   "documentName": "Doc0001",
   "suaMethod": "{\"unary\": [\"FR\"]}"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If the user completes authentication by scanning QR code (i.e. different device), then "true" will be returned to Online Service.
`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger "iAM Smart" Mobile App when authByQR is `false` The value valid for 18 minutes.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {docDigest}"
}
```

6.5.3 Open "iAM Smart" Mobile App for Digital Signing

- **Using URL Scheme**

Name	Description
Service Full Name	Open "iAM Smart" Mobile App for Digital Signing
URI (as in RESTFUL API)	For Digital Signing v1: <"iAM Smart" app URL scheme>://hash-sign Or

	<"iAM Smart" app URL scheme>://pdf-sign For Digital Signing appv2: <"iAM Smart" app URL scheme>://v2_hash-sign Or <"iAM Smart" app URL scheme>://v2_pdf-sign
Service Version	V1.0.0 and V2.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific Digital Signing in "iAM Smart" Mobile App. The URL schemes supported on iOS and Android versions of "iAM Smart" Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
ticketID	String	Required	ticketID is a unique identifier provided by "iAM Smart" System, online service retrieve ticketID while requesting the digital signing function. It is ASCII string with length less than or equal 36 chars.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://v2_hash-sign
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

6.5.4 Callback to Receive Digital Signing Result

- **API Description**

Name	Description
Service Full Name	Callback to Receive Digital Signing Result
URI (as in RESTFUL API)	https://<rp_domain>/<rp_context>/<call_back_endpoint>
Request Type	POST
Service Version	1.0
Description of Service	This callback will provide the digital signing result to Online Service. Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. Online service can use the `businessID` to relate the callback message with the original request.
`state`	String	Optional	If the `state` parameter has been present in the request message, the exact value of state will be returned. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random value.
`hashCode`	String	Optional	The document hash submitted by online service in the API request.
`timestamp`	Long	Optional	Timestamp in milliseconds since January 1, 1970 00:00:00 GMT. “iAM Smart” System will provide this to online service only when digital signing is successful.
`signature`	String	Optional	Base64-encoded signature result string. “iAM Smart” System will provide this to online service only when digital signing is successful.
`cert`	String	Optional	Base64-encoded DER format certificate for the “iAM Smart” user. “iAM Smart” System will provide this to online service only when the digital signing is successful.

- **Example Callback**

```
// The descriptions of txID, code, and message are in Section 2.4.2
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>
// Callback Body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
          "content": {
            "businessID": "2YotnFZFEjr1zCsicMWpAA",
            "state": "unesidkd",
            "hashCode": "tGzv3JOkF0XG5Qx2TlKWIA",
            "timestamp": 1556450176000,
            "signature": "nnoadisauflanefhykdjf",
            "cert": "sdfGSDGsdfaGDEHfjslgGQG……GSGjljlkjwmh",
          }
}
```

6.5.5 Callback to Receive PDF Digital Signing Result

- **API Description**

Name	Description
Service Full Name	Callback to Receive PDF Digital Signing Result
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	1.0
Description of Service	This callback will provide the pdf digital signing result to Online Service. Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **Callback Parameters**

Parameter	Type	Presence	Description
businessID	String	Required	businessID is a unique identifier for online service to differentiate different request. Online service can use the businessID to relate the callback message with the original request.
state	String	Optional	If the state parameter has been present in the request message, the exact value of state will be returned. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random value.
docDigest	String	Optional	The pdf document digest submitted by online service in the API request.
pdfSignature	String	Optional	Base64-encoded PKCS#7 object that is the actual PDF signature value. It contains signer's certificate, signed hash value, and the digital signing timestamp information. Online service can embed this value to the PDF document for future verification. "iAM Smart" System will provide this to online service only when the digital signing is successful.

- **Example Callback**

```
// The descriptions of txID, code, and message are in Section 2.4.2
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>
// Callback Body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "businessID": "2YotnFZFEjr1zCsicMWpAA",
    "state": "unesidkd",
    "docDigest": "tGzv3JOkF0XG5Qx2TlKWIA",
```

```
    "pdfSignature": "sdfGSDGsdfaGDEHfjslgGQG……GSGjljlkjwmh",
  }
}
```

6.5.6 Online Service Acknowledges Digital Signing Result

- **API Description**

Name	Description
Service Full Name	Online Service acknowledges digital signing result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/account/signing/ackResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service calls this API to acknowledge “iAM Smart” System if the result of the digital signature is accepted or not.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`signingResult`	String	Required	"SR001": digital signature is accepted "SR002": digital signature is rejected "SR003": no digital signature was received

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/account/signing/ackResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
```

```
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
  "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
  "signingResult": "SR001",
}
```

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS"
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {signingResult}"
}
```

6.5.7 Request Digital Signing (appv2)

- **API Description**

Name	Description
Service Full Name	Request digital signing
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/appv2/account/signing/initiateRequest`
Request Type	POST
Service Version	V2.0.0
Description of Service	Online service can use this API making request for digital signing by sending `accessToken` and `openID` to "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value Maximum Length: 64
`source`	String	Required	App_Link (iOS) or App_Package (Android)
`clientRedirectURI`	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (",") are not accepted.
`packageName`	String	Required (Android only)	Package name of the Online Service App for callback use.
`activityClass`	String	Required (Android only)	Activity class name of the Online Service App for callback use.
`activityParams`	String	Optional (Android only)	Optional params used during callback.
`serverRedirectURI`	String	Required	Callback URI.

`state`	String	Optional	If `state` parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
`hashCode`	String	Required	the document hash to be signed. Online service should compute the hash and send this hash to "iAM Smart" System for digital signing. The value should be Base64 encoded.
`sigAlgo`	String	Optional	signature algorithm to be used. Online service can specify either `SHA256withRSA` or `NONEwithRSA`. The default value is `SHA256withRSA`. While using `NONEwithRSA`, `hashCode` provided must be hashed with SHA256.
`HKICHash`	String	Optional	Signing request will only be processed when the hash of the HKIC number of the "iAM Smart" user is matched with the HKICHash provided by Online Service. Online service should convert the HKIC number into hash value using SHA256 before sending to "iAM Smart" System. Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
`department`	String	Optional	The department that initiates the digital signing request. Maximum Length: 100
`serviceName`	String	Required	The online service name. Maximum Length: 255

documentName	String	Required	The document name that the user is going to sign. Maximum Length: 255
suaMethod	String (JSON)	Optional	{"unary": ["FR"]}, {"unary": ["NFC"]},{"and": ["FR", "NFC"]}. Online service should have been granted the requested step-up authentication method(s) in the e-Service Management Portal to use the "NFC", "FR" methods to verify the user. The API returns unsuccessful immediately if the first method fails. No matter if this parameter is specified, FIDO will be used to verify the user at the beginning as before.
suaWaitPeriod	Integer	Optional	If suaMethod is provided, this waiting period defines the minimum number of hours since the user account was created. If system defined waiting period is greater than suaWaitPeriod, greater value of system defined waiting period would be used. If suaWaitPeriod is greater than system defined waiting period, greater value of suaWaitPeriod would be used. System default value is zero. Maximum Value: 720

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/appv2/account/signing/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
```

```
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
   "source": "App_Package",

   "packageName": "com.onlineservice.myapp",
   "activityClass": "callback_activity",
   "activityParams": "callback_param",
   "serverRedirectURI":
"https://<rp_domain>/<rp_context>/<call_back_endpoint>",
   "state": "eddd527b6",
   "hashCode": " 965e638fda4c70f667efc2d68c40c6111e5965bfc82356d",
   "HKICHash": " 6ea37e641260714009a880a8057032a414009a80f667efc",
   "department": " Digital Policy Office",
   "serviceName": "Online Service1",
   "documentName": "Doc0001",
   "suaMethod": "{\"unary\": [\"FR\"]}"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If the user completes authentication by scanning the QR code (i.e. different device), then "true" will be returned to online service. `authByQR` should always be `false` in appv2 API.
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier to trigger “iAM Smart” Mobile App when authByQR is `false`

			The value valid for 18 minutes.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": false,
    "ticketID": "bbb8aae57c104cda40c93843ad5e6db8"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {hashCode}"
}
```

6.5.8 Request PDF Digital Signing (appv2)

- **API Description**

Name	Description
Service Full Name	Request PDF digital signing
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/appv2/account/pdfsigning/initiateRequest`
Request Type	POST
Service Version	V2.0.0
Description of Service	Online service can use this API to request for PDF digital signing by sending `accessToken` and `openID` to "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different

			request. It should be ASCII string with length less than or equal to 36 chars.
accessToken	String	Required	accessToken value
openID	String	Required	Tokenised ID value Maximum Length: 64
source	String	Required	App_Link (iOS) or App_Package (Android)
clientRedirectURI	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (“,”) are not accepted.
packageName	String	Required (Android only)	Package name of the Online Service App for callback use.
activityClass	String	Required (Android only)	Activity class name of the Online Service App for callback use.
activityParams	String	Optional (Android only)	Optional params used during callback.
serverRedirectURI	String	Required	Callback URI.
state	String	Optional	If state parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
docDigest	String	Required	the PDF document digest to be signed. Online service should compute the digest using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter. The value should be Base64 encoded.
HKICHash	String	Optional	Signing request will only be processed

			when the hash of the HKIC number of the “iAM Smart” user is matched with the HKICHash provided by Online Service. Online service should convert the HKIC number into hash value using SHA256 before sending to “iAM Smart” System. Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
`department`	String	Optional	The department that initiates the digital signing request. Maximum Length: 100
`serviceName`	String	Required	The online service name. Maximum Length: 255
`documentName`	String	Required	The document name that the user is going to sign. Maximum Length: 255
`suaMethod`	String (JSON)	Optional	`{"unary": ["FR"]}, {"unary": ["NFC"]},{"and": ["FR", "NFC"]}`. Online service should have been granted the requested step-up authentication method(s) in the e-Service Management Portal to use the `"NFC", "FR"` methods to verify the user. The API returns unsuccessful immediately if the first method fails. No matter if this parameter is specified, FIDO will be used to verify the user at the beginning as before.
`suaWaitPeriod`	Integer	Optional	If suaMethod is provided, this waiting period defines the minimum number of hours since the user account was created. If system defined waiting period is greater than `suaWaitPeriod`, greater value of system defined waiting

			period would be used. If `suaWaitPeriod` is greater than system defined waiting period, greater value of `suaWaitPeriod` would be used. System default value is zero. Maximum Value: 720

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/appv2/account/pdfsigning/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
   "source": "App_Package",
   "packageName": "com.onlineservice.myapp",
   "activityClass": "callback_activity",
   "activityParams": "callback_param",
   "serverRedirectURI":
"https://<rp_domain>/<rp_context>/<call_back_endpoint>",
   "state": "eddd527b6",
   "docDigest": " 965e638fda4c70f667efc2d68c40c6111e5965bfc82356d",
   "HKICHash": " 6ea37e641260714009a880a8057032a414009a80f667efc",
   "department": " Digital Policy Office",
   "serviceName": "Online Service1",
   "documentName": "Doc0001",
```

```
    "suaMethod": "{\"unary\": [\"FR\"]}"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If the user completes authentication by scanning QR code (i.e. different device), then "true" will be returned to Online Service. `authByQR` should always be `false` in appv2 API.
`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger “iAM Smart” Mobile App when authByQR is `false` The value valid for 18 minutes.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": false,
    "ticketID": "bbb8aae57c104cda40c93843ad5e6db8"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {docDigest}"
}
```

7. DIGITAL SIGNING WITHOUT SERVICE LOGIN (AKA ANONYMOUS DIGITAL SIGNING)

7.1 Overview

"iAM Smart+" version supports digital signing with legal backing under the Electronic Transactions Ordinance (Cap. 553) for handling statutory documents and procedures. Similar to Digital Signing API, online service can request "Anonymous Digital Signing" API to complete digital signing online upon user consent. Unlike the accessToken obtained from Authentication API, the accessToken received in this API can only be used once. It can be used in many cases, such as digital signing online application form and digital signing contract and agreement.

7.2 Prerequisite

- To request the Anonymous Digital Signing API, the "iAM Smart" user shall be a "iAM Smart+" user. Online service shall prompt the message in their website / mobile app and guide the user to upgrade his/her user account if he/she is not the "iAM Smart+" user.
- Online service shall refer to "Reference System Flow and User Interface Specifications for "iAM Smart" Adoption" for the detail UI/UX requirements.
- Online Service must submit the common parameter of "rateLimitFactor" while requesting the Step-up Authentication function.

7.3 Scope

Online service shall submit the application for using the following scope values to "iAM Smart" Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_sign`	Scope for Digital Signing

Online service may demand user to confirm the signing request by verifying his/her personal identity using NFC and/or FR. In this case, online service shall submit the application for using NFC and/or FR to "iAM Smart" Support team in advance and provide the required values based on the business requirement specified in the API request.

7.4 Use Cases and Scenarios

7.4.1 Anonymous Digital Signing (Online Service Website in Different Device)

The sequence diagram below shows how an anonymous user authorises and signs the document hash when online service website and the "iAM Smart" Mobile App are running in different devices.

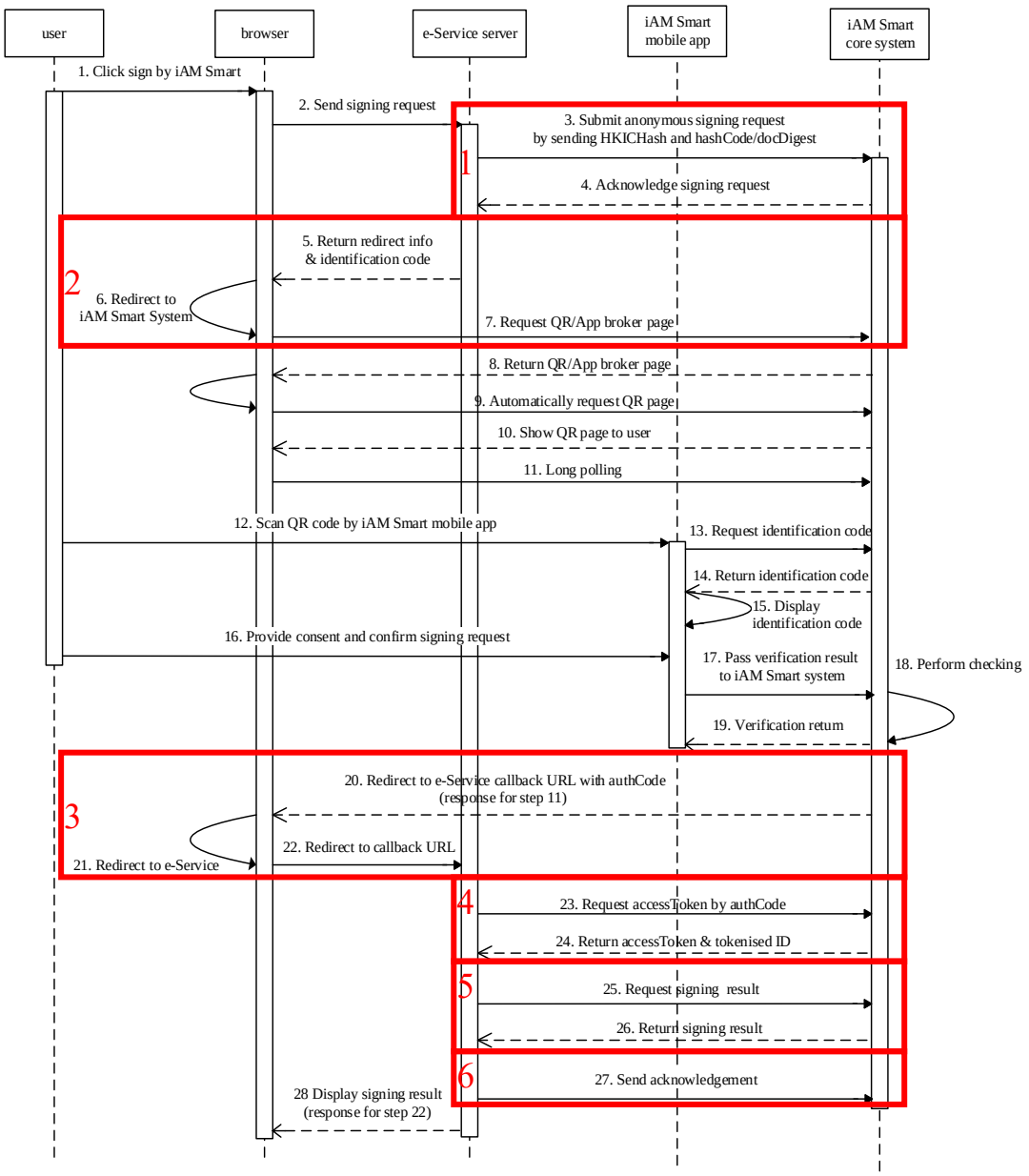

Figure-22 Anonymous Digital Signing (Online Service Website in Different Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Digital Signing	7.5.1
	Request Anonymous PDF Digital Signing	7.5.2
2	Request QR Page	7.5.3
3	Callback with authCode to Online Service Server	7.5.6
4	Request accessToken & Tokenised ID	7.5.7
5	Obtain Anonymous Digital Signing Result	7.5.8
	Obtain Anonymous PDF Digital Signing Result	7.5.9
6	Online Service Acknowledges Digital Signing Result	7.5.10

7.4.2 Anonymous Digital Signing (Online Service Website in Same Device)

The sequence diagram below shows how an anonymous user authorises and signs the document hash when online service website and the "iAM Smart" Mobile App are running in the same devices.

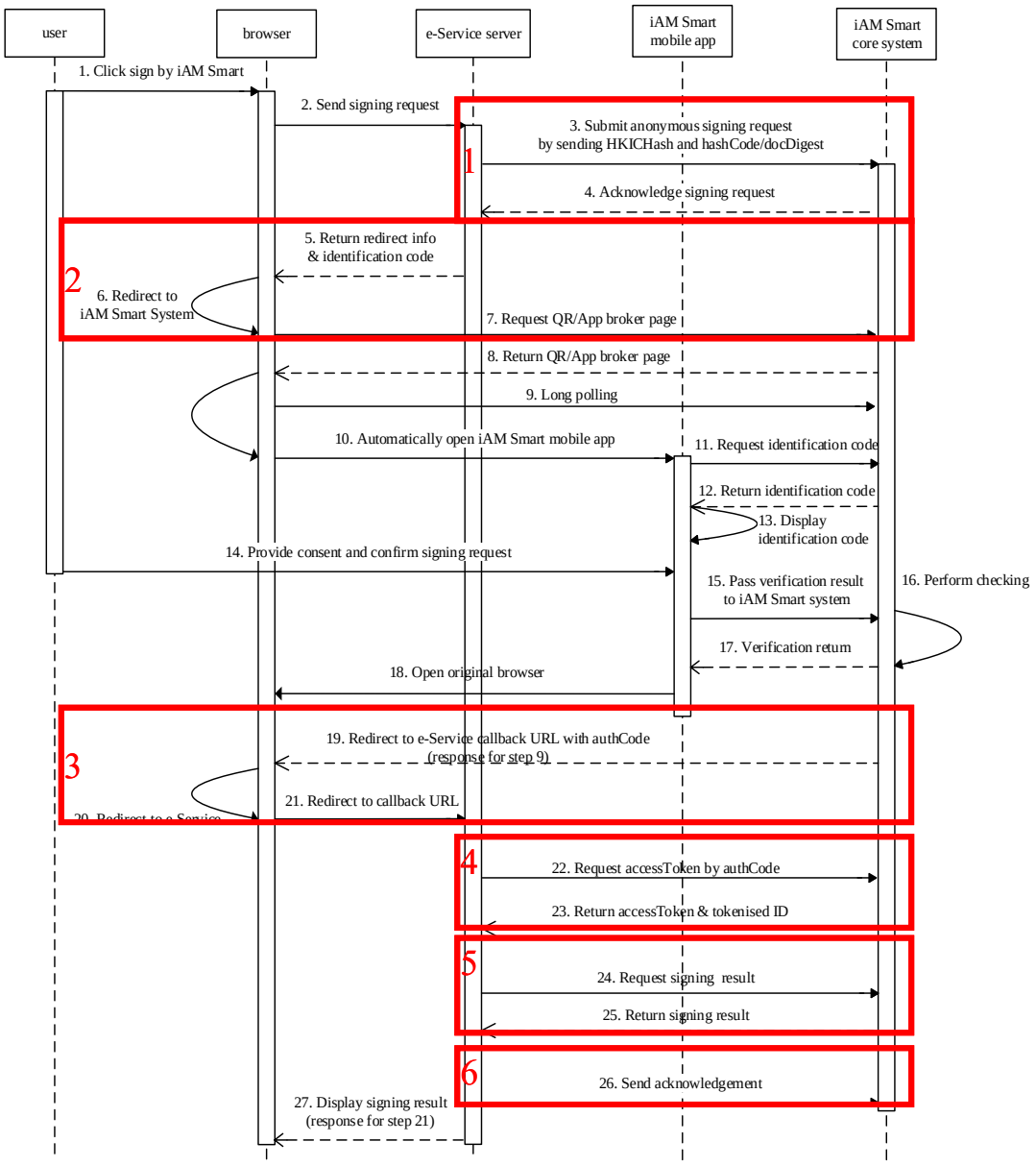

Figure-23 Anonymous Digital Signing (Online Service Website in Same Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Digital Signing	7.5.1
	Request Anonymous PDF Digital Signing	7.5.2
2	Request QR Page	7.5.3
3	Callback with authCode to Online Service Server	7.5.6
4	Request accessToken & Tokenised ID	7.5.7
5	Obtain Anonymous Digital Signing Result	7.5.8
	Obtain Anonymous PDF Digital Signing Result	7.5.9
6	Online Service Acknowledges Digital Signing Result	7.5.10

7.4.3 Anonymous Digital Signing (Online Service App in Different Device)

The sequence diagram below shows how an anonymous user authorises and signs the document hash when online service App and the "iAM Smart" Mobile App are running in different devices.

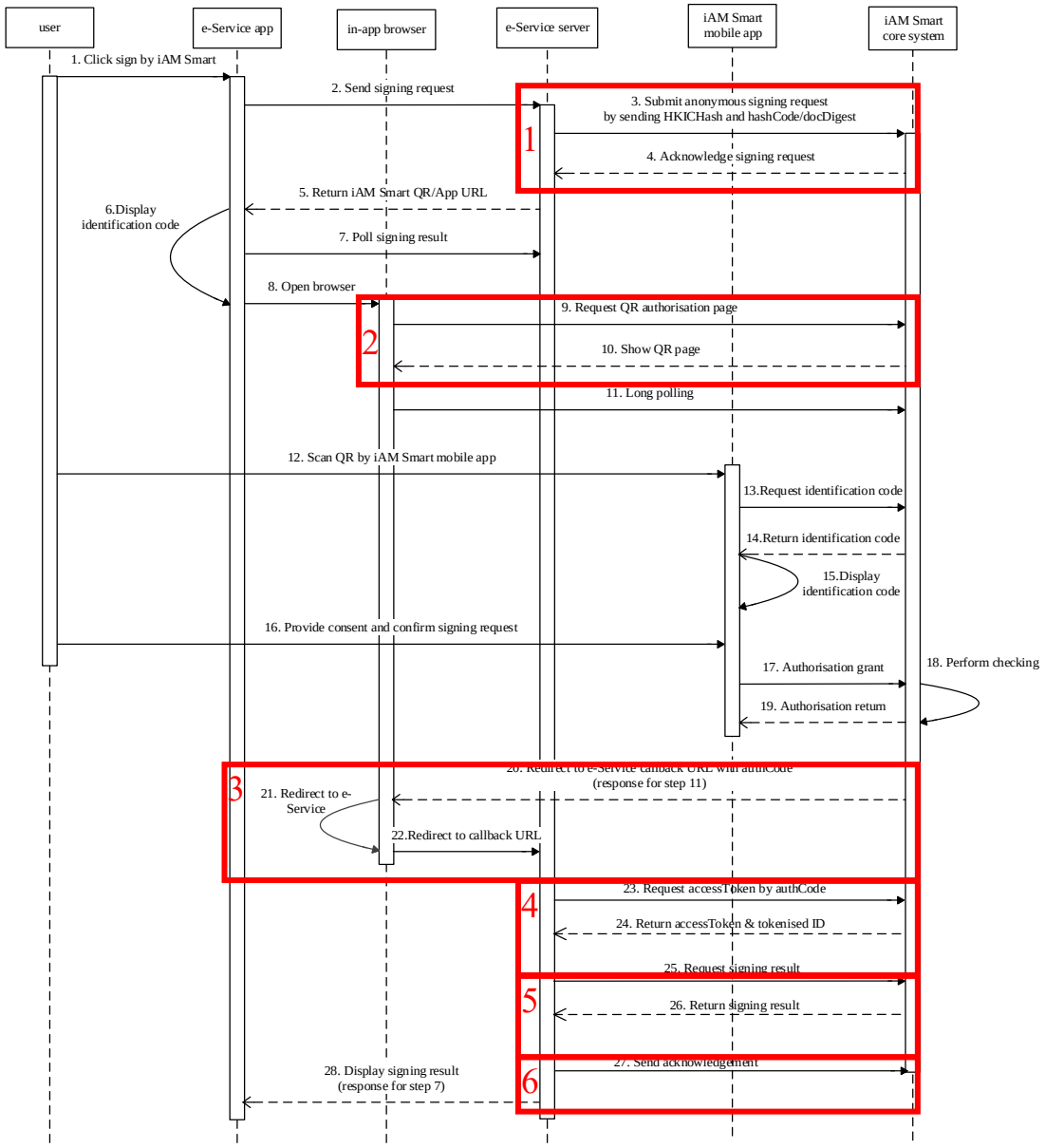

Figure-24 Anonymous Digital Signing (Online Service App in Different Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Digital Signing	7.5.1
	Request Anonymous PDF Digital Signing	7.5.2
2	Request QR Page	7.5.3
3	Callback with authCode to Online Service Server	7.5.6
4	Request accessToken & Tokenised ID	7.5.7
5	Obtain Anonymous Digital Signing Result	7.5.8
	Obtain Anonymous PDF Digital Signing Result	7.5.9
6	Online Service Acknowledges Digital Signing Result	7.5.10

7.4.4 Anonymous Digital Signing (Online Service App in Same Device)

The sequence diagram below shows how an anonymous user authorises and signs the document hash when online service App and the "iAM Smart" Mobile App are running in the same device.

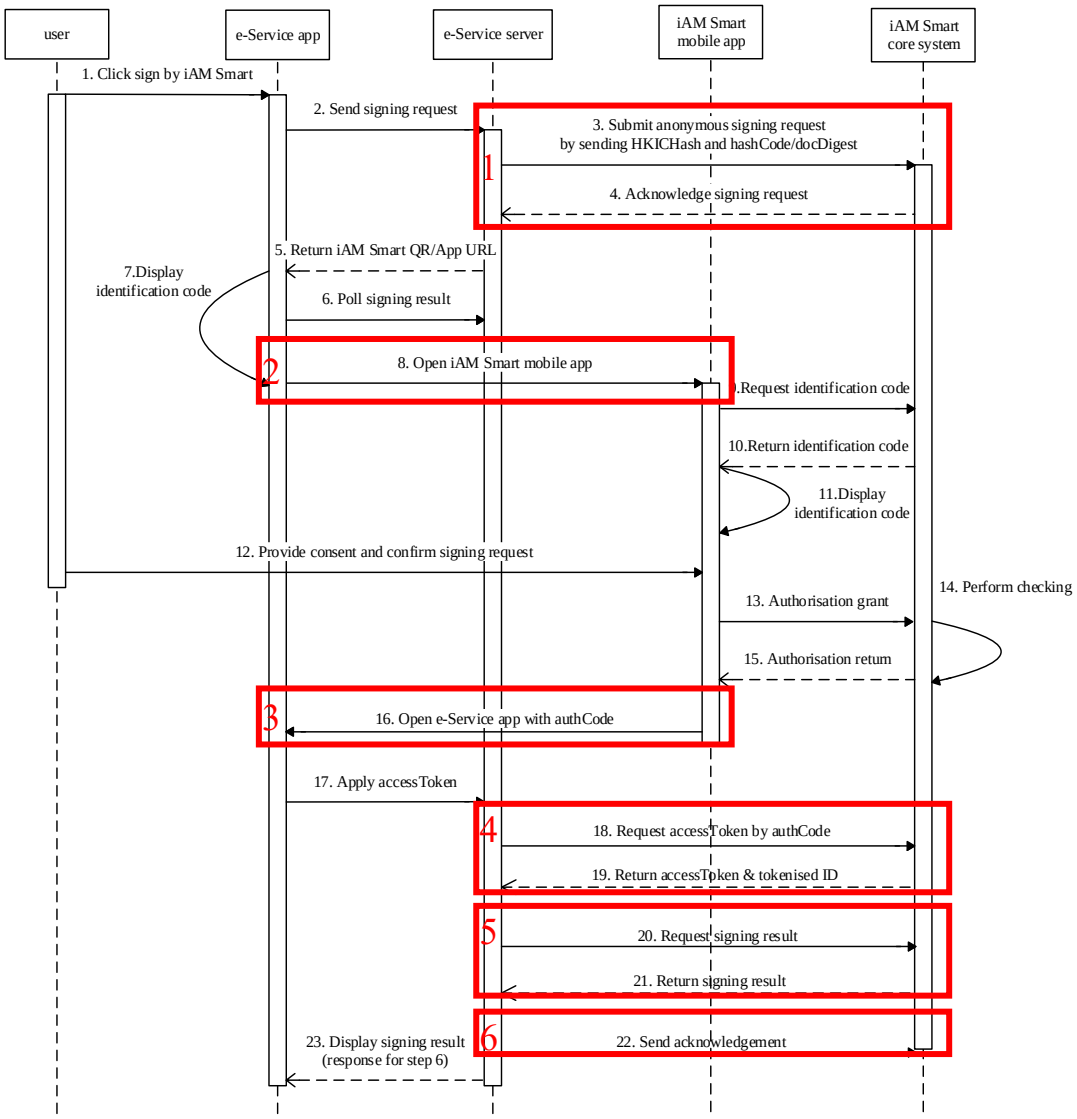

Figure-25 Anonymous Digital Signing (Online Service App in Same Device)

APIs interactions between online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Digital Signing	7.5.1
	Request Anonymous PDF Digital Signing	7.5.2

2	Open “iAM Smart” Mobile App for Anonymous Digital Signing (appv1)	7.5.4
	Open “iAM Smart” Mobile App for Anonymous Digital Signing (appv2)	7.5.11
3	Callback with authCode to Online Service App	7.5.5
4	Request accessToken & Tokenised ID	7.5.7
5	Obtain Anonymous Digital Signing Result	7.5.8
	Obtain Anonymous PDF Digital Signing Result	7.5.9
6	Online Service Acknowledges Digital Signing Result	7.5.10

7.5 API Implementation Details

7.5.1 Request Anonymous Digital Signing

- **API Description**

Name	Description
Service Full Name	Request anonymous digital signing
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/anonymous/signing/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to initiate anonymous digital signing request by sending `hashCode` and `HKICHash` to "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It could be ASCII string with length less than or equal to 36 chars.
`hashCode`	String	Required	the document hash to be signed. Online service should compute the hash and send this hash to "iAM Smart" System for digital signing. The value should be Base64 encoded.
`sigAlgo`	String	Optional	signature algorithm to be used. Online service can specify either `SHA256withRSA` or `NONEwithRSA`. The default value is `SHA256withRSA`. While using `NONEwithRSA`, `hashCode` provided must be hashed with SHA256.
`HKICHash`	String	Required	Signing request will only be processed when the hash of the HKIC number of the "iAM Smart" user is matched with the HKICHash provided by Online Service. Online service should convert the HKIC number into hash value using SHA256 before sending to "iAM Smart" System.

			Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
`department`	String	Optional	The department that initiates the digital signing request. Maximum Length: 100
`serviceName`	String	Required	The online service name. Maximum Length: 255
`documentName`	String	Required	The document name that the user is going to sign. Maximum Length: 255
`suaMethod`	String (JSON)	Optional	`{"unary": ["FR"]}, {"unary": ["NFC"]},{"and": ["FR", "NFC"]}`. Online service should have been granted the requested step-up authentication method(s) in the e-Service Management Portal to use the `"NFC", "FR"` methods to verify the user. The API returns unsuccessful immediately if the first method fails. No matter if this parameter is specified, FIDO will be used to verify the user at the beginning as before.
`suaWaitPeriod`	Integer	Optional	If `suaMethod` is provided, this waiting period defines the minimum number of hours since the user account was created. If system defined waiting period is greater than `suaWaitPeriod`, greater value of system defined waiting period would be used. If `suaWaitPeriod` is greater than system defined waiting period, greater value of `suaWaitPeriod` would be used. System default value is zero. Maximum Value: 720

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/anonymous/signing/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "hashCode": " 965e638fda4c70f667efc2d68c40c6111e5965bfc82356d",
   "HKICHash": " 6ea37e641260714009a880a8057032a414009a80f667efc",
   "department": " Digital Policy Office",
   "serviceName": "Online Service1",
   "documentName": "Doc0001",
   "suaMethod": "{\"unary\": [\"FR\"]}"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier that will be used to identify this request in the following steps of the digital signing workflow. The ticketID will be expired 12 minutes after issuance.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
```

```
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "ticketID": "561b6f1ce5a043ba8aae33fb5c50386d"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {hashCode}"
}
```

7.5.2 Request Anonymous PDF Digital Signing

- **API Description**

Name	Description
Service Full Name	Request anonymous PDF digital signing
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/anonymous/pdfsigning/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to initiate anonymous PDF digital signing request by sending `docDigest` and `HKICHash` to "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It could be ASCII string with length less than or equal to 36 chars.
`docDigest`	String	Required	the PDF document digest to be signed. Online service should compute the digest using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter. The value should be Base64 encoded.
`HKICHash`	String	Required	Signing request will only be processed when the hash of the HKIC number of the "iAM Smart" user is matched with the HKICHash provided by Online Service. Online service should convert the HKIC number into hash value using SHA256 before sending to "iAM Smart" System. Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
`department`	String	Optional	The department that initiates the digital signing request. Maximum Length: 100

serviceName	String	Required	The online service name. Maximum Length: 255
documentName	String	Required	The document name that the user is going to sign. Maximum Length: 255
suaMethod	String (JSON)	Optional	{"unary": ["FR"]}, {"unary": ["NFC"]},{"and": ["FR", "NFC"]}. Online service should have been granted the requested step-up authentication method(s) in the e-Service Management Portal to use the "NFC", "FR" methods to verify the user. The API returns unsuccessful immediately if the first method fails. No matter if this parameter is specified, FIDO will be used to verify the user at the beginning as before.
suaWaitPeriod	Integer	Optional	If suaMethod is provided, this waiting period defines the minimum number of hours since the user account was created. If system defined waiting period is greater than suaWaitPeriod, greater value of system defined waiting period would be used. If suaWaitPeriod is greater than system defined waiting period, greater value of suaWaitPeriod would be used. System default value is zero. Maximum Value: 720

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/anonymous/pdfsigning/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
```

```
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "docDigest": " 965e638fda4c70f667efc2d68c40c6111e5965bfc82356d",
   "HKICHash": " 6ea37e641260714009a880a8057032a414009a80f667efc",
   "department": "Digital Policy Office",
   "serviceName": "Online Service1",
   "documentName": "Doc0001",
   "suaMethod": "{\"unary\": [\"FR\"]}"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier that will be used to identify this request in the following steps of the digital signing workflow. The ticketID will be expired 12 minutes after issuance.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "ticketID": "561b6f1ce5a043ba8aae33fb5c50386d"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
```

```
  "code": "D20002",
  "message": "empty parameter {hashCode}"
}
```

7.5.3 Request QR Page

- **API Description**

Name	Description
Service Full Name	Request QR Page
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/auth/getQR`
Request Type	GET
Service Version	1.0.0
Description of Service	Online service calls this API to get QR/App broker page or QR page. After user authorises anonymous request on the "iAM Smart" Mobile App, the page will be redirected to the `redirectURI` with `authCode` and `state` parameters. If user denies, only the `state` parameter will be redirected.

- **Request Parameters**

Parameter	Type	Presence	Description
`clientID`	String	Required	Online service client identifier. The clientID will be assigned to online service at the initial registration.
`responseType`	String	Required	value MUST be set to `code`.
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.
`redirectURI`	String	Required	Callback redirect URI. The value should be URL encoded and registered in self-service portal.
`scope`	String	Required	The value should be URL encoded. Specify the scope of "iAM Smart" functions: - `eidapi_sign` The value of the scope parameter is expressed as a list of space-delimited, case-sensitive strings.

`ticketID`	String	Required (Conditional)	Required in anonymous anonymous digital signing workflows for getting QR page.
`lang`	String	Optional	Language to display: en-US, zh-HK, or zh-CN. If this parameter is not specified, zh-HK will be shown.
`state`	String	Optional	If `state` parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
`brokerPage`	Boolean	Optional	If `brokerPage` is set to `true`, Universal Link (iOS) / App Link (Android) will be leveraged to open "iAM Smart" Mobile App. This feature is useful for online service supporting mobile web version while trigger "iAM Smart" Mobile App or showing QR page automatically. (i.e. Show QR page without detecting whether "iAM Smart" Mobile App is installed). The default value is `false`.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
GET
https://<iAM_Smart_domain>/api/v1/auth/getQR?clientID=Online Service1
&responseType=code
&source=Android_Chrome
&redirectURI=https%3A%2F%2Frp_domain%2Frp_context%2Fcall_back_endpoint
&scope=eidapi_sign
&lang=en-US
&state=eb9b7b8eddd5
```

- **Response Parameters**

N/A

7.5.4 Open "iAM Smart" Mobile App for Anonymous Digital Signing (appv1)

- **Using URL Scheme**

Name	Description
Service Full Name	Open "iAM Smart" Mobile App for Anonymous Digital Signing

URI (as in RESTFUL API)	`<"iAM Smart" app URL scheme>://anon_hash-sign`
	or
	`<"iAM Smart" app URL scheme>://anon_pdf-sign`

Service Version	V1.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific Digital Signing in "iAM Smart" Mobile App. The URL schemes supported on iOS and Android versions of "iAM Smart" Mobile App. *Remark: appv1 does not allow one client id to support multiple app.*

- **Request Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required	`TicketID` is a unique identifier provided by "iAM Smart" System, online service retrieve `ticketID` while requesting the Anonymous Digital Signing function. It is ASCII string with length less than or equal 36 chars.
`source`	String	Required (Conditional)	App_Link/ App_Scheme Please use App_Link unless the support team approved.
`redirectURI`	String	Required (Conditional)	Callback redirect URI. The value should be URL encoded and registered in self-service portal.
`state`	String	Required (Conditional)	If `state` parameter is presented in the request message, the same state value will be returned to online service during

			callback. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://anon_hash-sign
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

7.5.5 Callback with authCode to Online Service App

- **URL Scheme and Package Name**

Name	Description
Service Full Name	Callback with authCode to Online Service App
URL Scheme	`<Universal / App Link>`. The online service custom app scheme or Universal / App Link should be registered with the "iAM Smart" System during onboarding.
Package name (Android only)	`<package name>` and `<activity name>`. The online service package name must be registered in the self-service portal. *Remark: Direct Login v2 (App) and appv2 API adopts package name verification instead of App Link*
Description of Service	For appv1 API, the Online Service App will be invoked and launched by Universal / App Link. It makes use of deep linking to redirect users to the Online Services app. App Link can only work for mobile devices with Google Mobile Services (GMS). For appv2 API, the Android Online Service App will be invoked and launched by the package name. It make use of intent to redirect users to the Online Service app. The iOS Online Service App will be invoked and launched via Universal link. The Universal / App Link with the landing location as well as the

	package name plus activity name must be registered in the self-service portal and enabled by the support team.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	12 minutes	Online service could treat the request as failed if it doesn't get callback within 12 minutes.

- **URL Scheme Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required (Conditional)	`businessID` will only be returned in anonymous workflows. `businessID` is a unique identifier for online service to differentiate different request. Online service can use the `businessID` to relate the callback message with the original request.
`code`	String	Required (Conditional)	The authorisation code generated by the "iAM Smart" System. The authorisation code will be expired in 1 minute after issuance. Online service MUST NOT use the authorisation code more than once. An error message will be returned if an authorisation code is expired or re-used.
`state`	String	Optional	If the `state` parameter has been presented in the request message, the same state value will be returned. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random value.
`error_code`	String	Required (Conditional)	Return error code when exception.

- **Example URL Scheme**

Allow

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
```

```
?businessID=b2c99aa83b0049e9ba370c5341681225
&code=0ad186353c424c64897fcc00445c9ba1
&state=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
?error_code=D20001
&state=eddd527b6
```

- **Example Package Name**

Allow

```
Intent ii=new Intent(<package name>, <activity class name>);
ii.putExtra("code", "0ad186353c424c64897fcc00445c9ba1");
ii.putExtra("state", "eddd527b6");
ii.putExtra("businessID", "b2c99aa83b0049e9ba370c5341681225");
startActivity(ii);
```

Deny

```
Intent ii=new Intent(<package name>, <activity name>);
ii.putExtra("error_code", "D20001");
ii.putExtra("state", "eddd527b6");
startActivity(ii);
```

7.5.6 Callback with authCode to Online Service Server

- **API Description**

Name	Description
Service Full Name	Callback with authCode to Online Service Server
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	GET
Service Version	V1.0.0
Description of Service	This callback is used to pass `authCode` to online service Server. The URI must be registered in the self-service portal.

- **API Specific Timeout**

Title	Timeout value	Description

Callback	12 minutes	Online service could treat the request as failed if it doesn't get callback within 12 minutes.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required (Conditional)	`businessID` will only be returned in anonymous workflows. `businessID` is a unique identifier for online service to differentiate different requests. Online service can use the `businessID` to relate the callback message with the original request.
`code`	String	Required (Conditional)	The authorisation code is generated by the "iAM Smart" System. The authorisation code will be expired 1 minute after issuance. Online service MUST NOT use the authorisation code more than once. An error message will be returned if an authorisation code is expired or re-used.
`state`	String	Optional	If the `state` parameter has been presented in the request, the same state value will be returned during the callback. It is used to prevent the CSRF attack. The value of the state is defined by Online Service, and it should be a secure random value.
`error_code`	String	Required (Conditional)	Return error code when exception.

- **Example Callback**

Allow

```
// Line breaks are for legibility only.
GET
https://<call_back_endpoint>
?code=0ad186353c424c64897fcc00445c9ba1
&state=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
GET
https://<call_back_endpoint>
?error_code=D20001
&state=eddd527b6
```

7.5.7 Request accessToken & Tokenised ID

- **API Description**

Name	Description
Service Full Name	Request access token and tokenised ID with authCode
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/auth/getToken`
Request Type	POST
Service Version	1.0.0
Description of Service	Online service uses this API to retrieve the access token and Tokenised ID (openID). An authorisation code is necessary during the process. The `accessToken` and `openID` will be used to call corresponding "iAM Smart" services subsequently.

- **Request Parameters**

Parameter	Type	Presence	Description
`code`	String	Required	The authorisation code is received from the authorisation server. One time use only and will be expired in 1 minute.
`grantType`	String	Required	the value MUST be set to `authorization_code`.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/auth/getToken
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
```

```
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
  "code": "xxxa42e76bf4cb0846a68e6d83d6096",
  "grantType": "authorization_code"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
accessToken	String	Required	accessToken value can only be used **once**.
tokenType	String	Required	Token type, support "Bearer" only
issueAt	Long	Required	The accessToken issue time is expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
expiresIn	Long	Required	The lifetime in milliseconds of the token. The value may vary for different Online Services.
openID	String	Required	Tokenised ID, uniquely generated for each user of each online service website or mobile application.
lastModifiedDate	Long	Required	The datetime of the user complete registration at the "iAM Smart" System. The value will be updated when either of the following is valid:- (1) If any one of the following verified data are changed. English name Chinese name (* not applicable if it was marked as unverified during registration) Gender Date of birth (2) User re-register "iAM Smart" after "iAM Smart" de-registration.

			The modification time will be expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
`userType`	String	Required	`default` or `sign` `default`: "iAM Smart" user `sign`: "iAM Smart+" user (digital signing capability)
`scope`	String	Required	The scope of the token. Please refer to the corresponding section specified in each API function.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "accessToken": "0ad186353c424c64897fcc00445c9ba1",
    "tokenType": "Bearer",
    "issueAt": 1557053922938,
    "expiresIn": 14400000,
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
    "lastModifiedDate": 1560849218006,
    "userType": "sign",
    "scope": "eidapi_sign"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D40004",
  "message": "authCode not exist or expired",
}
```

7.5.8 Obtain Anonymous Digital Signing Result

- **API Description**

Name	Description
Service Full Name	Obtain anonymous digital signing result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/anonymous/signing/getResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to retrieve anonymous digital signing result.

- **Request Parameters**

Parameter	Type	Presence	Description
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/anonymous/signing/getResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
    "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
hashCode	String	Optional	The document hash submitted by online service in the API request.
timestamp	Long	Optional	Timestamp in milliseconds since January 1, 1970 00:00:00 GMT. "iAM Smart" System will provide this to online service only when digital signing is successful.
signature	String	Optional	Base64-encoded signature result string. "iAM Smart" System will provide this to online service only when digital signing is successful.
cert	String	Optional	Base64-encoded DER format certificate for the "iAM Smart" user. "iAM Smart" System will provide this to online service only when the digital signing is successful.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
            "hashCode": "tGzv3JOkF0XG5Qx2TlKWIA",
            "timestamp": 1556450176000,
            "signature": "nnoadisauflanefhykdjf",
            "cert": "sdfGSDGsdfaGDEHfjslgGQG……GSGjljlkjwmh",
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
```

```
  "message": "empty parameter {accessToken}"
}
```

7.5.9 Obtain Anonymous PDF Digital Signing Result

- **API Description**

Name	Description
Service Full Name	Obtain anonymous PDF digital signing result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/anonymous/pdfsigning/getResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to retrieve anonymous PDF digital signing result.

- **Request Parameters**

Parameter	Type	Presence	Description
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/anonymous/signing/getResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
    "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
```

```
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
docDigest	String	Optional	The pdf document digest submitted by online service in the API request.
pdfSignature	String	Optional	Base64-encoded PKCS#7 object that is the actual PDF signature value. It contains the signer's certificate, signed hash value, and the digital signing timestamp information. Online service can embed this value to the PDF document for future verification. "iAM Smart" System will provide this to online service only when the digital signing is successful.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// Decrypted Body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
            "docDigest": "tGzv3JOkF0XG5Qx2TlKWIA",
            "pdfSignature": "nnoadisauflanefhykdjf",
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {accessToken}"
}
```

7.5.10 Online Service Acknowledges Digital Signing Result

- **API Description**

Name	Description
Service Full Name	Online Service acknowledges digital signing result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/account/signing/ackResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service calls this API to acknowledge "iAM Smart" System if the result of the digital signature is accepted or not.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It should be ASCII string with a length of less than or equal to 36 chars.
`signingResult`	String	Required	"SR001": the digital signature is accepted "SR002": the digital signature is rejected "SR003": no digital signature was received

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/account/signing/ackResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
```

```
{
  "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
  "signingResult": "SR001",
}
```

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS"
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {signingResult}"
}
```

7.5.11 Open "iAM Smart" Mobile App for Anonymous Digital Signing (appv2)

- **Using URL Scheme**

Name	Description
Service Full Name	Open "iAM Smart" Mobile App for Anonymous Digital Signing
URI (as in RESTFUL API)	<"iAM Smart" app URL scheme>://v2_anon_hash-sign or <"iAM Smart" app URL scheme>://v2_anon_pdf-sign
Service Version	V2.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific Digital Signing in "iAM Smart" Mobile App. The URL schemes supported on iOS and Android versions of "iAM Smart" Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
ticketID	String	Required	TicketID is a unique identifier provided by "iAM Smart" System, online service retrieve ticketID while requesting the Anonymous Digital Signing function. It is ASCII string with length less than or equal 36 chars.
source	String	Required	App_Link (iOS) or App_Package (Android)
clientRedirectURI	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (",") are not accepted.
packageName	String	Required (Android only)	Package name of the Online Service App for callback use.
activityClass	String	Required (Android only)	Activity class name of the Online Service App for callback use.

`activityParams`	String	Optional (Android only)	Optional params used during callback.
`state`	String	Required (Conditional)	If `state` parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://v2_anon_hash-sign
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
&source=App_Package
&packageName=com.onlineservice.myapp
&activityClass=callback_activity
&activityParams=callback_param
```

8. RE-AUTHENTICATION WITH SERVICE LOGIN

8.1 Overview

Online service can request "Re-authentication" API to to ask the same "iAM Smart" user to authenticate again with his/her digital identity as long as user has logged in to online service via "iAM Smart" and online service still holds a valid accessToken. "Re-authentication" API is provided to the online service that required user confirmation in "iAM Smart" App.

8.2 Prerequisite

- Online service must undergo the authentication process (i.e. Request accessToken & Tokenised ID) in Section 3 in order to obtain the accessToken and openID as the input of Re-Authentication API.
- Online service shall refer to "Reference System Flow and User Interface Specifications for "iAM Smart" Adoption" for the detail UI/UX requirements.

8.3 Scope

Online service shall submit the application for using the following scope values to "iAM Smart" Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_fr`	Scope for Re-authentication

8.4 Use Cases and Scenarios

8.4.1 Re-authentication (Online Service Website/App in Different Device)

The sequence diagram below shows how online service performs "iAM Smart" Re-authentication when online service and the "iAM Smart" Mobile App are running in different devices.

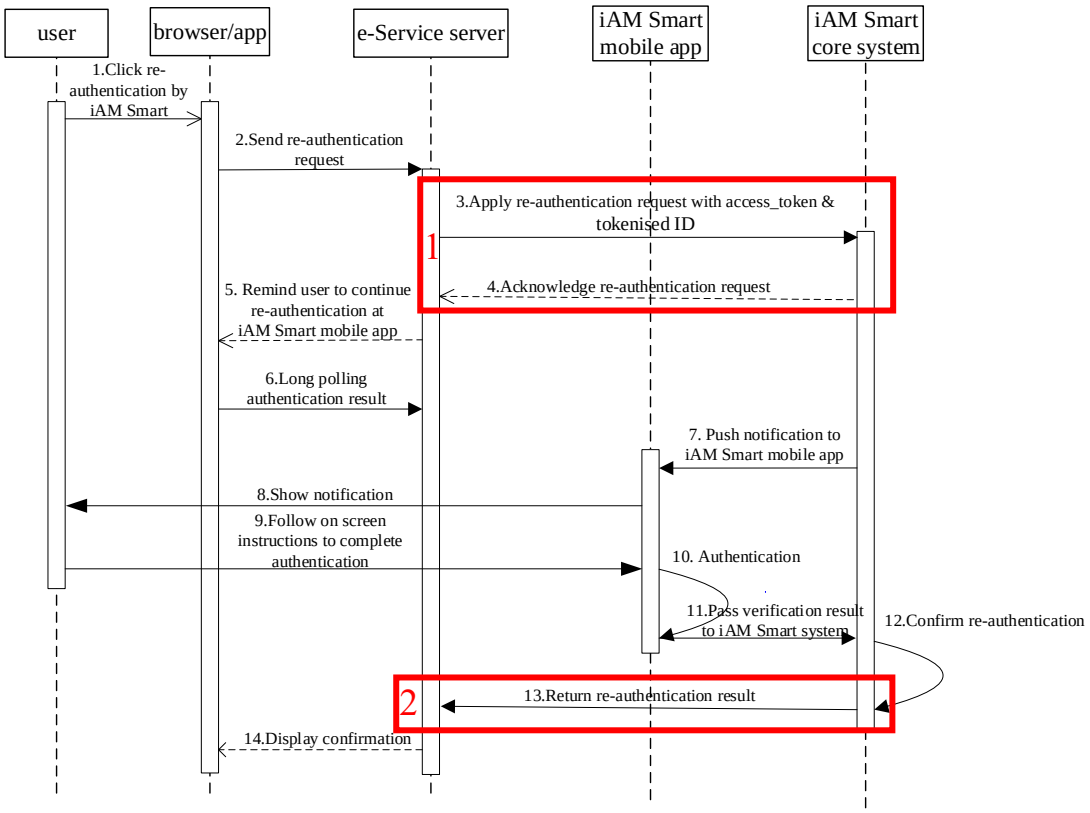

Figure-26 Re-authentication (Online Service Website/App in Different Device)

APIs interactions between Online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Request Re-authentication (appv1)	8.5.1
2	Callback to Receive Re-authentication Result	8.5.3

8.4.2 Re-authentication (Online Service Website in Same Device)

The sequence diagram below shows how online service performs “iAM Smart” Re-authentication when the online service website and the “iAM Smart” Mobile App are running in the same device.

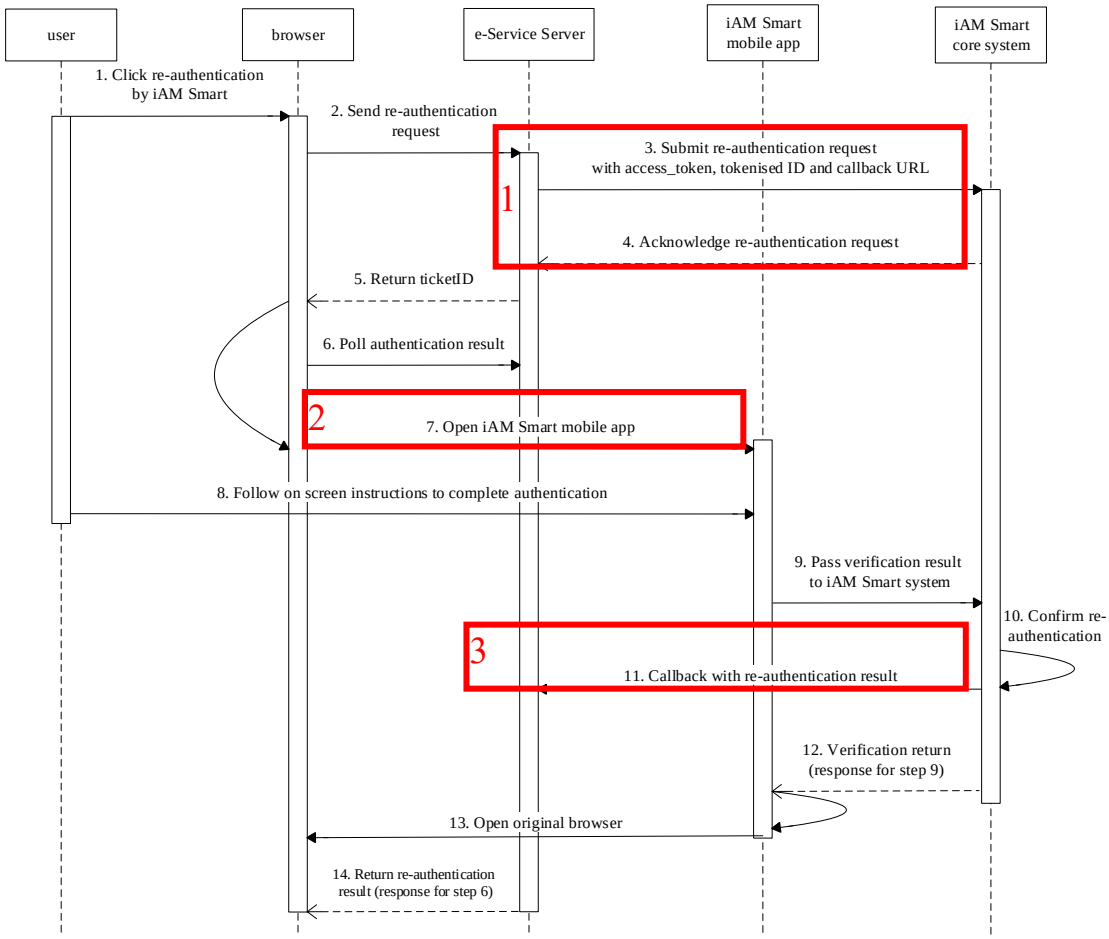

Figure-27 Re-authentication (Online Service Website in Same Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Request Re-authentication (appv1)	8.5.1
2	Open the “iAM Smart” Mobile App for Re-authentication	8.5.2
3	Callback to Receive Re-authentication Result	8.5.3

8.4.3 Re-authentication (Online Service App in Same Device)

The sequence diagram below shows how online service performs "iAM Smart" Re-authentication when the online service mobile application and the "iAM Smart" Mobile App are running in the same device.

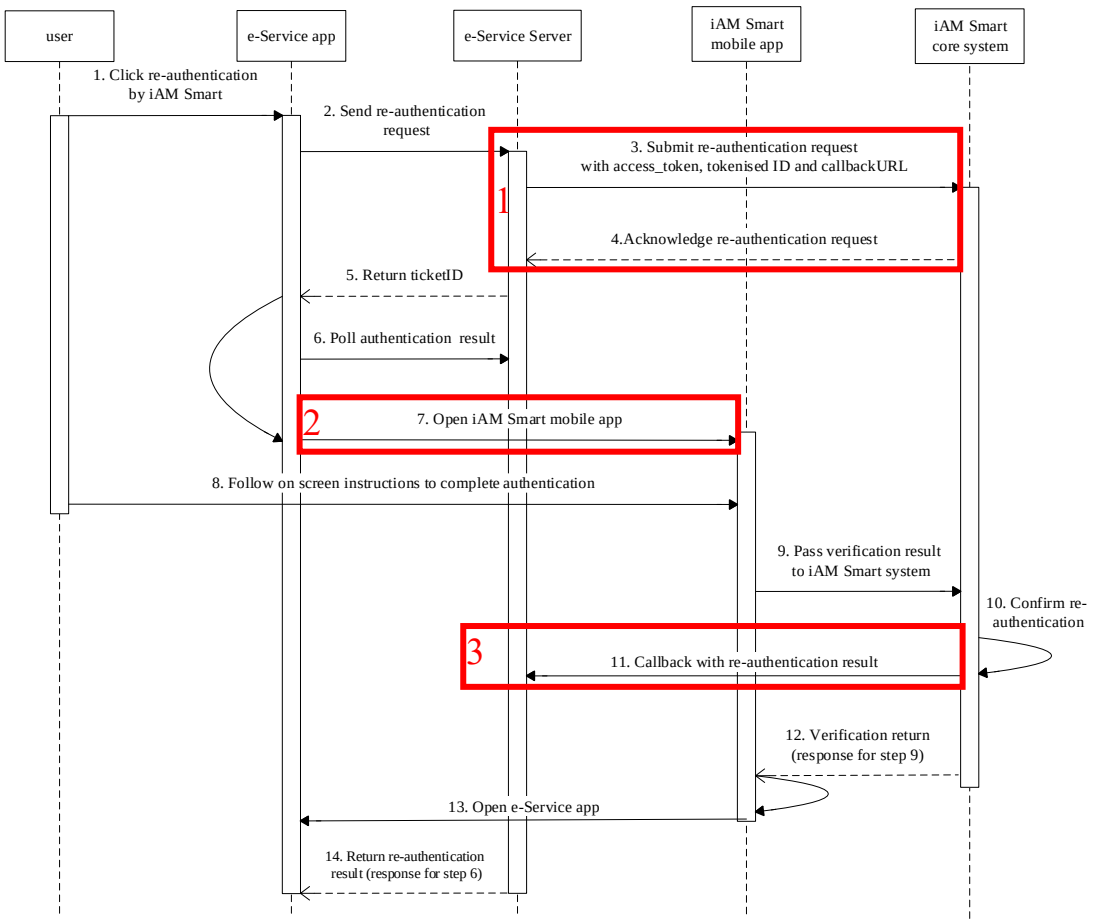

Figure-28 Re-authentication (Online Service App on Same Device)

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Request Re-authentication (appv1)	8.5.1
	Request Re-authentication (appv2)	8.5.4
2	Open the "iAM Smart" Mobile App for Re-authentication	8.5.2
3	Callback to Receive Re-authentication Result	8.5.3

8.5 API Implementation Details

8.5.1 Request Re-authentication (appv1)

- **API Description**

Name	Description
Service Full Name	Request Re-authentication
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/account/stepup/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to request re-authentication by sending `accessToken` and `openID` to the "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different requests. It should be an ASCII string with a length of less than or equal to 36 chars.
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value Maximum Length: 64
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.
`redirectURI`	String	Required	Callback URI.
`state`	String	Optional	If the `state` parameter is presented in the request message, the same state value will be returned to online service during the callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers,

			underscore and hyphens are accepted.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/account/stepup/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
 "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
 "source": "Android_Chrome",
 "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
 "state": "eddd527b6"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If the user completes authentication by scanning the QR code (i.e. different device), then "true" will be returned to online service.
`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger the “iAM Smart” Mobile App when authByQR is `false` The value valid for 18 minutes.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
```

```
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {openID}"
}
```

8.5.2 Open the “iAM Smart” Mobile App for Re-authentication

- **Using URL Scheme**

Name	Description
Service Full Name	Open the “iAM Smart” Mobile App for Re-authentication
URI (as in RESTFUL API)	For Re-authentication v1: <“iAM Smart” app URL scheme>://re-auth For Re-authentication v2: <“iAM Smart” app URL scheme>:v2_re-auth
Service Version	V1.0.0 and V2.0.0
Description of Service	The URL scheme uses deep linking to redirect users to re-authentication in the “iAM Smart” Mobile App. The URL schemes are supported on iOS and Android versions of the “iAM Smart” Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
ticketID	String	Required	TicketID is a unique identifier provided by the “iAM Smart” System. Online service retrieves ticketID while requesting the Re-authentication function. It is an ASCII string with a length of less than or equal to 36 chars.

- **Example Scheme**

```
// Line breaks are for legibility only.
<“iAM Smart” app URL scheme>://v2_re-auth
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

8.5.3 Callback to Receive Re-authentication Result

- **API Description**

Name	Description
Service Full Name	Callback to Receive Re-authentication Result
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	V1.0.0
Description of Service	This callback returns the Re-authentication result to online service upon user consent. Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different requests. Online services can use the `businessID` to relate the callback message with the original request. Maximum Length: 36
`state`	String	Optional	The same state value will be returned if the state parameter is in the request message. It is used to prevent the CSRF attack. The value of the state is defined by Online Service, and it should be a secure random value. Maximum Length: 36
`isPassed`	String	Required	Re-authentication result. `true` - same person, otherwise `false`.

- **Example Callback**

```
// Line breaks are for legibility only.
```

```
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>
// Callback Body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
          "content": {
           "businessID": "3e47be25-66a6-43fb-89f6-7e2dd138aff8",
            "state": "unesidkd",
            "isPassed": true
          }
}
```

8.5.4 Request Re-authentication (appv2)

- **API Description**

Name	Description
Service Full Name	Request Re-authentication
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/appv2/account/stepup/initiateRequest`
Request Type	POST
Service Version	V2.0.0
Description of Service	Online service can use this API to request re-authentication by sending `accessToken` and `openID` to the "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different requests. It should be an ASCII string with a length of less than or equal to 36 chars.
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value Maximum Length: 64
`source`	String	Required	App_Link (iOS) or App_Package (Android)
`clientRedirectURI`	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (",") are not accepted.
`packageName`	String	Required (Android only)	Package name of the Online Service App for callback use.
`activityClass`	String	Required (Android only)	Activity class name of the Online Service App for callback use.
`activityParams`	String	Optional (Android only)	Optional params used during callback.

serverRedirectURI	String	Required	Callback URI.
state	String	Optional	If the state parameter is presented in the request message, the same state value will be returned to online service during the callback. It is used to prevent the CSRF attack. The value of state is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/appv2/account/stepup/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
 "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
 "source": "App_Package",
 "packageName": "com.onlineservice.myapp",
 "activityClass": "callback_activity",
 "activityParams": "callback_param",
 "serverRedirectURI":
"https://<rp_domain>/<rp_context>/<call_back_endpoint>",
 "state": "eddd527b6"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If the user completes authentication by scanning the QR code (i.e. different device), then "true" will be returned to online service. `authByQR` should always be `false` in appv2 API.
`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger the “iAM Smart” Mobile App when authByQR is `false` The value valid for 18 minutes.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": false,
    "ticketID": "bbb8aae57c104cda40c93843ad5e6db8"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {openID}"
}
```

9. BULK DIGITAL SIGNING WITH SERVICE LOGIN

9.1 Overview

"iAM Smart+" version supports bulk digital signing with legal backing under the Electronic Transactions Ordinance (Cap. 553) for handling statutory documents and procedures.
Online Services can make use of Bulk Digital Signing API to enable "iAM Smart" users complete digital signing online for multiple documents with only one digital signing cycle. It can be used in many cases, such as bulk digital signing online application form and bulk digital signing contract and agreement.

9.2 Prerequisite

- Online service must undergo the authentication process (i.e. Request accessToken & Tokenised ID) in Section 3.4.5 in order to obtain the accessToken and openID as the input of Bulk Digital Signing API.
- To request the Bulk Digital Signing API, the "iAM Smart" user shall be a "iAM Smart+" user. Online service shall prompt the message in their website / mobile app and guide the user to upgrade his/her user account if he/she is not the "iAM Smart+" user.
- Online service shall refer to "Reference System Flow and User Interface Specifications for "iAM Smart" Adoption" for the detail UI/UX requirements.

9.3 Scope

Online service shall submit the application for using the following scope values to "iAM Smart" Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_bulksign`	Scope for Bulk Digital Signing

9.4 Use Cases and Scenarios

9.4.1 Bulk Digital Signing (Online Service Website/App in Different Device)

The sequence diagram below shows how an authenticated user authorises and signs multiple document hashes and/or digests when Online Service website/app and the "iAM Smart" Mobile App are running in different devices.

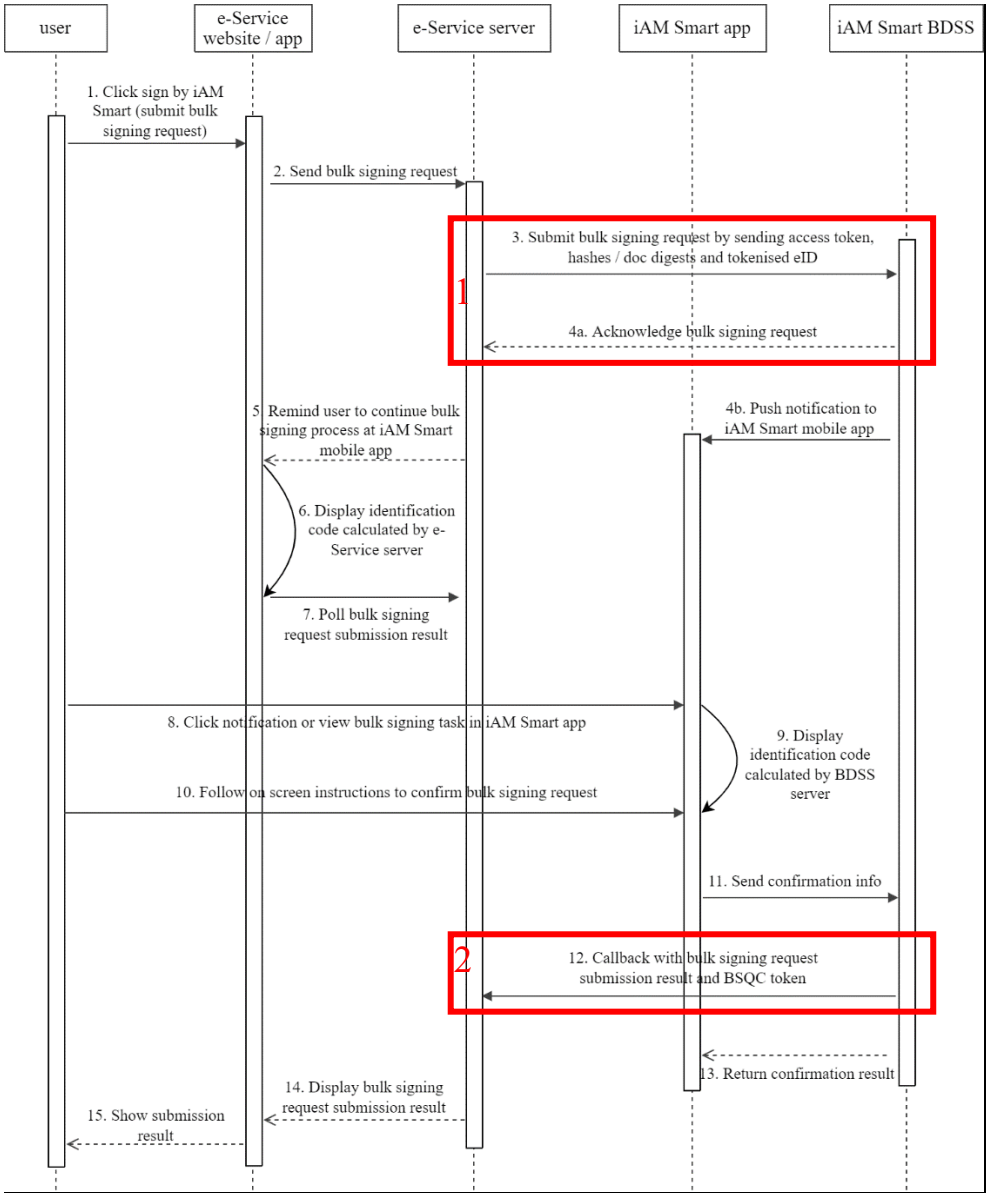

Figure-29 Bulk Digital Signing (Online Service Website/App in Different Device)

APIs interactions between Online Service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Bulk Digital Signing (appv1)	9.5.1
2	Callback to Receive BSQC Token	9.5.3

9.4.2 Bulk Digital Signing (Online Service Website/App in Same Device)

The sequence diagram below shows how an authenticated user authorises and signs multiple document hashes and/or digests when Online Service website/app and the "iAM Smart" Mobile App are running in the same device.

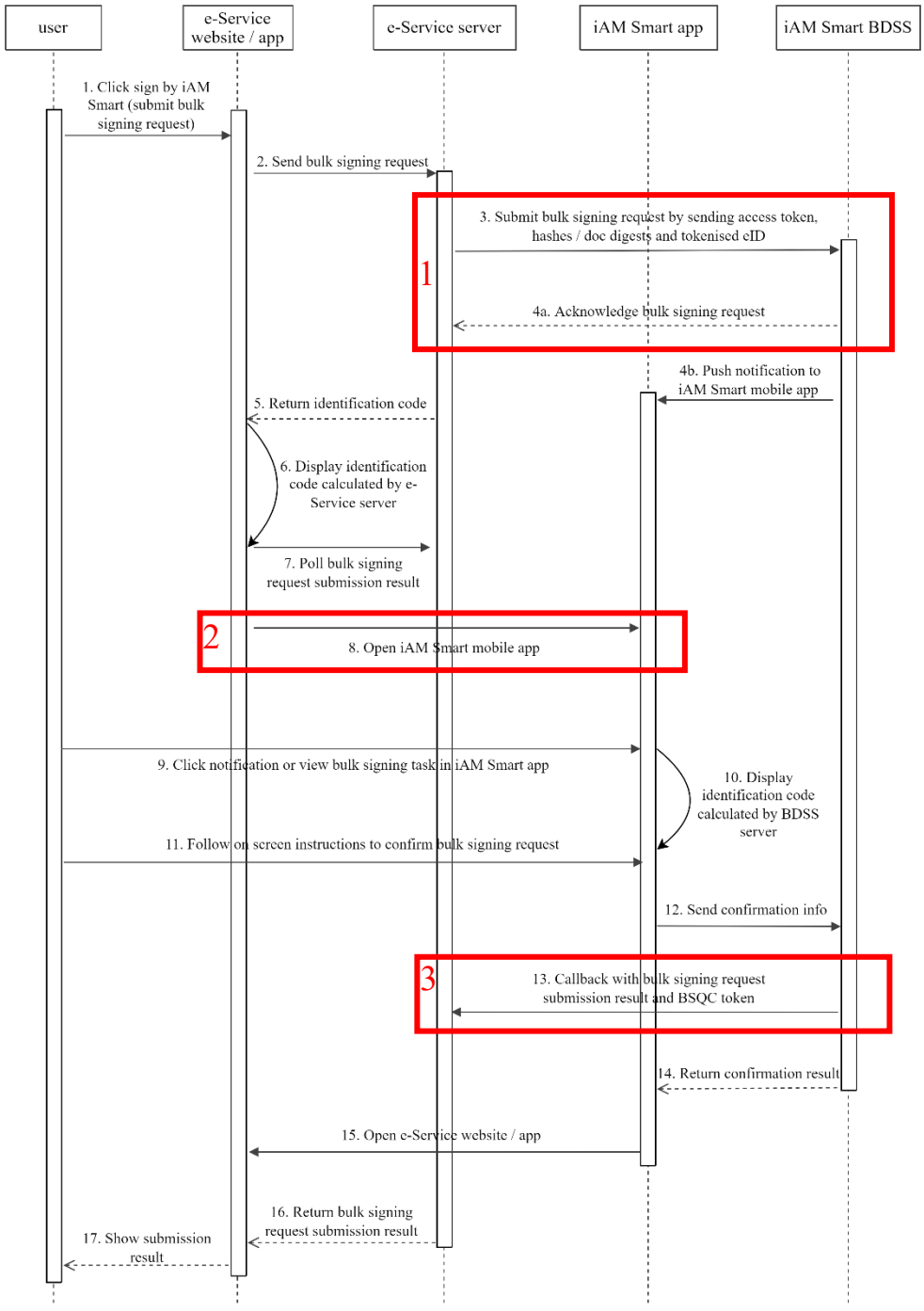

Figure-30 Bulk Digital Signing (Online Service Website/App in Same Device)

APIs interactions between Online Service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	*API Name*	*API Reference (Section)*
1	Request Bulk Digital Signing (appv1)	9.5.1
	Request Bulk Digital Signing (appv2)	9.5.8
2	Open “iAM Smart” Mobile App for Bulk Digital Signing	9.5.2
3	Callback to Receive BSQC Token	9.5.3

9.4.3 Bulk Digital Signing Result Callback

The sequence diagram below shows how the bulk digital signing result is sent to Online Service's callback endpoint after the digital signing process is completed.

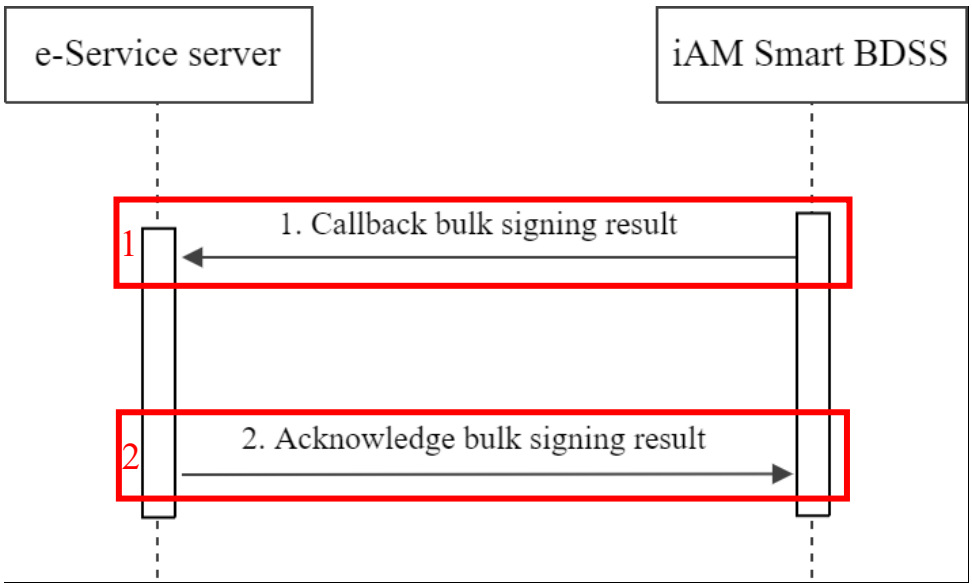

Figure-31 Bulk Digital Signing Callback

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Callback to Receive Bulk Digital Signing Result	9.5.4
2	Online Service Acknowledges Bulk Digital Signing Result	9.5.5

9.4.4 Enquire Bulk Digital Signing Result

The sequence diagram below shows how Online Service queries the status of the submitted digital signing request.

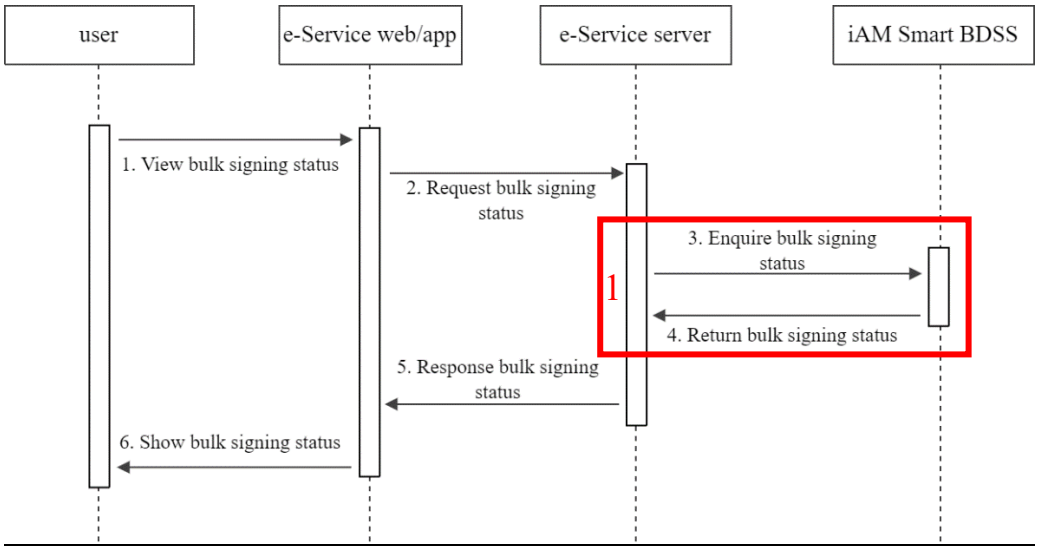

Figure-32 Bulk Digital Signing Status Enquiry

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Enquire Bulk Digital Signing Status	9.5.6

9.4.5 Cancel Bulk Digital Signing Request

The sequence diagram below shows how Online Service cancels the submitted digital signing request.

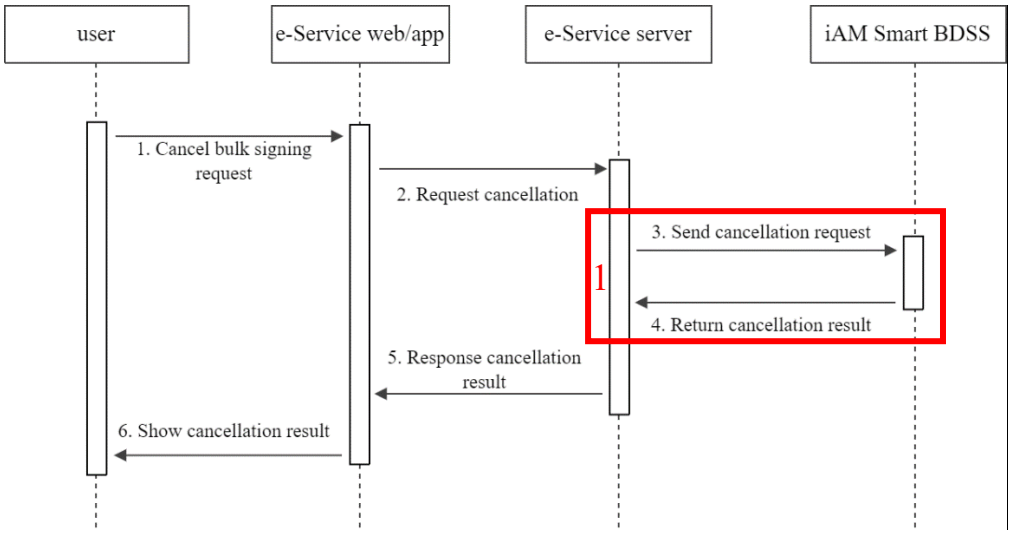

Figure-33 Bulk Digital Signing Cancellation

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Cancel Bulk Digital Signing Request	9.5.7

9.5 API Implementation Details

9.5.1 Request Bulk Digital Signing (appv1)

- **API Description**

Name	Description
Service Full Name	Request Bulk Digital Signing
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/bdss/account/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online Service can use this API to make request for bulk digital signing by sending `accessToken` and `openID` to "iAM Smart" System.

- **Request Parameters**

Online Service can specify the `Content-Type` of this as either `application/json` or `multipart/form-data`. The default `Content-Type` is `application/json`.

- If the request Content-Type is `application/json`, the fields in the following table should be put in the request body in JSON format.
- If the request Content-Type is `multipart/form-data`, the request body should be seperated into two parts by a `$<boundary>`. One part (`name="reqFile"`) is a JSON file that contains the request document hashes or digests, i.e. the `documents` and its sub-fields in the following table. The other part (`name="reqMetada"`) is a JSON string that contains request metadata, i.e. the fields except `documents` and its sub-fields. These two parts should be encrypted with the same CEK. The `$<boundary>` is automatically generated and is different in each request, e.g. `--ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9`.

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for Online Service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value Maximum Length: 64
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of the "iAM

			Smart" API Specification.
`redirectURI`	String	Required	Callback URI for the endpoint that Online Service uses to receive BSQC Token (Section 9.5.3)
`callbackResultURI`	String	Required	Callback URI for the endpoint that Online Service uses to receive bulk digital signing result (Section 9.5.4)
`state`	String	Optional	If `state` parameter is presented in the request message, the same state value will be returned to Online Service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
`HKICHash`	String	Optional	Bulk digital signing request will only be processed when the hash of the HKIC number of the "iAM Smart" user is matched with the HKICHash provided by Online Service. Online Service should convert the HKIC number into hash value using SHA256 before sending to "iAM Smart" System. Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
`department`	String	Optional	The department that initiates the bulk digital signing request. This will be displayed on the consent page of the "iAM Smart" mobile app. Maximum Length: 100
`serviceName`	String	Required	The service name that will be displayed on the consent page of the "iAM Smart" mobile app. Maximum Length: 255
`requestName`	String	Required	The bulk digital signing request name that will be displayed on the consent page of the "iAM Smart" mobile app. Maximum Length: 255

maxCallbackSigs	Integer	Optional	The maximal number of signatures that the Online Service endpoint can accept in one callback for receiving the bulk digital signing result. The default value is 1000.

documents	Array	Required	The document hashes and/or pdf digests that are to be signed by "iAM Smart". The number of array items should be greater than 0 and less than 1001. The array items are in one of the following JSON dictionary formats:
			For document hash
			{
			"id": "egh…cva",
			"documentName": "Document",
			"hashCode": "965…56d",
			"sigAlgo": "SHA256withRSA"
			}
			For PDF document digest
			{
			"id": "egh…cva",
			"documentName": "PDF Document",
			"docDigest": "965…56d"
			}
			The following rows show the detail of each dictionary key.

Details for "documents"			
id	String	Required	The document unique identifier. It should be ASCII string with length less than or equal to 36 chars.
documentName	String	Required	The document name that will be displayed in the "iAM Smart" mobile app when asking user's consent for the bulk digital signing request. Maximum Length: 255
hashCode	String	Required (Conditional)	The document hash to be signed. Online Service should compute the hash and send this hash to "iAM Smart" System for digital signing. The value should be Base64 encoded.

sigAlgo	String	Optional	Signature algorithm to be used. Online Service can specify either SHA256withRSA or NONEwithRSA. The default value is SHA256withRSA. While using NONEwithRSA, hashCode provided must be hashed with SHA256.
docDigest	String	Required (Conditional)	The PDF document digest to be signed. Online Service should compute the digest using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter. The value should be Base64 encoded.

- **Example Request**

The following is the example request when Content-Type is set as application/json.

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/bdss/account/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Request Body
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
   "source": "Android_Chrome",
   "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
   "callbackResultURI": "https://<rp_domain>/<rp_context>/<endpoint>",
   "state": "eddd527b6",
   "HKICHash": "6ea37e641260714009a880a8057032a414009a80f667efc",
   "department": "Digital Policy Office",
   "serviceName": "Online Service 1",
   "requestName": "Bulk Digital Signing for Contracts",
```

```
    "maxCallbackSigs": 200,
    "documents": [
      {
        "id": "aef…cth",
        "documentName": "Document",
        "hashCode": "965e…356d",
        "sigAlgo": " SHA256withRSA"
      },
      ……
      {
        "id": "trx…hnm",
        "documentName": "PDF Document",
        "docDigest": "7rtg…y67d"
      },
    ]
}
```

The following is the example request when `Content-Type` is set as `multipart/form-data`.

```
// Line breaks are for legibility only.
POST
https://<iAM_Smart_domain>/api/v1/bdss/account/initiateRequest

// Request Header
// The $<boundary> looks like --ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9
// It is automatically generated and is different in every callback
Content-Type: multipart/form-data;boundary=$<boundary>
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"

// Request Body
$<boundary>
Content-Disposition: form-data; name="reqFile"; filename="documents.txt"
Content-Type: text/plain
Content-Length: 62892
// The content of the text file looks like the following.
// Line breaks are for legibility only.
 {
    "documents": [
      {
```

```
      "id": "aef…cth",
      "documentName": "Document",
      "hashCode": "965e…356d",
      "sigAlgo": " SHA256withRSA"
    },
    ……
    {
      "id": "trx…hnm",
      "documentName": "PDF Document",
      "docDigest": "7rtg…y67d"
    }
  ]
}

$<boundary>
Content-Disposition: form-data; name="reqMetadata"
Content-Type: application/json;charset=UTF-8
{
  "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
  "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
  "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
  "source": "Android_Chrome",
  "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
  "callbackResultURI": "https://<rp_domain>/<rp_context>/<endpoint>",
  "state": "eddd527b6",
  "HKICHash": "6ea37e641260714009a880a8057032a414009a80f667efc",
  "department": "Digital Policy Office",
  "serviceName": "Online Service1",
  "requestName": " Bulk Digital Signing for Contracts",
  "maxCallbackSigs": 200,
}
$<boundary>
```

- **Response Parameters**

Parameter	Type	Presence	Description
authByQR	Boolean	Required	If user completes authentication by scanning QR code (i.e. different device), then "true" will be returned to Online Service.

`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger “iAM Smart” mobile app when authByQR is `false`.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {documents}"
}
```

9.5.2 Open “iAM Smart” Mobile App for Bulk Digital Signing

- **Using URL Scheme**

Name	Description
Service Full Name	Open “iAM Smart” Mobile App for Bulk Digital Signing

URI (as in RESTFUL API)	For Bulk Digital Signing v1:
	`<“iAM Smart” app URL scheme>://bulk-sign`
	
	For Bulk Digital Signing appv2:
	`<“iAM Smart” app URL scheme>://v2_bulk-sign`

Service Version	V1.0.0 and V2.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific Bulk Digital Signing in “iAM Smart” Mobile App. The URL schemes supported on iOS and Android versions of “iAM Smart” Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required	`TicketID` is a unique identifier provided by "iAM Smart", Online service retrieve `ticketID` while requesting the context. It is ASCII string with length less than or equal 36 chars.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://bulk-sign
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

9.5.3 Callback to Receive BSQC Token

- **API Description**

Name	Description
Service Full Name	Callback to Receive BSQC Token
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	1.0
Description of Service	This callback will provide the BSQC Token to Online Service.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online Service could treat the bulk digital signing request as failed if it doesn't get BSQC Token callback within 18 minutes.
BSQCToken	1440 minutes (1 day)	The BSQC Token used to enquire digital signing status or cancel signing request is valid for 1 day.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for

			Online Service to differentiate different request. Online Service can use the `businessID` to relate the callback message with the original request.
`state`	String	Optional	If the `state` parameter has been present in the request message, the exact value of state will be returned. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service and it should be a secure random value.
`BSQCToken`	String	Required	The BSQC Token that can be used by the corresponding Online Service to enquire the status of the submitted bulk digital signing request or cancel the corresponding request

- **Example Callback**

```
// The descriptions of txID, code, and message are in Section 2.4.2
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>
// Callback Body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "businessID": "2YotnFZFEjr1zCsicMWpAA",
    "state": "unesidkd",
    "BSQCToken": "nnoadisauflanefhykdjf…f83h96e"
  }
}
```

9.5.4 Callback to Receive Bulk Digital Signing Result

This callback will be in the `multipart/form-data` Content-Type. The callback body is seperated into two parts by a `$<boundary>`. One part (`name="sigFile"`) is a JSON file that contains the signatures of the requested document hashes and/or PDF digests. The other part (`name="sigMetada"`) is a JSON string that contains digital signing metadata. These two parts are encrypted with the same CEK. The `$<boundary>` is automatically generated and is different in each callback, e.g. `--ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9`.

- **API Description**

Name	Description
Service Full Name	Callback to Receive Bulk Digital Signing Result
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	1.0
Description of Service	This callback will provide the bulk digital signing result to Online Service.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	1440 minutes (1 day)	Online Service could treat the bulk digital signing as failed if it doesn't get digital signing result callback within 1 day.

- **Callback Parameters**

Parameter	Type	Presence	Description
sigMetada			
`businessID`	String	Required	`businessID` is a unique identifier for Online Service to differentiate different request. Online Service can use the `businessID` to relate the callback message with the original request.
`state`	String	Optional	If the `state` parameter has been present in the request message, the exact value of state will be returned. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service and it should be a secure random value.
`cert`	String	Optional	Base64-encoded DER format certificate for the "iAM Smart" user. "iAM Smart" System will provide this to Online Service only when the digital signing is successful.
`totalCallbacks`	Integer	Optional (Conditional)	If `maxCallbackSigs` (m) is less than the number of hashes and digests (n) when

			submitting request (Section 9.5.1), the digital signing results of these documents will be called back in `totalCallbacks = int((n-1) / m + 1)` times. `m` signatures will be returned in each of the first `totalCallbacks-1` callbacks. `n-(totalCallbacks-1)*m` signatures will be returned in the last callback. If `totalCallbacks` is greater than 1, it must be specified. The default value is 1.
`callbackSeq`	Integer	Optional (Conditional)	If `totalCallbacks` is greater than 1, this `callbackSeq` should be `1, 2, …, totalCallbacks`. If `totalCallbacks` is greater than 1, this `callbackSeq` must be specified. The default value is 1.
`totalSignNum`	Integer	Optional (Conditional)	The Number of signed documents. It should be equal to the number of documents submitted by the user.
`failSignNum`	Integer	Optional (Conditional)	The Number of signed failed documents. It should be less than the number of documents submitted by users.
sigFile (For Hash Signature)			
`id`	String	Required	The document unique identifier. It should be ASCII string with length less than or equal to 36 chars.
`hash`	String	Required	The document hash submitted by Online Service in the API request
`timestamp`	Long	Required (Conditional)	Timestamp in milliseconds since January 1, 1970 00:00:00 GMT. “iAM Smart” System will provide this to Online Service only when digital signing is successful.
`signature`	String	Required (Conditional)	Base64-encoded signature result string. “iAM Smart” System will provide this to Online Service only when digital signing is successful.

sigFile (For PDF Signature)			
id	String	Required	The document unique identifier. It should be ASCII string with length less than or equal to 36 chars.
docDigest	String	Required	The pdf document digest submitted by Online Service in the API request
pdfSignature	String	Required (Conditional)	Base64-encoded PKCS#7 object that is the actual PDF signature value. It contains signer's certificate, signed hash value, and the digital signing timestamp information. Online Service can embed this value to the PDF document for future verification. "iAM Smart" System will provide this to Online Service only when the digital signing is successful.

- **Example Callback (Signing Success)**

```
// Line breaks are for legibility only.
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>

// Callback Header
// The $<boundary> looks like --ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9
// It is automatically generated and is different in every callback
Content-Type: multipart/form-data;boundary=$<boundary>

// Callback Body
$<boundary>
Content-Disposition: form-data; name="sigFile"; filename="signatures.txt"
Content-Type: text/plain
Content-Length: 62892
// The content of the text file looks like the following.
// Line breaks are for legibility only.
"signatures": [
  {
    "id": "egh…cva",
    "hash": "tGzv3JOkF0XG5Qx2TlKWIA",
    "timestamp": 1556450176000,
    "signature": "nnoadisauflanefhykdjf"
  },
  ……
  {
    "id": "hrf…bnd",
```

```
    "docDigest": "tGzv3JOkF0XG5Qx2TlKWIA",
    "pdfSignature": "sdfGSDGsdfaGDEHfjslgGQG……GSGjljlkjwmh"
  }
]

$<boundary>
Content-Disposition: form-data; name="sigMetada"
Content-Type: application/json;charset=UTF-8
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "businessID": "2YotnFZFEjr1zCsicMWpAA",
    "state": "unesidkd",
    "cert": "sdfGSDGsdfaGDEHfjslgGQG……GSGjljlkjwmh",
    "totalCallbacks": 10,
    "callbackSeq": 2,
    "totalSignNum": 100,
    "failSignNum": 20

  }
}
$<boundary>
```

- **Example Callback (Signing Failure)**

```
// Line breaks are for legibility only.
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>

// Callback Header
// The $<boundary> looks like --ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9
// It is automatically generated and is different in every callback
Content-Type: multipart/form-data;boundary=$<boundary>

// Callback Body
$<boundary>
Content-Disposition: form-data; name="sigMetada"
Content-Type: application/json;charset=UTF-8
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D71002",
  "message": "failed to request signing",
  "content": {
```

```
    "businessID": "2YotnFZFEjr1zCsicMWpAA",
    "state": "unesidkd",
  }
}
$<boundary>
```

9.5.5 Online Service Acknowledges Bulk Digital Signing Result

- **API Description**

Name	Description
Service Full Name	Online Service acknowledges bulk digital signing result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/bdss/ackResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online Service calls this API to acknowledge "iAM Smart" System if the result of the bulk digital signature is accepted or not. Online Service should send the acknowledgement regardless of whether the digital signing result (Section 9.5.4) is success or failure.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for Online Service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`signingResult`	String	Required	"SR001": digital signature is accepted "SR002": digital signature is rejected "SR003": no digital signature was received
`signatures`	ListArray	Required *(Conditional)*	When signingResult has no value, there must be values here
Details for "signatures"			
`id`	String	Required	The document unique identifier. It should be ASCII string with length less than or equal to 36 chars.
`signingResult`	String	Required	"SR001": digital signature is accepted "SR002": digital signature is rejected "SR003": no digital signature was received

- **Example Request**

```
// Line breaks are for legibility only.
POST
https://<iAM_Smart_domain>/api/v1/bdss/ackResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Request Body
{
  "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
  "signingResult": "SR001"
}
or
{
"businessID": "bbb8aae57c104cda40c93843ad5e6db8",
"signingResult": "",
"signatures": :[
       {
          "id":"egh…cva",
          "signingResult":"SR001"
       },
       {
          "id":"egh…cva",
          "signingResult":"SR002"
       },
       {
          "id":"egh…cva",
          "signingResult":"SR003"
       }
 ]
}
```

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
```

```
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS"
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {signingResult}"
}
```

9.5.6 Enquire Bulk Digital Signing Status

- **API Description**

Name	Description
Service Full Name	Enquire Bulk Digital Signing Status
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/bdss/enquireStatus`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online Service calls this API to enquire the digital signing status of the submitted bulk digital signing request.

- **Request Parameters**

Parameter	Type	Presence	Description
`BSQCToken`	String	Required	The BSQC Token which is returned in section 9.5.3. Maximum Length: 32
`openID`	String	Required	Tokenised ID value Maximum Length: 64

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
```

```
https://<iAM_Smart_domain>/api/v1/bdss/enquireStatus
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Request Body
{
  "BSQCToken": "nnoadisauflanefhykdjf…f83h96e",
  "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
status	String	Required	The digital signing status of the submitted bulk digital signing request. Its value should be: “SS001”: Pending for Signing “SS002”: Signing in Progress “SS003”: Signing Completed “SS004”: Signing Cancelled “SS005”: Signing Failed “SS006”: No Record Found

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "status": "SS003"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
```

```
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {BSQCToken}"
}
```

9.5.7 Cancel Bulk Digital Signing Request

- **API Description**

Name	Description
Service Full Name	Online Service acknowledges digital signing result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/bdss/ackResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online Service calls this API to cancel the submitted bulk digital signing request. Only the request of which status is “Pending for Signing” can be cancelled.

- **Request Parameters**

Parameter	Type	Presence	Description
`BSQCToken`	String	Required	The BSQC Token which is returned in section 9.5.3. Maximum Length: 32
`openID`	String	Required	Tokenised ID value Maximum Length: 64

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/bdss/cancelRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
```

```
// Request Body
{
  "BSQCToken": "nnoadisauflanefhykdjf…f83h96e",
  "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS"
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D71007",
  "message": "the request to be cancelled is not Pending for Signing"
}
```

9.5.8 Request Bulk Digital Signing (appv2)

- **API Description**

Name	Description
Service Full Name	Request Bulk Digital Signing
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/appv2/bdss/account/initiateRequest`
Request Type	POST
Service Version	V2.0.0
Description of Service	Online Service can use this API to make request for bulk digital signing by sending `accessToken` and `openID` to "iAM Smart" System.

- **Request Parameters**

Online Service can specify the `Content-Type` of this as either `application/json` or `multipart/form-data`. The default `Content-Type` is `application/json`.

- If the request Content-Type is `application/json`, the fields in the following table should be put in the request body in JSON format.
- If the request Content-Type is `multipart/form-data`, the request body should be seperated into two parts by a `$<boundary>`. One part (`name="reqFile"`) is a JSON file that contains the request document hashes or digests, i.e. the `documents` and its sub-fields in the following table. The other part (`name="reqMetada"`) is a JSON string that contains request metadata, i.e. the fields except `documents` and its sub-fields. These two parts should be encrypted with the same CEK. The `$<boundary>` is automatically generated and is different in each request, e.g. `--ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9`.

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for Online Service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value Maximum Length: 64
`source`	String	Required	App_Link (iOS) or App_Package (Android)
`clientRedirectURI`	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal.

			Encoded and unencoded commas (“,”) are not accepted.
`packageName`	String	Required (Android only)	Package name of the Online Service App for callback use.
`activityClass`	String	Required (Android only)	Activity class name of the Online Service App for callback use.
`activityParams`	String	Optional (Android only)	Optional params used during callback.
`serverRedirectURI`	String	Required	Callback URI for the endpoint that Online Service uses to receive BSQC Token (Section 9.5.3)
`callbackResultURI`	String	Required	Callback URI for the endpoint that Online Service uses to receive bulk digital signing result (Section 9.5.4)
`state`	String	Optional	If `state` parameter is presented in the request message, the same state value will be returned to Online Service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
`HKICHash`	String	Optional	Bulk digital signing request will only be processed when the hash of the HKIC number of the “iAM Smart” user is matched with the HKICHash provided by Online Service. Online Service should convert the HKIC number into hash value using SHA256 before sending to “iAM Smart” System. Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
`department`	String	Optional	The department that initiates the bulk digital signing request. This will be displayed on the consent page of the “iAM Smart” mobile app.

			Maximum Length: 100
serviceName	String	Required	The service name that will be displayed on the consent page of the "iAM Smart" mobile app. Maximum Length: 255
requestName	String	Required	The bulk digital signing request name that will be displayed on the consent page of the "iAM Smart" mobile app. Maximum Length: 255
maxCallbackSigs	Integer	Optional	The maximal number of signatures that the Online Service endpoint can accept in one callback for receiving the bulk digital signing result. The default value is 1000.

documents	Array	Required	The document hashes and/or pdf digests that are to be signed by "iAM Smart". The number of array items should be greater than 0 and less than 1001. The array items are in one of the following JSON dictionary formats:
			<u>For document hash</u>
			{
			"id": "egh…cva",
			"documentName": "Document",
			"hashCode": "965…56d",
			"sigAlgo": "SHA256withRSA"
			}
			<u>For PDF document digest</u>
			{
			"id": "egh…cva",
			"documentName": "PDF Document",
			"docDigest": "965…56d"
			}
			The following rows show the detail of each dictionary key.

Details for "documents"			
id	String	Required	The document unique identifier. It should be ASCII string with length less than or equal to 36 chars.

documentName	String	Required	The document name that will be displayed in the "iAM Smart" mobile app when asking user's consent for the bulk digital signing request. Maximum Length: 255
hashCode	String	Required (Conditional)	The document hash to be signed. Online Service should compute the hash and send this hash to "iAM Smart" System for digital signing. The value should be Base64 encoded.
sigAlgo	String	Optional	Signature algorithm to be used. Online Service can specify either SHA256withRSA or NONEwithRSA. The default value is SHA256withRSA. While using NONEwithRSA, hashCode provided must be hashed with SHA256.
docDigest	String	Required (Conditional)	The PDF document digest to be signed. Online Service should compute the digest using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter. The value should be Base64 encoded.

- **Example Request**

The following is the example request when Content-Type is set as application/json.

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/appv2/bdss/account/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Request Body
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
```

```
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
    "source": "App_Package",
    "packageName": "com.onlineservice.myapp",
    "activityClass": "callback_activity",
    "activityParams": "callback_param",
    "serverRedirectURI":
"https://<rp_domain>/<rp_context>/<call_back_endpoint>",
    "callbackResultURI": "https://<rp_domain>/<rp_context>/<endpoint>",
    "state": "eddd527b6",
    "HKICHash": "6ea37e641260714009a880a8057032a414009a80f667efc",
    "department": "Digital Policy Office",
    "serviceName": "Online Service 1",
    "requestName": "Bulk Digital Signing for Contracts",
    "maxCallbackSigs": 200,
    "documents": [
      {
        "id": "aef…cth",
        "documentName": "Document",
        "hashCode": "965e…356d",
        "sigAlgo": " SHA256withRSA"
      },
      ……
      {
        "id": "trx…hnm",
        "documentName": "PDF Document",
        "docDigest": "7rtg…y67d"
      },
    ]
}
```

The following is the example request when `Content-Type` is set as `multipart/form-data`.

```
// Line breaks are for legibility only.
POST
https://<iAM_Smart_domain>/api/appv2/bdss/account/initiateRequest

// Request Header
// The $<boundary> looks like --ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9
// It is automatically generated and is different in every callback
Content-Type: multipart/form-data;boundary=$<boundary>
clientID: "edae2e2529ff46228af1e4d18c8405d1"
```

```
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"

// Request Body
$<boundary>
Content-Disposition: form-data; name="reqFile"; filename="documents.txt"
Content-Type: text/plain
Content-Length: 62892
// The content of the text file looks like the following.
// Line breaks are for legibility only.
 {
   "documents": [
     {
       "id": "aef…cth",
       "documentName": "Document",
       "hashCode": "965e…356d",
       "sigAlgo": " SHA256withRSA"
     },
     ……
     {
       "id": "trx…hnm",
       "documentName": "PDF Document",
       "docDigest": "7rtg…y67d"
     }
   ]
}

$<boundary>
Content-Disposition: form-data; name="reqMetadata"
Content-Type: application/json;charset=UTF-8
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
   "source": "App_Package",
   "packageName": "com.onlineservice.myapp",
   "activityClass": "callback_activity",
   "activityParams": "callback_param",
   "serverRedirectURI":
```

```
"https://<rp_domain>/<rp_context>/<call_back_endpoint>",
   "callbackResultURI": "https://<rp_domain>/<rp_context>/<endpoint>",
   "state": "eddd527b6",
   "HKICHash": "6ea37e641260714009a880a8057032a414009a80f667efc",
   "department": "Digital Policy Office",
   "serviceName": "Online Service1",
   "requestName": " Bulk Digital Signing for Contracts",
   "maxCallbackSigs": 200,
}
$<boundary>
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If user completes authentication by scanning QR code (i.e. different device), then "true" will be returned to Online Service. `authByQR` should always be `false` in appv2 API.
`ticketID`	String	Required (Conditional)	`TicketID` is a unique identifier to trigger "iAM Smart" mobile app when authByQR is `false`.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": false,
    "ticketID": "bbb8aae57c104cda40c93843ad5e6db8"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
```

```
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {documents}"
}
```

10. BULK DIGITAL SIGNING WITHOUT SERVICE LOGIN (AKA ANONYMOUS BULK DIGITAL SIGNING)

10.1 Overview

"iAM Smart+" version supports bulk digital signing with legal backing under the Electronic Transactions Ordinance (Cap. 553) for handling statutory documents and procedures. Similar to Bulk Digital Signing API, online service can request "Anonymous Bulk Digital Signing" API to complete digital signing online for multiple documents with only one digital signing cycle. Unlike the accessToken obtained from Authentication API, the accessToken received in this API can only be used once. It can be used in many cases, such as digital signing online application forms and digital signing contracts and agreements.

10.2 Prerequisite

- To request the Anonymous Bulk Digital Signing API, the "iAM Smart" user shall be a "iAM Smart+" user. Online service shall prompt the message in their website / mobile app and guide the user to upgrade his/her user account if he/she is not the "iAM Smart+" user.
- Online service shall refer to "Reference System Flow and User Interface Specifications for "iAM Smart" Adoption" for the detail UI/UX requirements.

10.3 Scope

Online service shall submit the application for using the following scope values to "iAM Smart" Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_bulksign`	Scope for Bulk Digital Signing

10.4 Use Cases and Scenarios

10.4.1 Anonymous Bulk Digital Signing (Online Service Website in Different Device)

The sequence diagram below shows how an anonymous user authorises and signs multiple document hashes and/or digests when Online Service website and the "iAM Smart" Mobile App are running in the different devices.

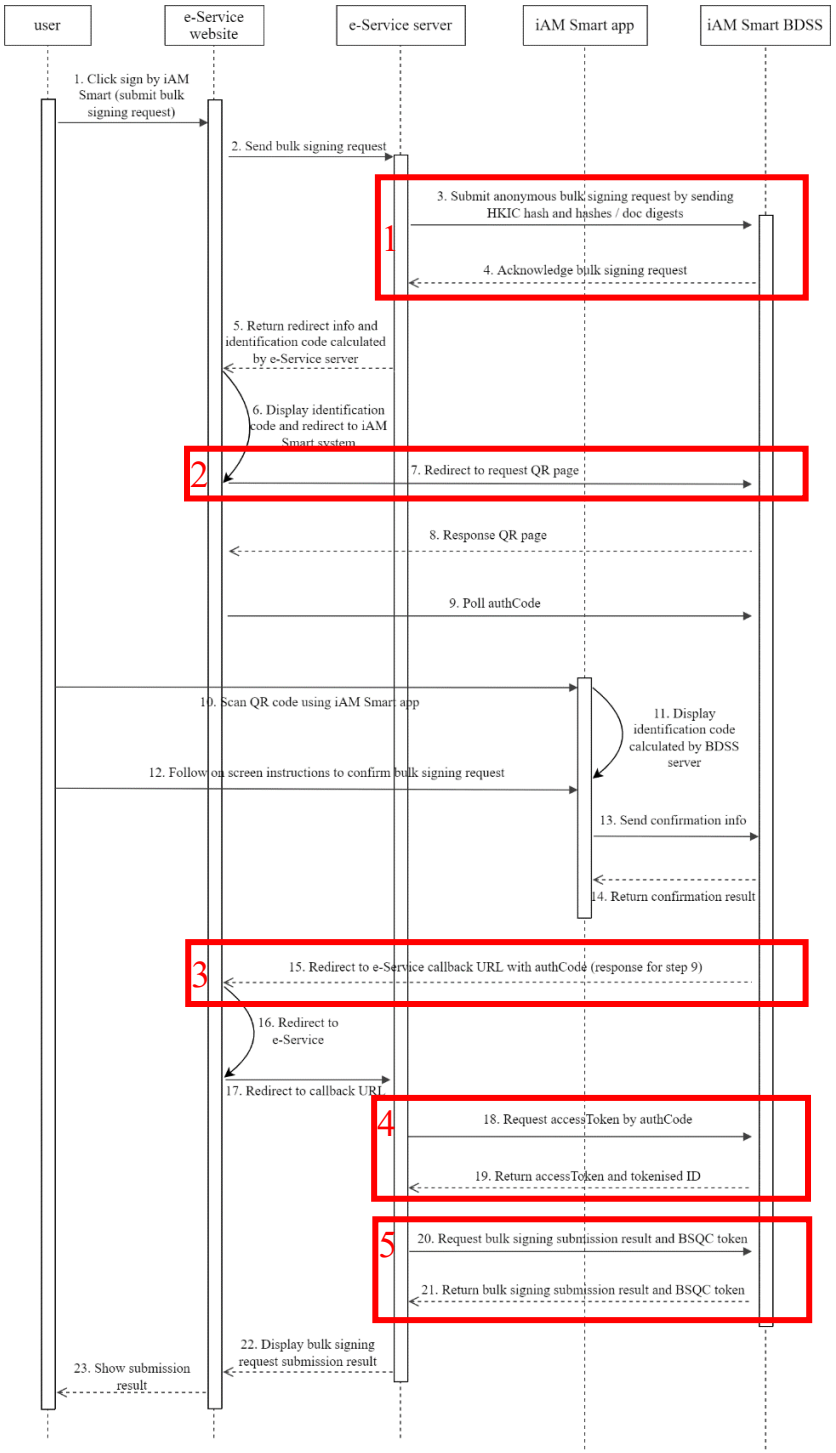

Figure-34 Anonymous Bulk Digital Signing (Online Service Website in Different Device)

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Bulk Digital Signing	10.5.1
2	Request QR Page	10.5.2
3	Callback with authCode to Online Service App	10.5.4
4	Request accessToken & Tokenised ID	10.5.5
5	Request BSQC Token	10.5.6

10.4.2 Anonymous Bulk Digital Signing (Online Service Website in Same Device)

The sequence diagram below shows how an anonymous user authorises and signs multiple document hashes and/or digests when Online Service website and the "iAM Smart" Mobile App are running in the same devices.

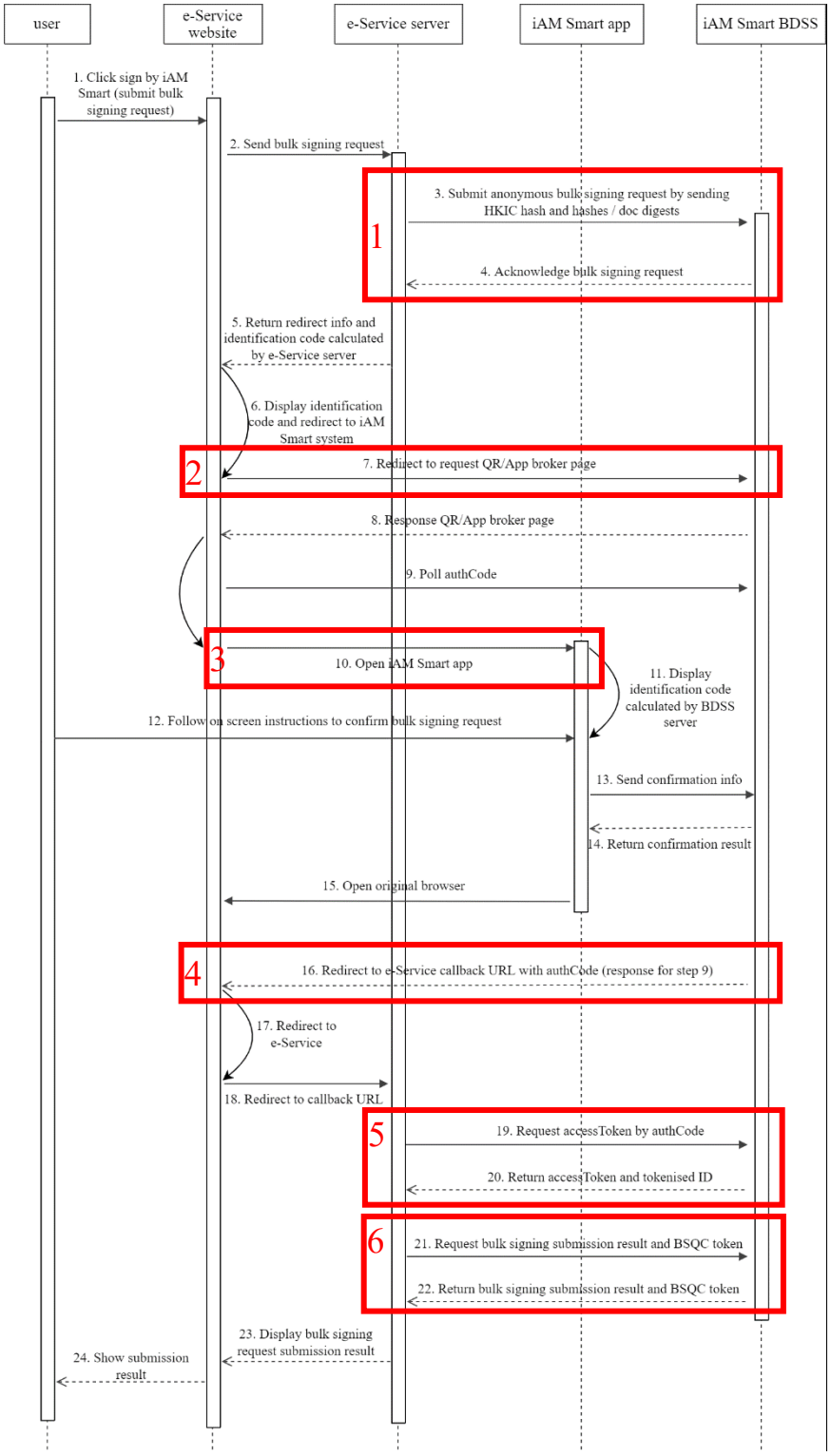

Figure-35 Anonymous Bulk Digital Signing (Online Service Website in Same Device)

APIs interactions between Online Service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Bulk Digital Signing	10.5.1
2	Request QR Page	10.5.2
3	Open “iAM Smart” Mobile App for Anonymous Bulk Digital Signing (appv1)	10.5.3
4	Callback with authCode to Online Service App	10.5.4
5	Request accessToken & Tokenised ID	10.5.5
6	Request BSQC Token	10.5.6

10.4.3 Anonymous Bulk Digital Signing (Online Service App in Different Device)

The sequence diagram below shows how an anonymous user authorises and signs multiple document hashes and/or digests when Online Service App and the "iAM Smart" Mobile App are running in different devices.

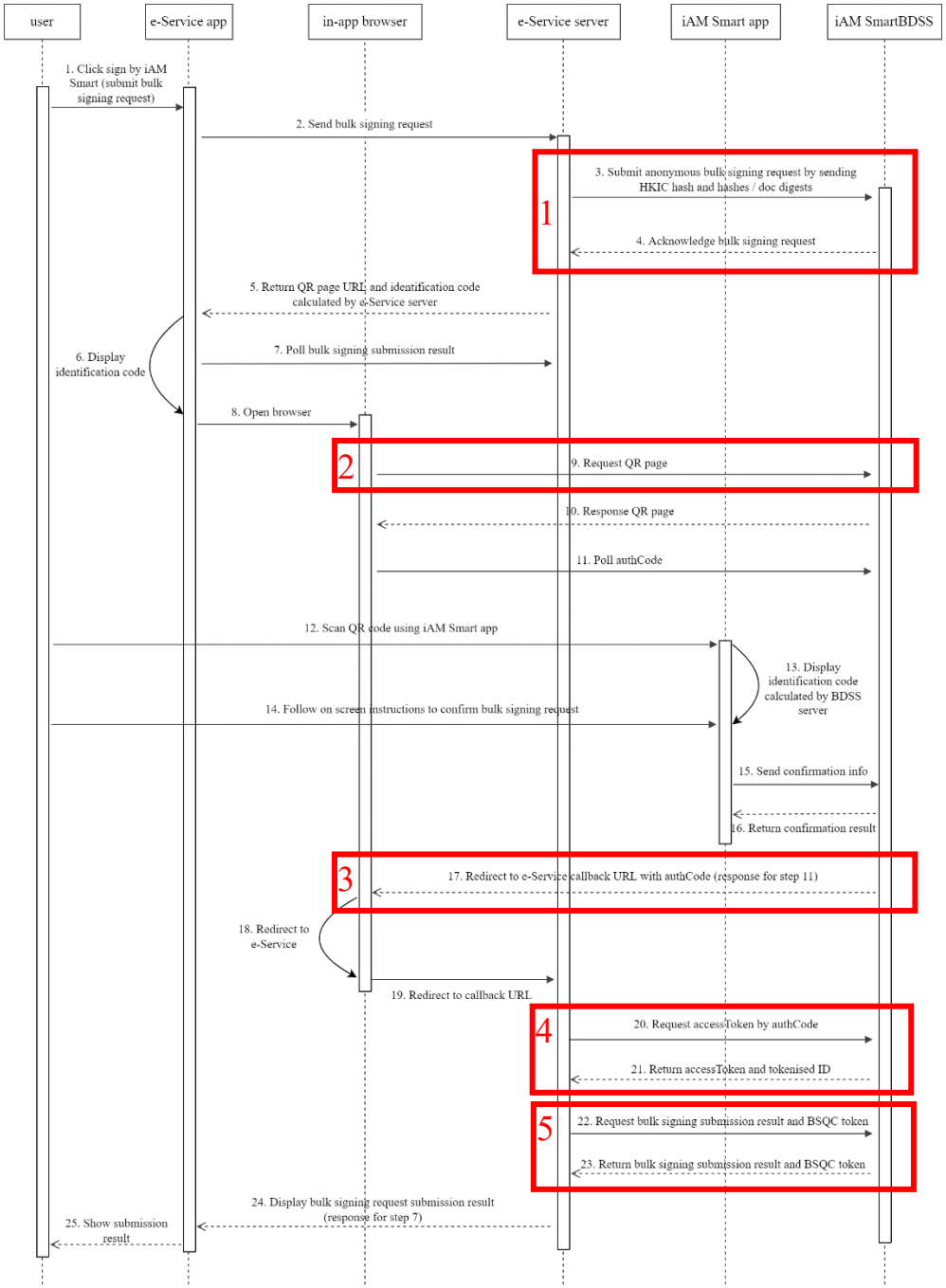

Figure-36 Anonymous Bulk Digital Signing (Online Service App in Different Device)

APIs interactions between Online Service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Bulk Digital Signing	10.5.1
2	Request QR Page	10.5.2
3	Callback with authCode to Online Service App	10.5.4
4	Request accessToken & Tokenised ID	10.5.5
5	Request BSQC Token	10.5.6

10.4.4 Anonymous Bulk Digital Signing (Online Service App in Same Device)

The sequence diagram below shows how an anonymous user authorises and signs multiple document hashes and/or digests when Online Service App and the "iAM Smart" Mobile App are running in the same device.

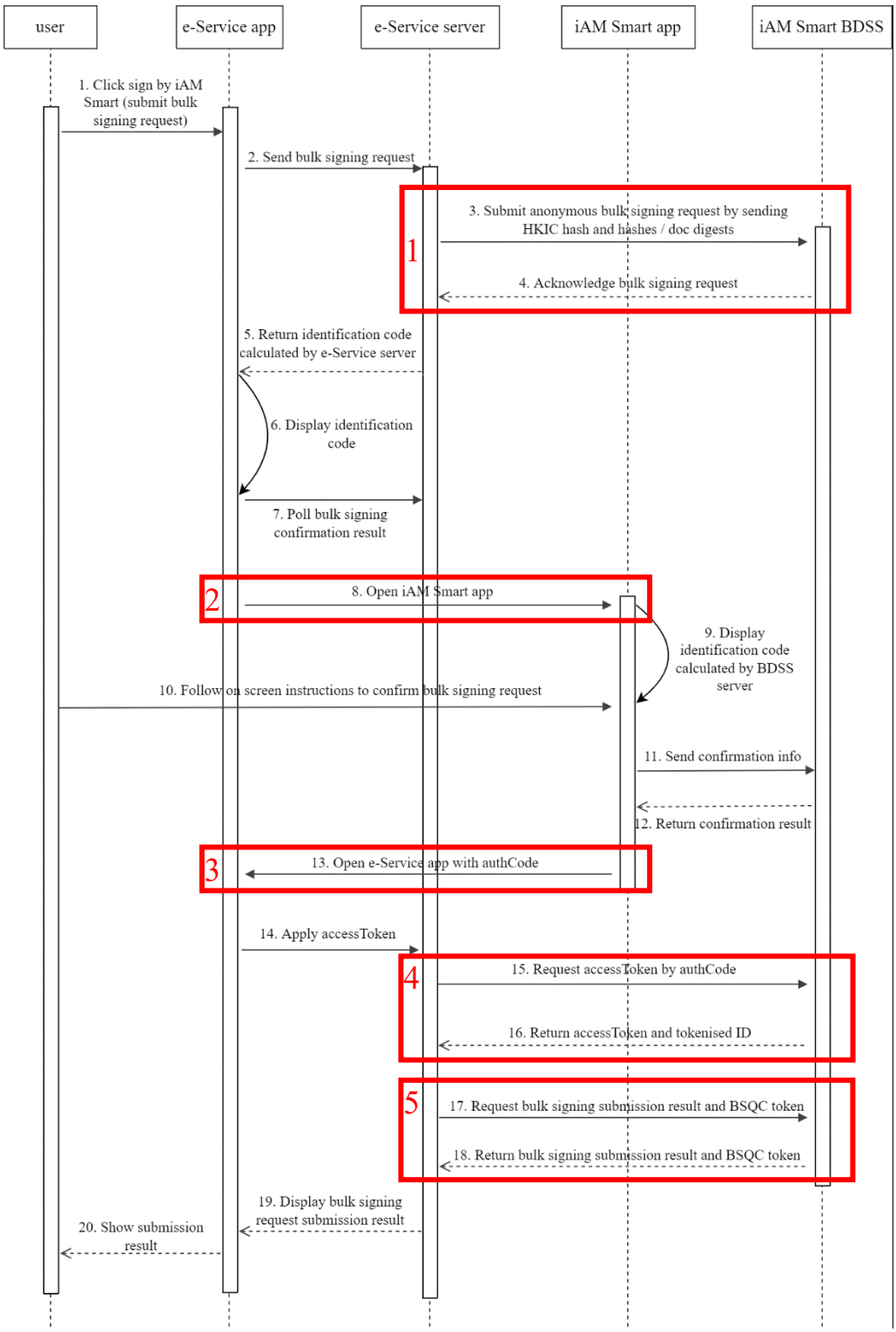

Figure-37 Anonymous Bulk Digital Signing (Online Service App in Same Device)

APIs interactions between Online Service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Bulk Digital Signing	10.5.1
2	Open “iAM Smart” Mobile App for Anonymous Bulk Digital Signing (appv1)	10.5.3
	Open “iAM Smart” Mobile App for Anonymous Bulk Digital Signing (appv2)	10.5.11
3	Callback with authCode to Online Service App	10.5.4
4	Request accessToken & Tokenised ID	10.5.5
5	Request BSQC Token	10.5.6

10.4.5 Bulk Digital Signing Result Callback

The sequence diagram below shows how the bulk digital signing result is sent to Online Service's callback endpoint after the digital signing process is completed.

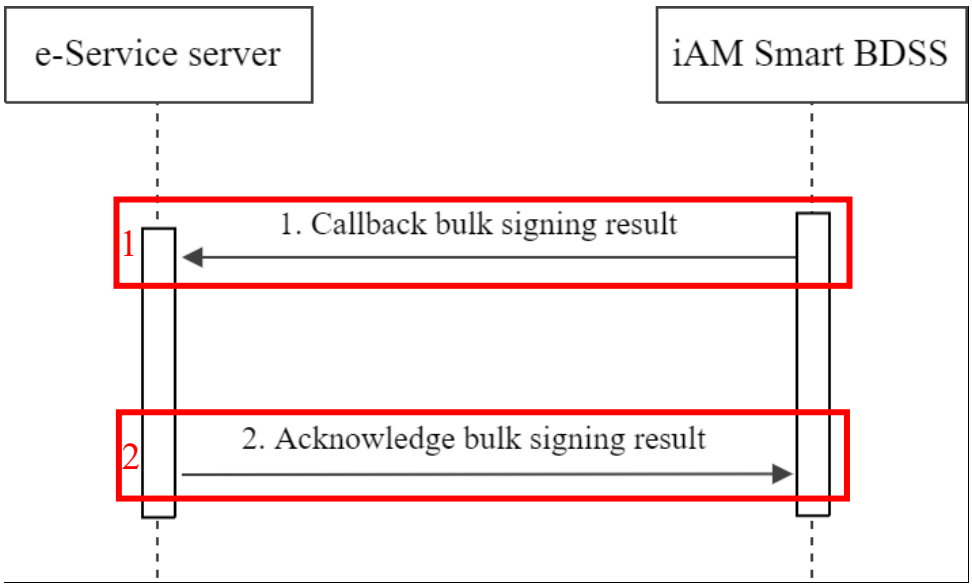

Figure-38 Bulk Digital Signing Callback

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Callback to Receive Bulk Digital Signing Result	10.5.7
2	Online Service Acknowledges Bulk Digital Signing Result	10.5.8

10.4.6 Enquire Bulk Digital Signing Result

The sequence diagram below shows how Online Service queries the status of the submitted digital signing request.

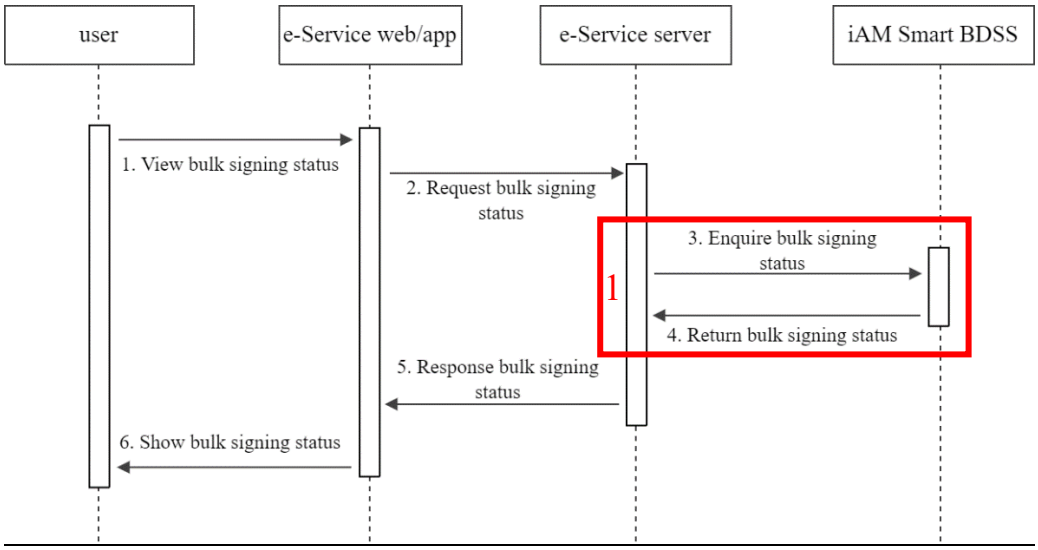

Figure-39 Bulk Digital Signing Status Enquiry

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Enquire Bulk Digital Signing Status	10.5.9

10.4.7 Cancel Bulk Digital Signing Request

The sequence diagram below shows how Online Service cancels the submitted digital signing request.

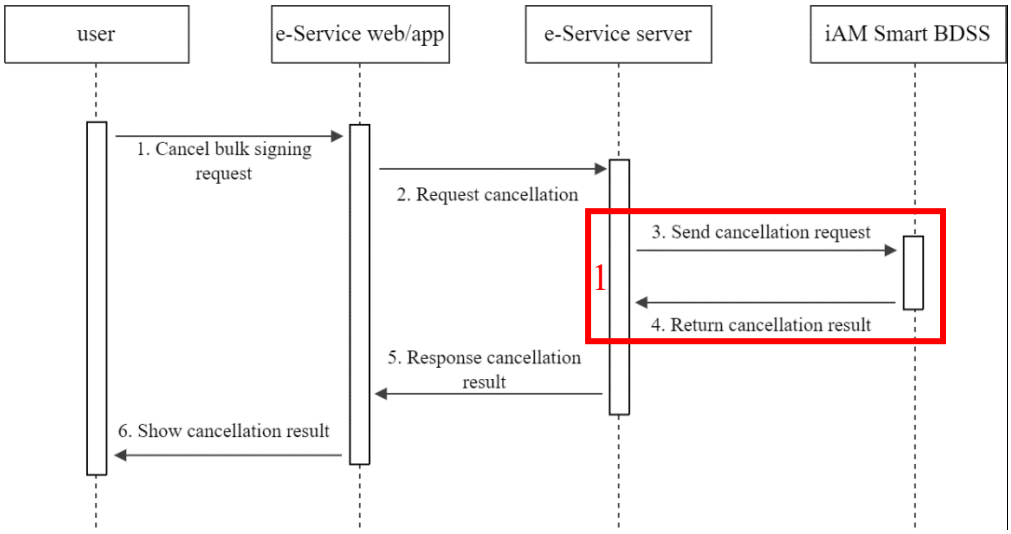

Figure-40 Bulk Digital Signing Cancellation

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Cancel Bulk Digital Signing Request	10.5.10

10.5 API Implementation Details

10.5.1 Request Anonymous Bulk Digital Signing

- **API Description**

Name	Description
Service Full Name	Request Anonymous Bulk Digital Signing
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/bdss/anonymous/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online Service can use this API to submit anonymous bulk digital signing request by sending `documents` and `HKICHash` to "iAM Smart" System.

- **Request Parameters**

Online Service can specify the `Content-Type` of this as either `application/json` or `multipart/form-data`. The default `Content-Type` is `application/json`.

- If the request Content-Type is `application/json`, the fields in the following table should be put in the request body in JSON format.
- If the request Content-Type is `multipart/form-data`, the request body should be seperated into two parts by a `$<boundary>`. One part (`name="reqFile"`) is a JSON file that contains the request document hashes or digests, i.e. the `documents` and its sub-fields in the following table. The other part (`name="reqMetada"`) is a JSON string that contains request metadata, i.e. the fields except `documents` and its sub-fields. These two parts should be encrypted with the same CEK. The `$<boundary>` is automatically generated and is different in each request, e.g. `--ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9`.

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for Online Service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`callbackResultURI`	String	Required	Callback URI for the endpoint that Online Service uses to receive bulk digital signing result (Section 10.5.7)
`HKICHash`	String	Required	Bulk digital signing request will only be processed when the hash of the HKIC number of the "iAM Smart" user is

			matched with the HKICHash provided by Online Service. Online Service should convert the HKIC number into hash value using SHA256 before sending to "iAM Smart" System. Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
`department`	String	Optional	The department that initiates the bulk digital signing request. This will be displayed on the consent page of the "iAM Smart" mobile app. Maximum Length: 100
`serviceName`	String	Required	The service name that will be displayed on the consent page of the "iAM Smart" mobile app. Maximum Length: 255
`requestName`	String	Required	The bulk digital signing request name that will be displayed on the consent page of the "iAM Smart" mobile app. Maximum Length: 255
`maxCallbackSigs`	Integer	Optional	The maximal number of signatures that the Online Service endpoint can accept in one callback for receiving the bulk digital signing result. The default value is 1000.

`documents`	Array	Required	The document hashes and/or pdf digests that are to be signed by "iAM Smart". The number of array items should be greater than 0 and less than 1001. The array items are in one of the following JSON dictionary formats:
			<u>For document hash</u>
			`{`
			`"id": "egh…cva",`
			`"documentName": "Document",`
			`"hashCode": "965…56d",`

			"sigAlgo": "SHA256withRSA"
			}
			For PDF document digest
			{
			"id": "egh…cva",
			"documentName": "PDF Document",
			"docDigest": "965…56d"
			}
			The following rows show the detail of each dictionary key.

Details for "documents"			
id	String	Required	The document unique identifier. It should be ASCII string with length less than or equal to 36 chars.
documentName	String	Required	The document name that will be displayed in the "iAM Smart" mobile app when asking user's consent for the bulk digital signing request. Maximum Length: 255
hashCode	String	Required (Conditional)	The document hash to be signed. Online Service should compute the hash and send this hash to "iAM Smart" System for digital signing. The value should be Base64 encoded.
sigAlgo	String	Optional	Signature algorithm to be used. Online Service can specify either SHA256withRSA or NONEwithRSA. The default value is SHA256withRSA. While using NONEwithRSA, hashCode provided must be hashed with SHA256.
docDigest	String	Required (Conditional)	The PDF document digest to be signed. Online Service should compute the digest using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter. The value should be Base64 encoded.

- **Example Request**

The following is the example request when `Content-Type` is set as `application/json`.

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/bdss/anonymous/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Request Body
{
   "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
   "callbackResultURI": "https://<rp_domain>/<rp_context>/<endpoint>",
   "HKICHash": " 6ea37e641260714009a880a8057032a414009a80f667efc",
   "department": " Digital Policy Office",
   "serviceName": "Online Service 1",
   "requestName": "Bulk Digital Signing for Contracts",
   "maxCallbackSigs": 200,
   "documents": [
     {
       "id": "aef…cth",
       "documentName": "Document",
       "hashCode": "965e…356d",
       "sigAlgo": " SHA256withRSA"
     },
     ……
     {
       "id": "trx…hnm",
       "documentName": "PDF Document",
       "docDigest": "7rtg…y67d"
     }
   ]
}
```

The following is the example request when `Content-Type` is set as `multipart/form-data`.

```
// Line breaks are for legibility only.
POST
https://<iAM_Smart_domain>/api/v1/bdss/anonymous/initiateRequest

// Request Header
// The $<boundary> looks like --ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9
// It is automatically generated and is different in every callback
Content-Type: multipart/form-data;boundary=$<boundary>

clientID: "edae2e2529ff46228af1e4d18c8405d1"

signatureMethod: "HmacSHA256"

signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"

timestamp: 1557048906183

nonce: "e893647dc4204eb9b7b8eddd527b687c"

// Request Body
$<boundary>
Content-Disposition: form-data; name="reqFile"; filename="documents.txt"
Content-Type: text/plain
Content-Length: 62892
// The content of the text file looks like the following.
// Line breaks are for legibility only.
 {
   "documents": [
     {
       "id": "aef…cth",
       "documentName": "Document",
       "hashCode": "965e…356d",
       "sigAlgo": " SHA256withRSA"
     },
     ……
     {
       "id": "trx…hnm",
       "documentName": "PDF Document",
       "docDigest": "7rtg…y67d"
     }
   ]
}

$<boundary>
Content-Disposition: form-data; name="reqMetadata"
Content-Type: application/json;charset=UTF-8

{
```

```
    "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
    "callbackResultURI": "https://<rp_domain>/<rp_context>/<endpoint>",
    "HKICHash": "6ea37e641260714009a880a8057032a414009a80f667efc",
    "department": "Digital Policy Office",
    "serviceName": "Online Service 1",
    "requestName": "Bulk Digital Signing for Contracts",
    "maxCallbackSigs": 200
}
$<boundary>
```

- **Response Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier that will be used to identify this request in the following steps of the digital signing workflow.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "ticketID": "561b6f1ce5a043ba8aae33fb5c50386d"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {documents}"
}
```

10.5.2 Request QR Page

- **API Description**

Name	Description
Service Full Name	Request QR Page
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/auth/getQR`
Request Type	GET
Service Version	1.0.0
Description of Service	Online service calls this API to get QR/App broker page or QR page. After user authorises login or anonymous request on the "iAM Smart" Mobile App, the page will be redirected to the `redirectURI` with `authCode` and `state` parameters. If user denies, only the `state` parameter will be redirected.

- **Request Parameters**

Parameter	Type	Presence	Description
`clientID`	String	Required	Online service client identifier. The clientID will be assigned to online service at the initial registration.
`responseType`	String	Required	value MUST be set to `code`.
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.
`redirectURI`	String	Required	Callback redirect URI. The value should be URL encoded and registered in self-service portal.
`scope`	String	Required	The value should be URL encoded. Specify the scope of "iAM Smart" functions: - `eidapi_bulksign` The value of the scope parameter is expressed as a list of space-delimited, case-sensitive strings.
`ticketID`	String	Required (Conditional)	Required in anonymous bulk digital signing workflows for getting QR page.
`lang`	String	Optional	Language to display: en-US, zh-HK, or zh-CN. If this parameter is not specified, zh-HK will be shown.

`state`	String	Optional	If `state` parameter is presented in the request message, the same state value will be returned to online service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by online service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.
`brokerPage`	Boolean	Optional	If `brokerPage` is set to `true`, Universal Link (iOS) / App Link (Android) will be leveraged to open "iAM Smart" Mobile App. This feature is useful for online service supporting mobile web version while trigger "iAM Smart" Mobile App or showing QR page automatically. (i.e. Show QR page without detecting whether "iAM Smart" Mobile App is installed). The default value is `false`.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
GET
https://<iAM_Smart_domain>/api/v1/auth/getQR?clientID=Online Service1
&responseType=code
&source=Android_Chrome
&redirectURI=https%3A%2F%2Frp_domain%2Frp_context%2Fcall_back_endpoint
&scope=eidapi_bulksign
&lang=en-US
&state=eb9b7b8eddd5
```

- **Response Parameters**

N/A

10.5.3 Open “iAM Smart” Mobile App for Anonymous Bulk Digital Signing (appv1)

- **Using URL Scheme**

Name	Description
Service Full Name	Open “iAM Smart” Mobile App for Anonymous Bulk Digital Signing
URI (as in RESTFUL API)	`<"iAM Smart" app URL scheme>://anon_bulk-sign`
Service Version	V1.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific Anonymous Bulk Digital Signing in “iAM Smart” Mobile App. The URL schemes supported on iOS and Android versions of “iAM Smart” Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required	`TicketID` is a unique identifier provided by “iAM Smart”, Online service retrieve `ticketID` while requesting the context. It is ASCII string with length less than or equal 36 chars.
`source`	String	Required (Conditional)	This parameter is required only in anonymous bulk digital signing workflows. Request initiator (App_Scheme, App_Link). This API does not support multiple mobile Apps for 1 client id (1 client for 1 mobile App), the dedicated mobile app shall register both self-service portal and support team.
`redirectURI`	String	Required (Conditional)	Callback redirect URI. This parameter is required only in anonymous bulk digital signing workflows and its value should be URL encoded.
`state`	String	Required (Conditional)	If `state` parameter is presented in the request message, the same state value will

			be returned to Online Service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://anon_bulk-sign
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

10.5.4 Callback with authCode to Online Service App

- **URL Scheme and Package Name**

Name	Description
Service Full Name	Callback with authCode to Online Service App
URL Scheme (iOS and Android)	`<Universal / App Link>`. The online service custom app scheme or Universal / App Link should be registered with the "iAM Smart" System during onboarding.
Package name (Android only)	`<package name>` and `<activity class name>`. The online service package name must be registered in the self-service portal. *Remark: Direct Login v2 (App) and appv2 API adopts package name verification instead of App Link.*
Description of Service	For appv1 API, the Online Service App will be invoked and launched by Universal / App Link. It makes use of deep linking to redirect users to the Online Services app. App Link can only work for mobile devices with Google Mobile Services (GMS). For appv2 API, the Android Online Service App will be invoked and launched by the package name. It make use of intent to redirect users to the Online Service app. The iOS Online Service App will be invoked and launched via Universal link.

	The Universal / App Link with the landing location as well as the package name plus activity name must be registered in the self-service portal and enabled by the support team.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **URL Scheme / activity Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required (Conditional)	`businessID` will only be returned in anonymous workflows. `businessID` is a unique identifier for online service to differentiate different requests. Online service can use the `businessID` to relate the callback message with the original request. Maximum Length: 36
`code`	String	Required (Conditional)	The authorisation code generated by the "iAM Smart" System. The authorisation code will be expired in 60 seconds after issuance. Online service MUST NOT use the authorisation code more than once. An error message will be returned if an authorisation code is expired or re-used.
`state`	String	Optional	If the `state` parameter has been presented in the request message, the same state value will be returned. It is used to prevent the CSRF attack. The value of state is defined by online service and it should be a secure random value.
`error_code`	String	Required (Conditional)	Return error code when exception.

- **Example URL Scheme**

Allow

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
?code=0ad186353c424c64897fcc00445c9ba1
&state=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
?error_code=D20001
&state=eddd527b6
```

- **Example Package Name**

Allow

```
Intent ii=new Intent(<package name>, <activity name>);
ii.putExtra("code", "0ad186353c424c64897fcc00445c9ba1");
ii.putExtra("state", "eddd527b6");
ii.putExtra("businessID", "b2c99aa83b0049e9ba370c5341681225");
startActivity(ii);
```

Deny

```
Intent ii=new Intent(<package name>, <activity name>);
ii.putExtra("error_code", "D20001");
ii.putExtra("state", "eddd527b6");
startActivity(ii);
```

10.5.5 Request accessToken & Tokenised ID

- **API Description**

Name	Description
Service Full Name	Request access token and tokenised ID with authCode
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/auth/getToken`
Request Type	POST
Service Version	1.0.0
Description of Service	Online service uses this API to retrieve the access token and Tokenised ID (openID). An authorisation code is necessary during the process. The `accessToken` and `openID` will be used to call corresponding “iAM Smart” services subsequently.

- **Request Parameters**

Parameter	Type	Presence	Description
code	String	Required	The authorisation code is received from the authorisation server. One time use only and will be expired in 1 minute.
grantType	String	Required	the value MUST be set to authorization_code.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/auth/getToken
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
  "code": "xxxa42e76bf4cb0846a68e6d83d6096",
  "grantType": "authorization_code"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
accessToken	String	Required	accessToken value can only be used **once**.
tokenType	String	Required	Token type, support "Bearer" only.
issueAt	Long	Required	The accessToken issue time is expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
expiresIn	Long	Required	The lifetime in milliseconds of the token. The value may vary for different Online Services.

`openID`	String	Required	Tokenised ID, uniquely generated for each user of each online service website or mobile application.g
`lastModifiedDate`	Long	Required	The datetime of the user complete registration at the "iAM Smart" System. The value will be updated when either of the following is valid:- (1) If any one of the following verified data are changed. English name Chinese name (* not applicable if it was marked as unverified during registration) Gender Date of birth (2) User re-register "iAM Smart" after "iAM Smart" de-registration. The modification time will be expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
`userType`	String	Required	`default` or `sign` `default`: "iAM Smart" user `sign`: "iAM Smart+" user (digital signing capability)
`scope`	String	Required	The scope of the token. Please refer to the corresponding section specified in each API function.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "accessToken": "0ad186353c424c64897fcc00445c9ba1",
    "tokenType": "Bearer",
```

```
    "issueAt": 1557053922938,
    "expiresIn": 14400000,
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
    "lastModifiedDate": 1560849218006,
    "userType": "sign",
    "scope": "eidapi_bulksign"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D40004",
  "message": "authCode not exist or expired",
}
```

10.5.6 Request BSQC Token

- **API Description**

Name	Description
Service Full Name	Request BSQC Token
URI (as in RESTFUL API)	https://<iAM_Smart_domain>/api/v1/bdss/getBSQCToken
Request Type	POST
Service Version	V1.0.0
Description of Service	Online Service uses this API to get BSQC Token for the submitted bulk digital signing request.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online Service could treat the bulk digital signing request as failed if it doesn't get BSQC Token callback within 18 minutes.

BSQCToken	1440 minutes (1 day)	The BSQC Token used to enquire digital signing status or cancel signing request is valid for 1 day.

- **Request Parameters**

Parameter	Type	Presence	Description
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value Maximum Length: 64

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/bdss/getBSQCToken
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Request Body
{
    "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`BSQCToken`	String	Required	The BSQC Token that can be used by the corresponding Online Service to enquire or cancel the submitted bulk digital signing request.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
```

```
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "BSQCToken": "nnoadisauflanefhykdjf…f83h96e"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {accessToken}"
}
```

10.5.7 Callback to Receive Bulk Digital Signing Result

This callback will be in the `multipart/form-data` Content-Type. The callback body is seperated into two parts by a `$<boundary>`. One part (`name="sigFile"`) is a JSON file that contains the signatures of the requested document hashes and/or PDF digests. The other part (`name="sigMetada"`) is a JSON string that contains digital signing metadata. These two parts are encrypted with the same CEK. The `$<boundary>` is automatically generated and is different in each callback, e.g. `--ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9`.

- **API Description**

Name	Description
Service Full Name	Callback to Receive Bulk Digital Signing Result
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	1.0
Description of Service	This callback will provide the bulk digital signing result to Online Service.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	1440 minutes (1 day)	Online Service could treat the bulk digital signing as failed if it doesn't get digital signing result callback within 1 day.

- **Callback Parameters**

Parameter	Type	Presence	Description
sigMetada			
`businessID`	String	Required	`businessID` is a unique identifier for Online Service to differentiate different request. Online Service can use the `businessID` to relate the callback message with the original request.
`state`	String	Optional	If the `state` parameter has been present in the request message, the exact value of state will be returned. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service and it should be a secure random value.
`cert`	String	Optional	Base64-encoded DER format certificate for the "iAM Smart" user. "iAM Smart" System will provide this to Online Service only when the digital signing is successful.
`totalCallbacks`	Integer	Optional (Conditional)	If `maxCallbackSigs` (m) is less than the number of hashes and digests (n) when submitting request (Section 10.5.1), the digital signing results of these documents will be called back in `totalCallbacks = int((n-1) / m + 1)` times. `m` signatures will be returned in each of the first `totalCallbacks-1` callbacks. `n-(totalCallbacks-1)*m` signatures will be returned in the last callback. If `totalCallbacks` is greater than 1, it must be specified.

			The default value is 1.
`callbackSeq`	Integer	Optional (Conditional)	If `totalCallbacks` is greater than 1, this `callbackSeq` should be `1, 2, …, totalCallbacks`. If `totalCallbacks` is greater than 1, this `callbackSeq` must be specified. The default value is 1.
`totalSignNum`	Integer	Optional (Conditional)	The Number of signed documents. It should be equal to the number of documents submitted by the user.
`failSignNum`	Integer	Optional (Conditional)	The Number of signed failed documents. It should be less than the number of documents submitted by users.
sigFile (For Hash Signature)			
`id`	String	Required	The document unique identifier. It should be ASCII string with length less than or equal to 36 chars.
`hash`	String	Required	The document hash submitted by Online Service in the API request
`timestamp`	Long	Required (Conditional)	Timestamp in milliseconds since January 1, 1970 00:00:00 GMT. "iAM Smart" System will provide this to Online Service only when digital signing is successful.
`signature`	String	Required (Conditional)	Base64-encoded signature result string. "iAM Smart" System will provide this to Online Service only when digital signing is successful.
sigFile (For PDF Signature)			
`id`	String	Required	The document unique identifier. It should be ASCII string with length less than or equal to 36 chars.
`docDigest`	String	Required	The pdf document digest submitted by Online Service in the API request
`pdfSignature`	String	Required (Conditional)	Base64-encoded PKCS#7 object that is the actual PDF signature value. It contains signer's certificate, signed hash value, and the digital signing timestamp information. Online Service can embed this value to the

			PDF document for future verification. "iAM Smart" System will provide this to Online Service only when the digital signing is successful.

- **Example Callback (Signing Success)**

```
// Line breaks are for legibility only.
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>

// Callback Header
// The $<boundary> looks like --ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9
// It is automatically generated and is different in every callback
Content-Type: multipart/form-data;boundary=$<boundary>

// Callback Body
$<boundary>
Content-Disposition: form-data; name="sigFile"; filename="signatures.txt"
Content-Type: text/plain
Content-Length: 62892
// The content of the text file looks like the following.
// Line breaks are for legibility only.
"signatures": [
  {
    "id": "egh…cva",
    "hash": "tGzv3JOkF0XG5Qx2TlKWIA",
    "timestamp": 1556450176000,
    "signature": "nnoadisauflanefhykdjf"
  },
  ……
  {
    "id": "hrf…bnd",
    "docDigest": "tGzv3JOkF0XG5Qx2TlKWIA",
    "pdfSignature": "sdfGSDGsdfaGDEHfjslgGQG……GSGjljlkjwmh"
  }
]

$<boundary>
Content-Disposition: form-data; name="sigMetada"
Content-Type: application/json;charset=UTF-8
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
```

```
    "businessID": "2YotnFZFEjr1zCsicMWpAA",
    "state": "unesidkd",
    "cert": "sdfGSDGsdfaGDEHfjslgGQG……GSGjljlkjwmh",
    "totalCallbacks": 10,
    "callbackSeq": 2,
    "totalSignNum": 100,
    "failSignNum": 20

  }
}
$<boundary>
```

- **Example Callback (Signing Failure)**

```
// Line breaks are for legibility only.
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>

// Callback Header
// The $<boundary> looks like --ZJpuW5l0YYKRmTVwkt76oSIv9-pw8Cm0nW9
// It is automatically generated and is different in every callback
Content-Type: multipart/form-data;boundary=$<boundary>

// Callback Body
$<boundary>
Content-Disposition: form-data; name="sigMetada"
Content-Type: application/json;charset=UTF-8
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D71002",
  "message": "failed to request signing",
  "content": {
    "businessID": "2YotnFZFEjr1zCsicMWpAA",
    "state": "unesidkd",
  }
}
$<boundary>
```

10.5.8 Online Service Acknowledges Bulk Digital Signing Result

- **API Description**

Name	Description
Service Full Name	Online Service acknowledges bulk digital signing result

URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/bdss/ackResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online Service calls this API to acknowledge "iAM Smart" System if the result of the bulk digital signature is accepted or not. Online Service should send the acknowledgement regardless of whether the digital signing result (Section 10.5.8) is success or failure.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for Online Service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`signingResult`	String	Required	"SR001": digital signature is accepted "SR002": digital signature is rejected "SR003": no digital signature was received
`signatures`	Array	Required *(Conditional)*	When signingResult has no value, there must be values here
Details for "signatures"			
`id`	String	*Required*	The document unique identifier. It should be ASCII string with length less than or equal to 36 chars.
`signingResult`	String	*Required*	"SR001": digital signature is accepted "SR002": digital signature is rejected "SR003": no digital signature was received

- **Example Request**

```
// Line breaks are for legibility only.
POST
https://<iAM_Smart_domain>/api/v1/bdss/ackResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
```

```
// Request Body
{
  "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
  "signingResult": "SR001"
}
```

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS"
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {signingResult}"
}
or
{
  "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
  "signingResult": "",
  "signatures": :[
       {
          "id":"egh…cva",
          "signingResult":"SR001"
       },
       {
          "id":"egh…cva",
          "signingResult":"SR002"
       },
       {
          "id":"egh…cva",
          "signingResult":"SR003"
       }
```

```
   ]
}
```

10.5.9 Enquire Bulk Digital Signing Status

- **API Description**

Name	Description
Service Full Name	Enquire Bulk Digital Signing Status
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/bdss/enquireStatus`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online Service calls this API to enquire the digital signing status of the submitted bulk digital signing request.

- **Request Parameters**

Parameter	Type	Presence	Description
`BSQCToken`	String	Required	The BSQC Token which is returned in section 10.5.6. Maximum Length: 32
`openID`	String	Required	Tokenised ID value Maximum Length: 64

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/bdss/enquireStatus
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Request Body
{
  "BSQCToken": "nnoadisauflanefhykdjf…f83h96e",
```

```
  "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
status	String	Required	The digital signing status of the submitted bulk digital signing request. Its value should be: “SS001”: Pending for Signing “SS002”: Signing in Progress “SS003”: Signing Completed “SS004”: Signing Cancelled “SS005”: Signing Failed “SS006”: No Record Found

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "status": "SS003"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {BSQCToken}"
}
```

10.5.10 Cancel Bulk Digital Signing Request

- **API Description**

Name	Description

Service Full Name	Online Service acknowledges digital signing result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/bdss/ackResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online Service calls this API to cancel the submitted bulk digital signing request. Only the request of which status is "Pending for Signing" can be cancelled.

- **Request Parameters**

Parameter	Type	Presence	Description
`BSQCToken`	String	Required	The BSQC Token which is returned in section 10.5.6. Maximum Length: 32
`openID`	String	Required	Tokenised ID value Maximum Length: 64

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/bdss/cancelRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Request Body
{
  "BSQCToken": "nnoadisauflanefhykdjf…f83h96e",
  "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
```

```
  "code": "D00000",
  "message": "SUCCESS"
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D71007",
  "message": "the request to be cancelled is not Pending for Signing"
}
```

10.5.11 Open “iAM Smart” Mobile App for Anonymous Bulk Digital Signing (appv2)

- **Using URL Scheme**

Name	Description
Service Full Name	Open “iAM Smart” Mobile App for Anonymous Bulk Digital Signing
URI (as in RESTFUL API)	`<"iAM Smart" app URL scheme>://v2_anon_bulk-sign`
Service Version	V2.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific Anonymous Bulk Digital Signing in “iAM Smart” Mobile App. The URL schemes supported on iOS and Android versions of “iAM Smart” Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required	`TicketID` is a unique identifier provided by “iAM Smart”, Online service retrieve `ticketID` while requesting the context. It is ASCII string with length less than or equal 36 chars.
`source`	String	Required	App_Link (iOS) or App_Package (Android)
`clientRedirectURI`	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (“,”) are not accepted.
`packageName`	String	Required (Android only)	Package name of the Online Service App for callback use.
`activityClass`	String	Required (Android only)	Activity class name of the Online Service App for callback use.
`activityParams`	String	Optional (Android only)	Optional params used during callback.
`state`	String	Required	If `state` parameter is presented in the

		(Conditional)	request message, the same state value will be returned to Online Service during callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphen are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://v2_anon_bulk-sign
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
&source=App_Package
&packageName=com.onlineservice.myapp
&activityClass=callback_activity
&activityParams=callback_param
```

11.STEP-UP AUTHENTICATION WITH SERVICE LOGIN

11.1 Overview

Online service can request “Step-up Authentication” API to ask the same “iAM Smart” user to authenticate again with his/her digital identity using NFC and/or Facial Recognition (FR) approaches as long as user has logged in to the online service via “iAM Smart” and online service still holds a valid accessToken. The NFC approach verifies that user does hold his/her own HKIC. The FR approach will use device’s front camera to take facial image. The image will be matched against the record in ImmD. The “Step-up Authentication” API is provided to the online service that required user confirmation in “iAM Smart” Mobile App.

11.2 Prerequisite

- Online service must undergo the authentication process (i.e. Request accessToken & Tokenised ID) in Section 3 in order to obtain the accessToken and openID as the input of Step-up Authentication API.
- Online service shall refer to “Reference System Flow and User Interface Specifications for “iAM Smart” Adoption” for the detail UI/UX requirements.
- Online Service must submit the common parameter of "rateLimitFactor" while requesting the Step-up Authentication function.

11.3 Scope

Online service shall submit the application for using the following scope values to “iAM Smart” Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_sua`	Scope for Step-up Authentication

11.4 Use Cases and Scenarios

11.4.1 Step-up Authentication (Online Service Website/App in Different Device)

The sequence diagram below shows how online service performs "iAM Smart" Step-up Authentication when online service and the "iAM Smart" Mobile App are running in different devices.

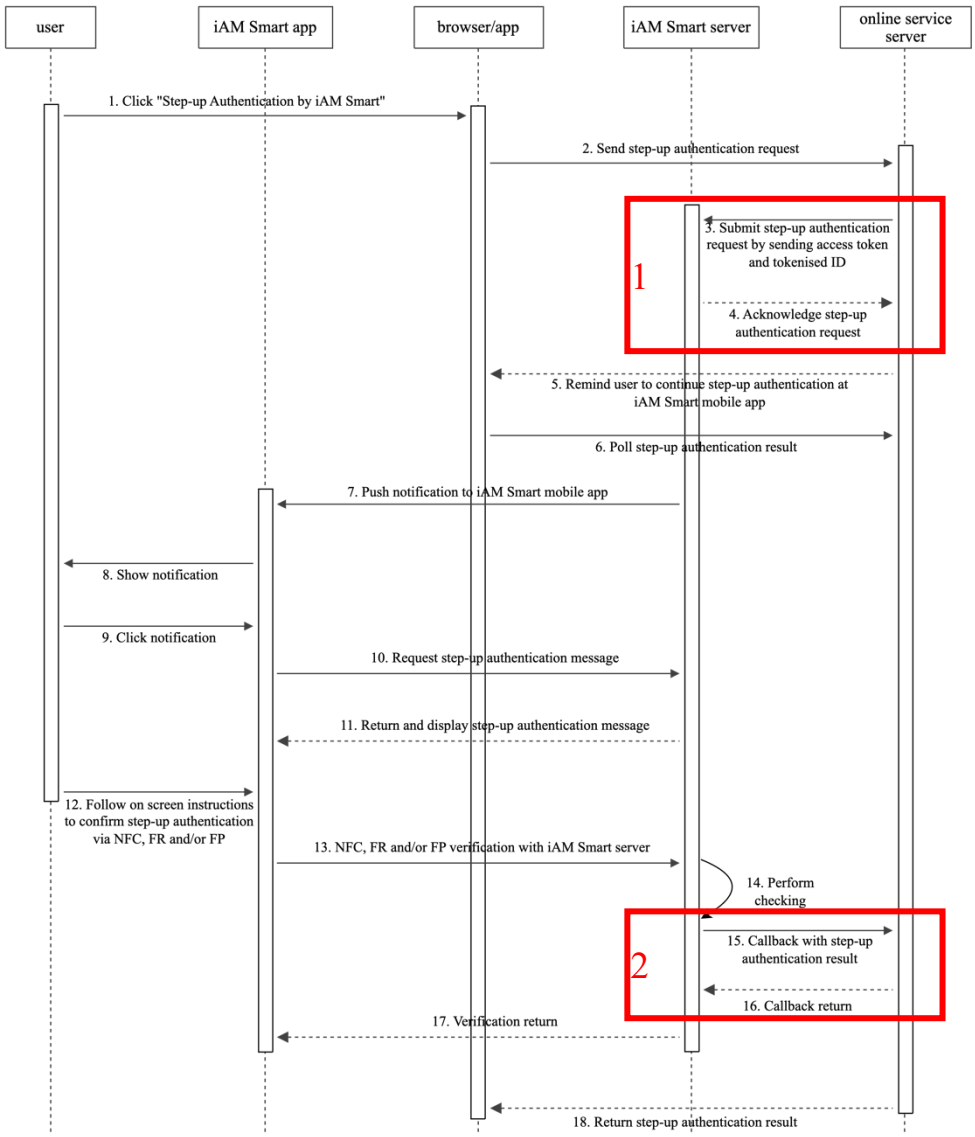

Figure-41 Step-up Authentication (Online Service Website/App in Different Device)

APIs interactions between Online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Request Step-up Authentication (appv1)	11.5.1
2	Callback to Receive Step-up Authentication Result	11.5.3

11.4.2 Step-up Authentication (Online Service Website in Same Device)

The sequence diagram below shows how online service performs "iAM Smart" Step-up Authentication when the online service website and the "iAM Smart" Mobile App are running in the same device.

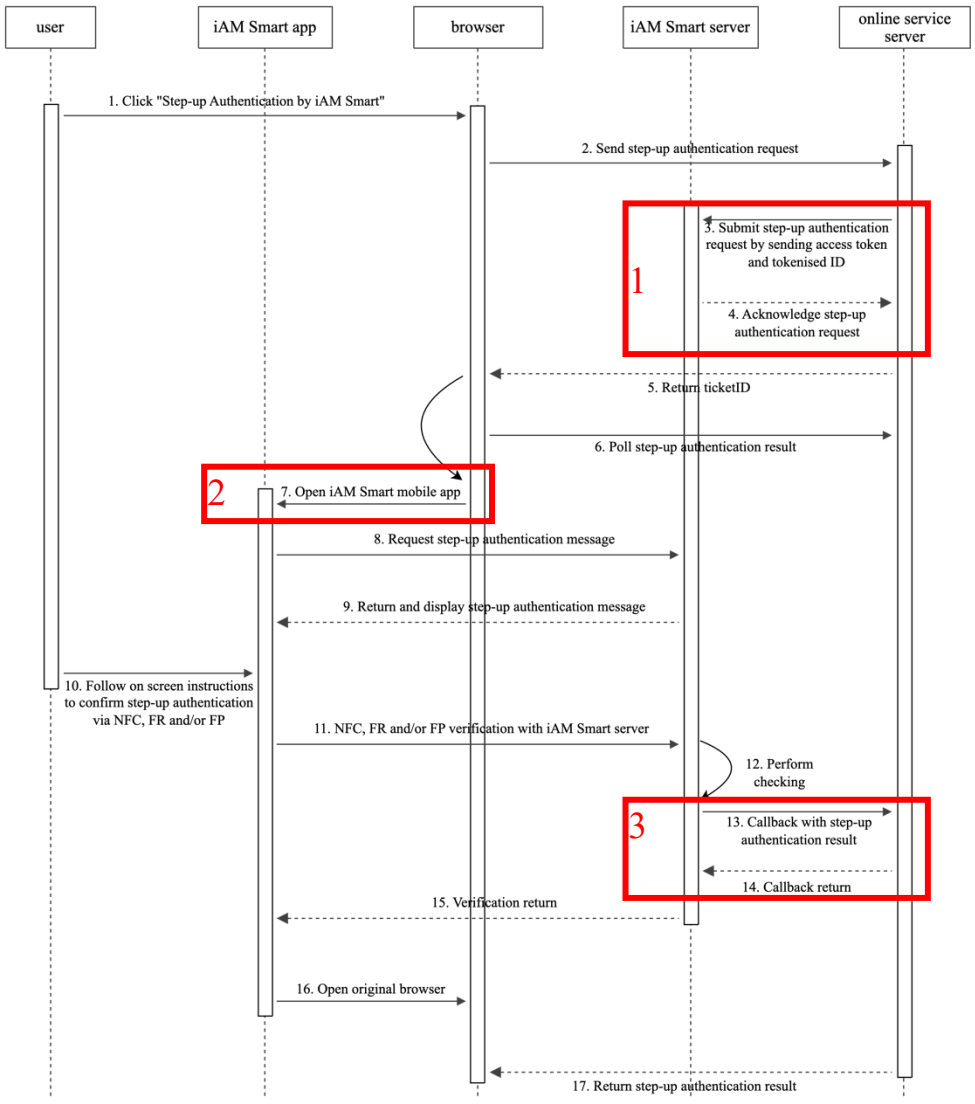

Figure-42 Step-up Authentication (Online Service Website in Same Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Request Step-up Authentication (appv1)	11.5.1
2	Open the “iAM Smart” Mobile App for Step-up Authentication	11.5.2
3	Callback to Receive Step-up Authentication Result	11.5.3

11.4.3 Step-up Authentication (Online Service App in Same Device)

The sequence diagram below shows how online service performs "iAM Smart" Step-up Authentication when the online service mobile application and the "iAM Smart" Mobile App are running in the same device.

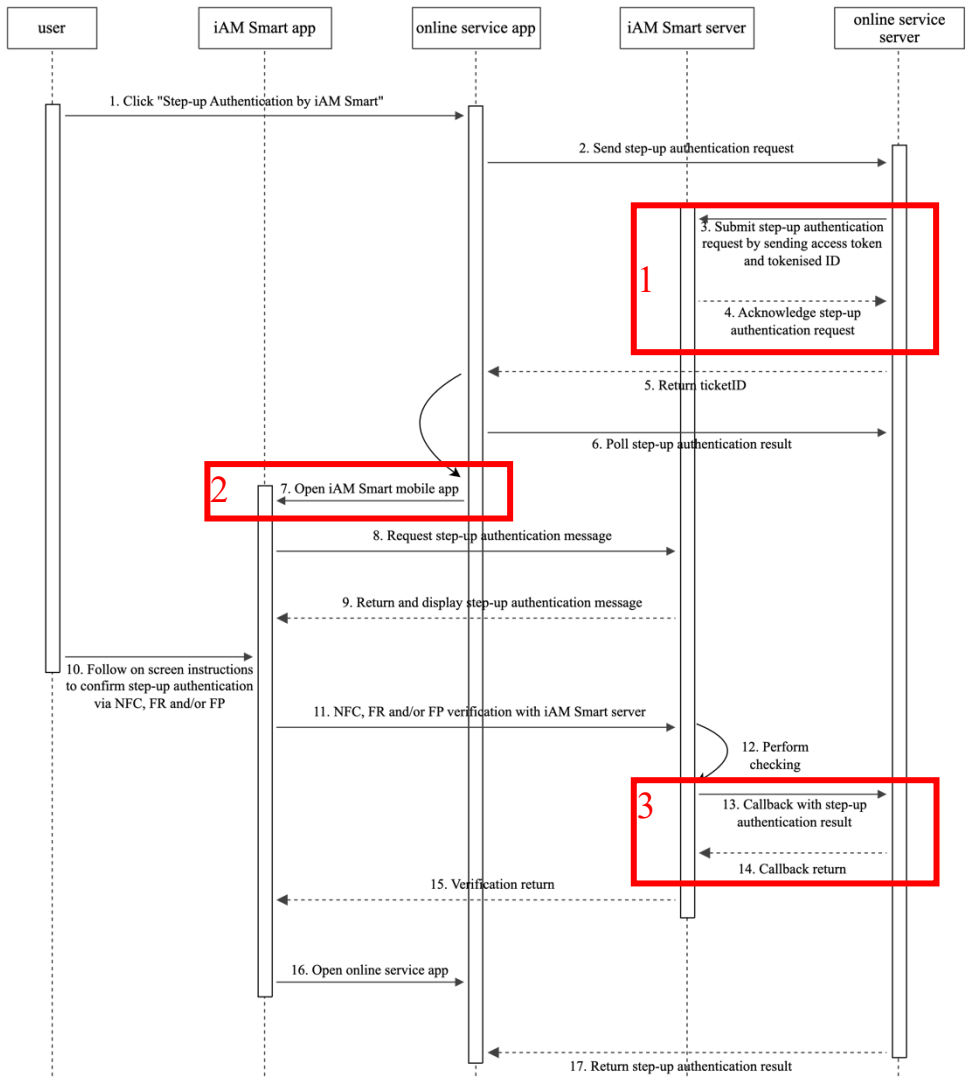

Figure-43 Step-up Authentication (Online Service App on Same Device)

APIs interactions between Online Service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Request Step-up Authentication (appv1)	11.5.1
	Request Step-up Authentication (appv2)	11.5.4

2	Open the “iAM Smart” Mobile App for Step-up Authentication	11.5.2
3	Callback to Receive Step-up Authentication Result	11.5.3

11.5 API Implementation Details

11.5.1 Request Step-up Authentication (appv1)

- **API Description**

Name	Description
Service Full Name	Request Step-up Authentication
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/account/sua/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to request step-up authentication by sending `accessToken` and `openID` to the “iAM Smart” System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different requests. It should be an ASCII string with a length of less than or equal to 36 chars.
`accessToken`	String	Required	accessToken value
`openID`	String	Required	Tokenised ID value Maximum Length: 64
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification. For Mobile App: App Link (iOS) or App Package (Android)
`redirectURI`	String	Required	callback URI.
`state`	String	Optional	If the `state` parameter is presented in the request message, the same state value will be returned to online service during the

			callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.
`suaMethod`	String (JSON)	Required	`{"unary": ["FR"]}, {"unary": ["NFC"]}, {"and": ["FR", "NFC"]}`. Online service should have been granted the requested step-up authentication method(s) in the e-Service Management Portal to use the `"NFC", "FR"` methods to verify the user. The API returns unsuccessful immediately if the first method fails.
`suaMessage`	String	Optional	Service message from online service to be displayed in iAM Smart before proceed Step-up Authentication. This only applies when the `suaMethod` is specified with one of the above values. `suaMessage` is a specified message code representing registered messages, it should be an ASCII string with a length of less than or equal to 10 chars.
`suaWaitPeriod`	Integer	Optional	This waiting period defines the minimum number of hours since the user account was created. If system defined waiting period is greater than `suaWaitPeriod`, greater value of system defined waiting period would be used. If `suaWaitPeriod` is greater than system defined waiting period, greater value of `suaWaitPeriod` would be used. System default value is zero. Maximum Value: 720

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/account/sua/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
 "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
 "source": "App_Package",
 "redirectURI": "https://<rp_domain>/<rp_context>/<call_back_endpoint>",
 "state": "eddd527b6",
 "suaMethod": "{\"and\": [\"FR\", \"NFC\"]}",
 "suaMessage": "0001",
 "suaWaitPeriod": 10
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If the user completes authentication by scanning the QR code (i.e. different device), then "true" will be returned to online service. `authByQR` should always be `false` in appv2 API.
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier provided by “iAM Smart” System,

			online service retrieve `ticketID` while requesting the step-up authentication function. It is ASCII string with length less than or equal 36 chars. The value valid for 18 minutes.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": false,
    "ticketID": "bbb8aae57c104cda40c93843ad5e6db8"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {openID}"
}
```

11.5.2 Open the "iAM Smart" Mobile App for Step-up Authentication (appv1 and appv2)

- **Using URL Scheme**

Name	Description
Service Full Name	Open the "iAM Smart" Mobile App for Step-up Authentication
URI (as in RESTFUL API)	For Step-up Authentication v1: <"iAM Smart" app URL scheme>://sua For Step-up Authentication v2: <"iAM Smart" app URL scheme>://v2_sua
Service Version	V1.0.0 and V2.0.0
Description of Service	The URL scheme uses deep linking to redirect users to step-up authentication in the "iAM Smart" Mobile App. The URL schemes are supported on iOS and Android versions of the "iAM Smart" Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
ticketID	String	Required	ticketID is a unique identifier provided by the "iAM Smart" System. Online service retrieves ticketID while requesting the Step-up Authentication function. It is an ASCII string with a length of less than or equal to 36 chars.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://v2_sua
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

11.5.3 Callback to Receive Step-up Authentication Result

- **API Description**

Name	Description
Service Full Name	Callback to Receive Step-up Authentication Result
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	V1.0.0
Description of Service	This callback returns the Step-up Authentication result to online service upon user consent. Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different requests. Online services can use the `businessID` to relate the callback message with the original request. Maximum Length: 36
`state`	String	Optional	The same state value will be returned if the state parameter is in the request message. It is used to prevent the CSRF attack. The value of the state is defined by Online Service, and it should be a secure random value. Maximum Length: 36
`isPassed`	Boolean	Required	Step-up Authentication result. `true` - same person, otherwise `false`.

- **Example Callback**

```
// Line breaks are for legibility only.
```

```
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>
// Callback Body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
   "businessID": "3e47be25-66a6-43fb-89f6-7e2dd138aff8",
   "state": "unesidkd",
   "isPassed": true
 }
}
```

11.5.4 Request Step-up Authentication (appv2)

- **API Description**

Name	Description
Service Full Name	Request Step-up Authentication
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/appv2/account/sua/initiateRequestV2`
Request Type	POST
Service Version	V2.0.0
Description of Service	Online service can use this API to request step-up authentication by sending `accessToken` and `openID` to the "iAM Smart" System.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different requests. It should be an ASCII string with a length of less than or equal to 36 chars.
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value Maximum Length: 64
`source`	String	Required	Request initiator. The supported source values can be found in Appendix

			B of this specification. For Mobile App: App Link (iOS) or App Package (Android)
`clientRedirectURI`	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (“,”) are not accepted.
`packageName`	String	Required (Android only)	Package name of the Online Service App for callback use and registered in the self-service portal and the application form.
`activityClass`	String	Required (Android only)	Activity class name of the Online Service App for callback use.
`activityParams`	String	Optional (Android only)	Optional params used during callback.
`serverRedirectURI`	String	Required	Callback URI.
`state`	String	Optional	If the `state` parameter is presented in the request message, the same state value will be returned to online service during the callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.
`suaMethod`	String (JSON)	Required	`{"unary": ["FR"]}, {"unary": ["NFC"]}, {"and": ["FR", "NFC"]}`. Online service should have been granted the requested step-up authentication method(s) in the e-Service Management Portal to use the `"NFC", "FR"` methods to verify the

			user. The API returns unsuccessful immediately if the first method fails.
suaMessage	String	Optional	Service message from online service to be displayed in iAM Smart before proceed Step-up Authentication. This only applies when the suaMethod is specified with one of the above values. suaMessage is a specified message code representing registered messages, it should be an ASCII string with a length of less than or equal to 10 chars.
suaWaitPeriod	Integer	Optional	This waiting period defines the minimum number of hours since the user account was created. If system defined waiting period is greater than suaWaitPeriod, greater value of system defined waiting period would be used. If suaWaitPeriod is greater than system defined waiting period, greater value of suaWaitPeriod would be used. System default value is zero. Maximum Value: 720

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/appv2/account/sua/initiateRequestV2
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
```

```
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
 "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
 "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
 "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
 "source": "App_Package",
 "packageName": "com.onlineservice.myapp",
 "activityClass": "callback_activity",
 "activityParams": "callback_param",
 "serverRedirectURI":
"https://<rp_domain>/<rp_context>/<call_back_endpoint>",
 "state": "eddd527b6",
 "suaMethod": "{\"and\": [\"FR\", \"NFC\"]}",
 "suaMessage": "0001",
 "suaWaitPeriod": 10
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`authByQR`	Boolean	Required	If the user completes authentication by scanning the QR code (i.e. different device), then "true" will be returned to online service. `authByQR` should always be `false` in appv2 API.
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier provided by "iAM Smart" System, online service retrieve `ticketID` while requesting the step-up authentication function. It is ASCII string with length less than or equal 36 chars. The value valid for 18 minutes.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
```

```
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": false,
    "ticketID": "bbb8aae57c104cda40c93843ad5e6db8"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {openID}"
}
```

12. STEP-UP AUTHENTICATION WITHOUT SERVICE LOGIN (AKA ANONYMOUS STEP-UP AUTHENTICATION)

12.1 Overview

Similar to Step-up Authentication API, online service can request “Anonymous Step-up Authentication” API to ask “iAM Smart” user to do step-up authentication with his/her digital identity using NFC and/or Facial Recognition (FR). The NFC approach verifies that user does hold his/her own HKIC. The FR approach will use device’s front camera to take facial image. The image will be matched against the record in ImmD. Unlike the accessToken obtained from Authentication API, the accessToken received in this API can only be used once.

12.2 Prerequisite

- Online service shall refer to “Reference System Flow and User Interface Specifications for “iAM Smart” Adoption” for the detail UI/UX requirements.
- Online Service must submit the common parameter of "rateLimitFactor" while requesting the Step-up Authentication function.

12.3 Scope

Online service shall submit the application for using the following scope values to “iAM Smart” Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_sua`	Scope for Step-up Authentication

12.4 Use Cases and Scenarios

12.4.1 Anonymous Step-up Authentication (Online Service Website in Different Device)

The sequence diagram below shows how an anonymous user performs step-up authentication when online service website and the "iAM Smart" Mobile App are running in different devices.

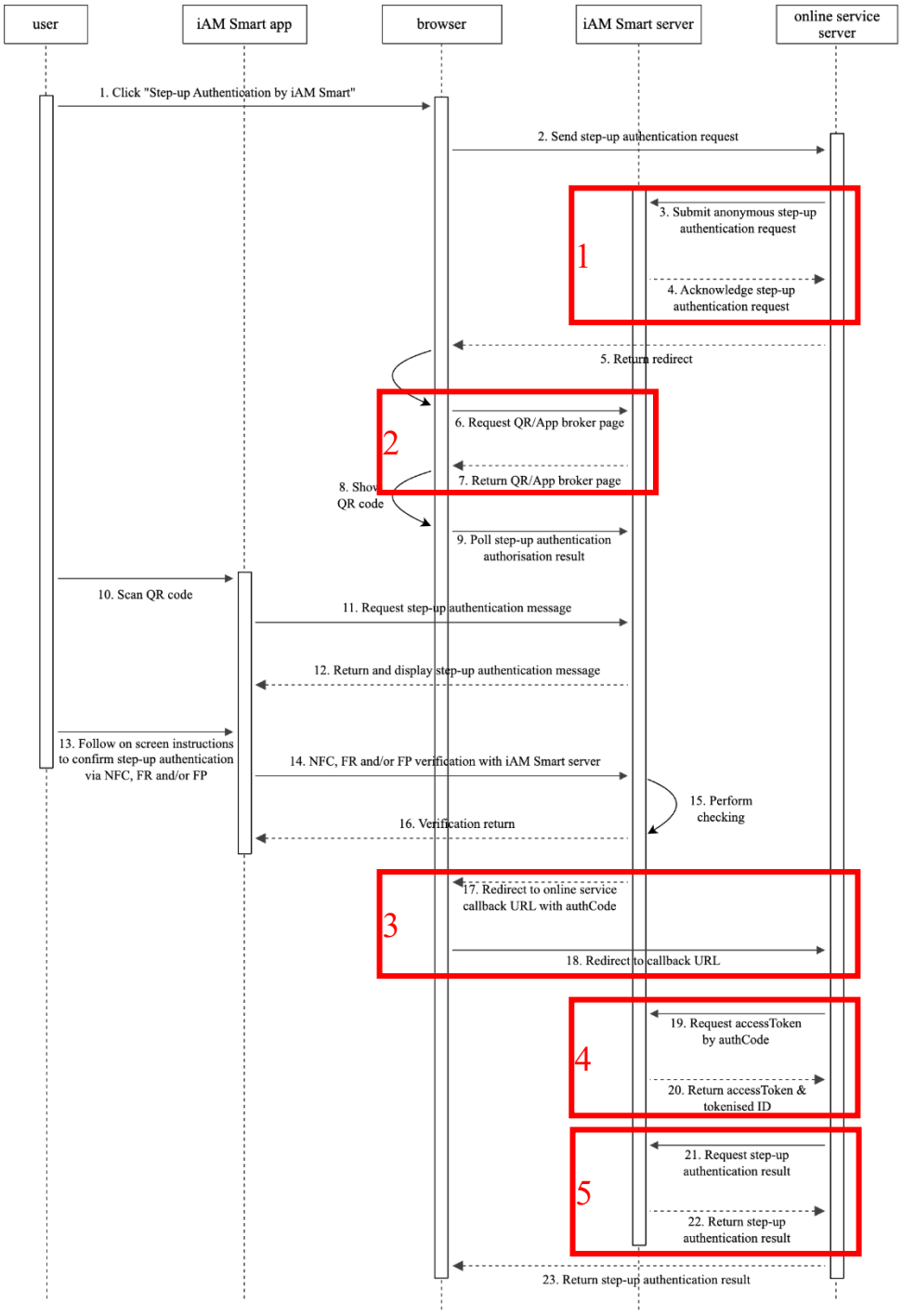

Figure-44 Anonymous Step-up Authentication (Online Service Website in Different Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Step-up Authentication	12.5.1
2	Request QR Page	12.5.2
3	Callback with authCode to Online Service Server	12.5.5
4	Request accessToken & Tokenised ID	12.5.6
5	Obtain Anonymous Step-up Authentication Result	12.5.7

12.4.2 Anonymous Step-up Authentication (Online Service Website in Same Device)

The sequence diagram below shows how an anonymous user performs step-up authentication when online service website and the "iAM Smart" Mobile App are running in the same device.

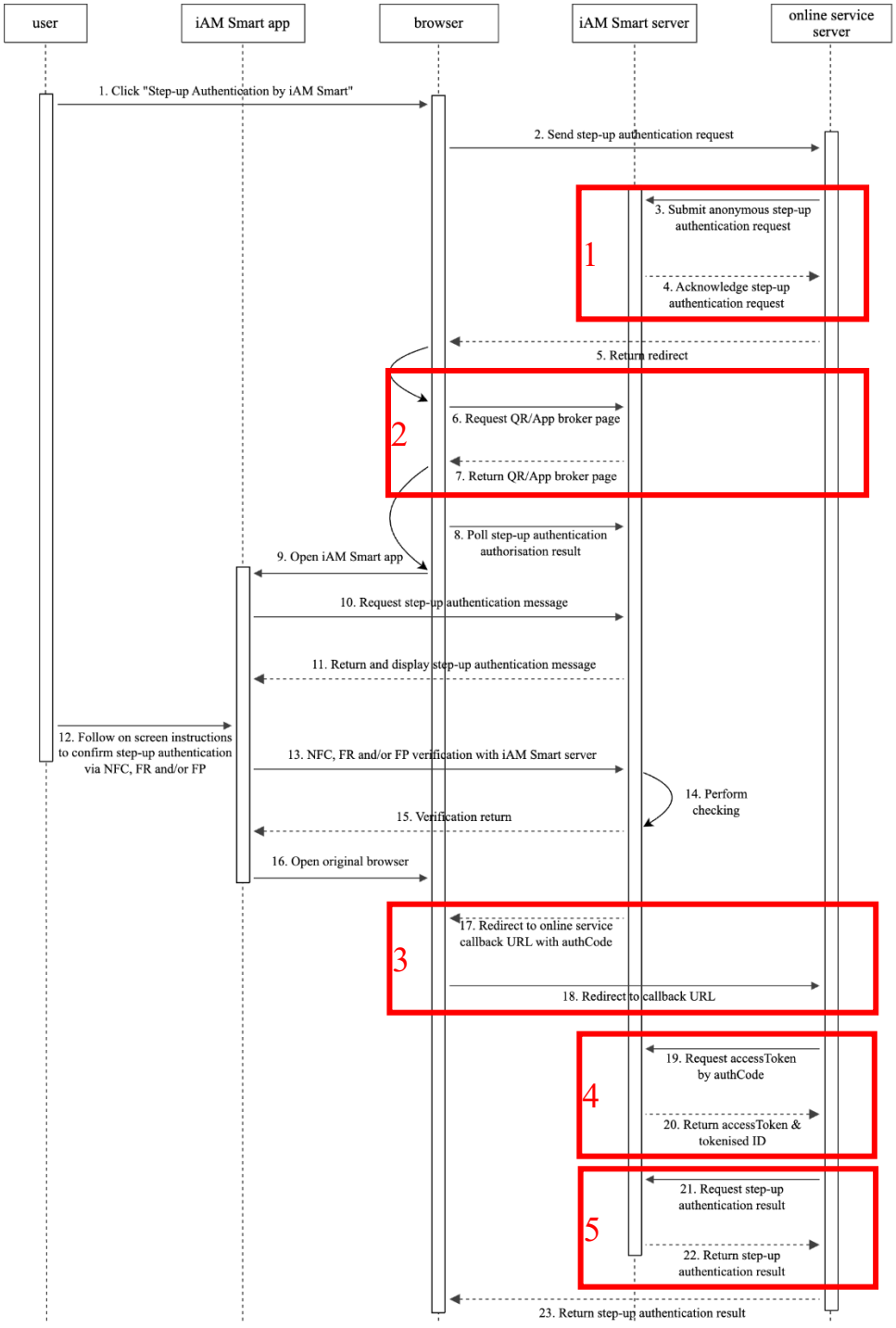

Figure-45 Anonymous Step-up Authentication (Online Service Website in Same Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Step-up Authentication	12.5.1
2	Request QR Page	12.5.2
3	Callback with authCode to Online Service Server	12.5.5
4	Request accessToken & Tokenised ID	12.5.6
5	Obtain Anonymous Step-up Authentication Result	12.5.7

12.4.3 Anonymous Step-up Authentication (Online Service App in Different Device)

The sequence diagram below shows how an anonymous user performs step-up authentication when online service App and the "iAM Smart" Mobile App are running in different devices.

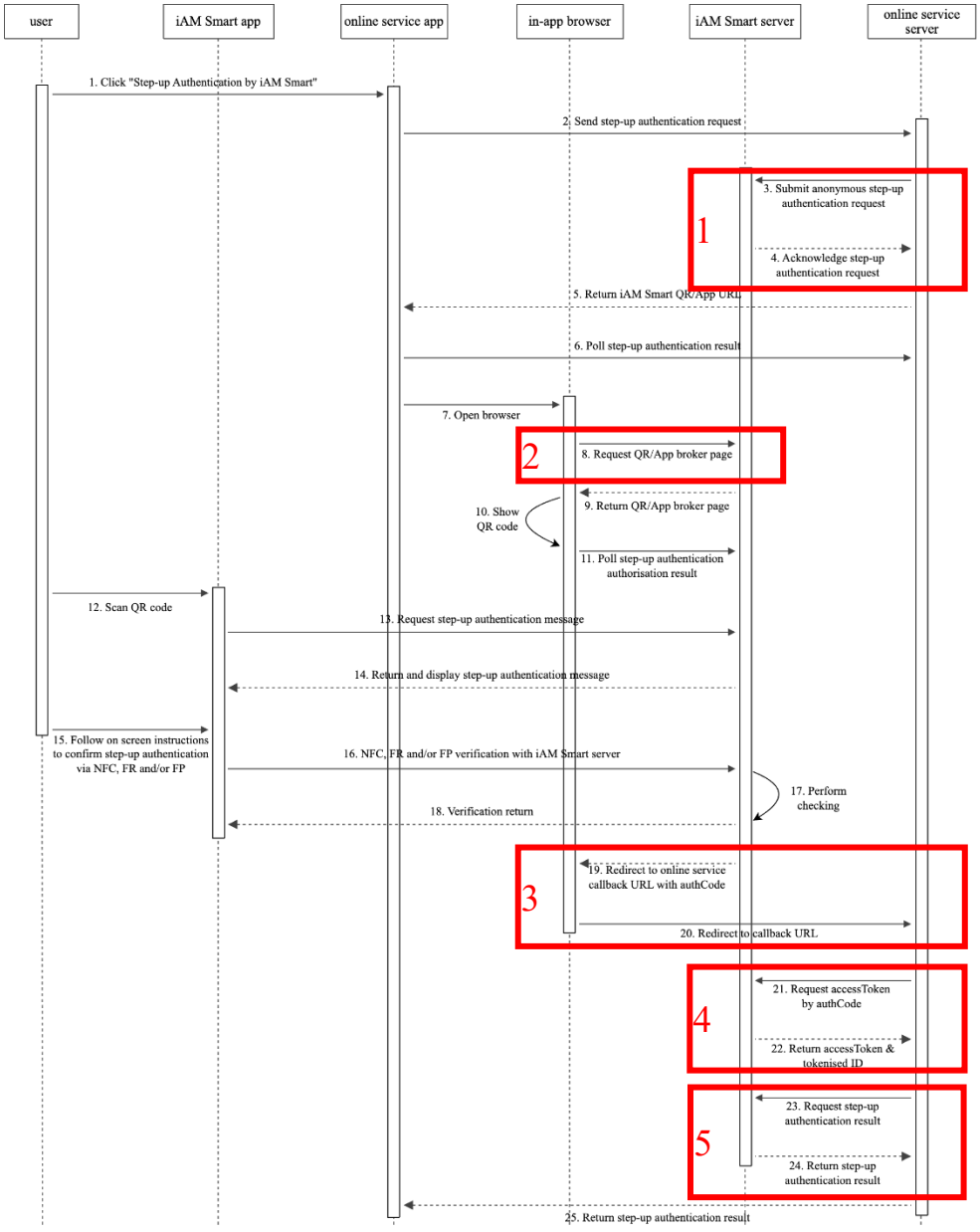

Figure-46 Anonymous Step-up Authentication (Online Service App in Different Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Step-up Authentication	12.5.1
2	Request QR Page	12.5.2
3	Callback with authCode to Online Service Server	12.5.5
4	Request accessToken & Tokenised ID	12.5.6
5	Obtain Anonymous Step-up Authentication Result	12.5.7

12.4.4 Anonymous Step-up Authentication (Online Service App in Same Device)

The sequence diagram below shows how an anonymous user performs step-up authentication when online service App and the "iAM Smart" Mobile App are running in the same device.

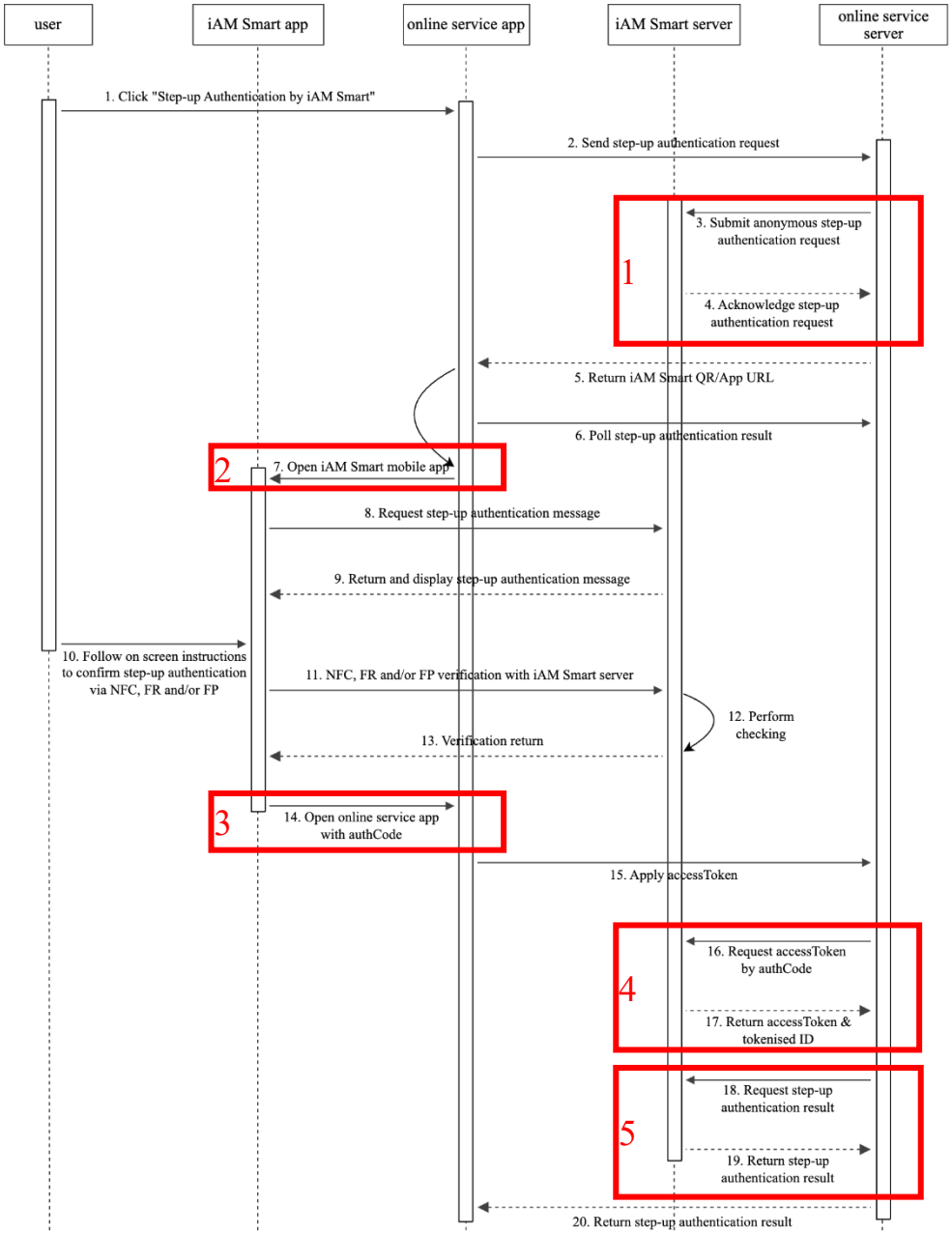

Figure-47 Anonymous Step-up Authenticatioin (Online Service App in Same Device)

APIs interactions between online service servers and “iAM Smart” System are marked in red. Technical details of respective APIs can be found at the following sections.

No.	API Name	API Reference (Section)
1	Request Anonymous Step-up Authentication	12.5.1
2	Open “iAM Smart” Mobile App for Anonymous Step-up Authentication (appv1)	12.5.3
	Open “iAM Smart” Mobile App for Anonymous Step-up Authentication (appv2)	12.5.8
3	Callback with authCode to Online Service App	12.5.4
4	Request accessToken & Tokenised ID	12.5.6
5	Obtain Anonymous Step-up Authentication Result	12.5.7

12.5 API Implementation Details

12.5.1 Request Anonymous Step-up Authentication

- **API Description**

Name	Description
Service Full Name	Request Anonymous Step-up Authentication
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/anonymous/sua/initiateRequest`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to initiate anonymous step-up authentication request.

- **Request Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
`HKICHash`	String	Required	Only when the HKIC hash of the "iAM Smart" user who authorises/confirms the anonymous step-up authentication workflow matches the HKICHash of this request, will the workflow be processed. Online service should convert the HKIC number into hash value using SHA256 before sending to "iAM Smart" System. Only the identifier of HKIC number will be hashed, no check digit is needed e.g. A123456. The value should be Base64 encoded.
`suaMethod`	String (JSON)	Required	`{"unary": ["FR"]}, {"unary": ["NFC"]}, {"and": ["FR", "NFC"]}`. Online service should have been granted the requested step-up authentication method(s) in the e-Service

			Management Portal to use the "NFC", "FR" methods to verify the user. The API returns unsuccessful immediately if the first method fails.
suaMessage	String	Optional	Service message from online service to be displayed in iAM Smart before proceed Step-up Authentication. This only applies when the suaMethod is specified with one of the above values. suaMessage is a specified message code representing registered messages, it should be an ASCII string with a length of less than or equal to 10 chars.
identifierCode	String	Required (Conditional)	Online service should pass the identifierCode string to the "iAM Smart" system in the different device workflows. The "iAM Smart" system will re-display the passed identifierCode in the "iAM Smart" mobile app. Maximum Length: 10
suaWaitPeriod	Integer	Optional	This waiting period defines the minimum number of hours since the user account was created. If system defined waiting period is greater than suaWaitPeriod, greater value of system defined waiting period would be used. If suaWaitPeriod is greater than system defined waiting period, greater value of suaWaitPeriod would be used. System default value is zero. Maximum Value: 720

- **Example Request**

```
// Line breaks are for legibility only.
```

```
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/anonymous/sua/initiateRequest
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
  "businessID": "bbb8aae57c104cda40c93843ad5e6db8",
  "HKICHash": "6ea37e641260714009a880a8057032a414009a80f667efc",
  "suaMethod": "{\"and\": [\"FR\", \"NFC\"]}",
  "suaMessage":"0001",
  "suaWaitPeriod": 10,
  "identifierCode": "1234"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required (Conditional)	`ticketID` is a unique identifier that will be used to identify this request in the following steps of the Anonymous Step-up Authentication workflow. The ticketID will be expired 12 minutes after issuance.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "ticketID": "561b6f1ce5a043ba8aae33fb5c50386d"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in  Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {businessID}"
}
```

12.5.2 Request QR Page

- **API Description**

Name	Description
Service Full Name	Request QR Page
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/auth/getQR`
Request Type	GET
Service Version	1.0.0
Description of Service	Online service calls this API to get the QR/App broker page or QR page. After the user authorises login or anonymous request on the "iAM Smart" Mobile App, the page will be redirected to the `redirectURI` with `authCode` and `state` parameters. If the user denies it, only the `state` parameter will be redirected.

- **Request Parameters**

Parameter	Type	Presence	Description
`clientID`	String	Required	Online service client identifier. The clientID will be assigned to online service at the initial

			registration.
`responseType`	String	Required	The value MUST be set to `code`.
`source`	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.
`redirectURI`	String	Required	The callback redirectURI. The value should be URL encoded and registered in the self-service portal.
`scope`	String	Required	The value should be URL encoded. Specify the scope of "iAM Smart" functions: - `eidapi_sua` The value of the scope parameter is expressed as a list of space-delimited, case-sensitive strings.
`ticketID`	String	Required	Required in anonymous step-up authentication workflows for getting QR page.
`lang`	String	Optional	Language to display: en-US, zh-HK, or zh-CN. If this parameter is not specified, zh-HK will be shown.
`state`	String	Required	If the `state` parameter is presented in the request message, the same state value will be returned to online service during the callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.
`brokerPage`	Boolean	Optional	If `brokerPage` is set to `true`, Universal Link (iOS) / App Link (Android) will be leveraged to open "iAM Smart" Mobile App. This feature is useful for Online Services that support mobile web versions while triggering the "iAM Smart" Mobile App or showing a QR page automatically. (i.e. Show QR page without detecting whether "iAM Smart" Mobile App is installed). The default value is `false`.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
GET
https://<iAM_Smart_domain>/api/v1/auth/getQR
?clientID=Online Service1
&responseType=code
&source=Android_Chrome
&redirectURI=https%3A%2F%2Frp_domain%2Frp_context%2Fcall_back_endpoint
&scope=eidapi_sua
&lang=en-US
&state=eb9b7b8eddd5
```

- **Response Parameters**

N/A

12.5.3 Open "iAM Smart" Mobile App for Anonymous Step-up Authentication (appv1)

- **Using URL Scheme**

Name	Description
Service Full Name	Open "iAM Smart" Mobile App for Anonymous Step-up Authentication
URI (as in RESTFUL API)	<"iAM Smart" app URL scheme>://anon_sua
Service Version	V1.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific Step-up Authentication page in "iAM Smart" Mobile App. The URL schemes are supported on iOS and Android versions of "iAM Smart" Mobile App. *Remark: appv1 does not allow one client id to support multiple app.*

- **Request Parameters**

Parameter	Type	Presence	Description
ticketID	String	Required	TicketID is a unique identifier provided by "iAM Smart" System. Online service retrieves ticketID while requesting the

			Anonymous Step-up Authentication function. It is an ASCII string with a length of less than or equal to 36 chars.
source	String	Required	App_Link/ App_Scheme Please use App_Link unless the support team approves.
redirectURI	String	Required	Callback redirect URI. The value should be **URL encoded** and **registered** in the self-service portal and application form.
state	String	Required	If the state parameter is presented in the request message, the same state value will be returned to Online Service during the callback. It is used to prevent the CSRF attack. The value of state is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://anon_sua
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
&source=App_Link
&redirectURI=https%3A%2F%2Frp_domain%2Frp_context%2Fcall_back_endpoint
&state=eb9b7b8eddd5
```

12.5.4 Callback with authCode to Online Service App

- **URL Scheme and Package Name**

Name	Description
Service Full Name	Callback with authCode to Online Service App
URL Scheme (iOS and Android)	<Universal / App Link>. The online service custom app scheme or Universal / App Link should be registered with the "iAM Smart" System during onboarding.
Package name (Android only)	<package name> and <activity name>. The online service package name must be registered in the self-service portal.

	Remark: Direct Login v2 (App) and appv2 API adopts package name verification instead of App Link
Description of Service	For appv1 API, the Online Service App will be invoked and launched by Universal / App Link. It makes use of deep linking to redirect users to the Online Services app. App Link can only work for mobile devices with Google Mobile Services (GMS). For appv2 API, the Android Online Service App will be invoked and launched by the package name. It make use of intent to redirect users to the Online Service app. The iOS Online Service App will be invoked and launched via Universal link. The Universal / App Link with the landing location as well as the package name plus activity name must be registered in the self-service portal and enabled by the support team.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	12 minutes	Online Service could treat the request as failed if it doesn't get callback within 12 minutes

- **URL Scheme Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required (Conditional)	`businessID` will only be returned in anonymous workflows. `businessID` is a unique identifier for Online Service to differentiate different requests. Online service can use the `businessID` to relate the callback message with the original request.
`code`	String	Required (Conditional)	The authorisation code is generated by the "iAM Smart" System. The authorisation code will be expired 1 minute after issuance. Online Service MUST NOT use the authorisation code more than once. An error message will be

			returned if an authorisation code is expired or re-used.
`state`	String	Optional	If the `state` parameter has been presented in the request message, the same state value will be returned. It is used to prevent the CSRF attack. The value of the state is defined by Online Service, and it should be a secure random value.
`error_code`	String	Required (Conditional)	Return error code when exception.

- **Example URL Scheme**

Allow

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
?businessID=b2c99aa83b0049e9ba370c5341681225
&code=0ad186353c424c64897fcc00445c9ba1
&state=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
<Universal / App Link>://<landing location>
?error_code=D20001
&state=eddd527b6
```

- **Example Package Name**

Allow

```
Intent ii=new Intent(<package name>, <activity class name>);
ii.putExtra("code", "0ad186353c424c64897fcc00445c9ba1");
ii.putExtra("state", "eddd527b6");
ii.putExtra("businessID", "b2c99aa83b0049e9ba370c5341681225");
startActivity(ii);
```

Deny

```
Intent ii=new Intent(<package name>, <activity name>);
ii.putExtra("error_code", "D20001");
ii.putExtra("state", "eddd527b6");
startActivity(ii);
```

12.5.5 Callback with authCode to Online Service Server

- **API Description**

Name	Description
Service Full Name	Callback with authCode to Online Service Server
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	GET
Service Version	V1.0.0
Description of Service	This callback is used to pass `authCode` to online service Server. The URI must be registered in the self-service portal.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	12 minutes	Online service could treat the request as failed if it doesn't get callback within 12 minutes.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required (Conditional)	`businessID` will only be returned in anonymous workflows. `businessID` is a unique identifier for online service to differentiate different requests. Online service can use the `businessID` to relate the callback message with the original request.
`code`	String	Required (Conditional)	The authorisation code is generated by the "iAM Smart" System. The authorisation code will be expired 1 minute after issuance. Online service MUST NOT use the authorisation code more than once. An error message will be returned if an authorisation code is expired or re-used.

state	String	Optional	If the state parameter has been presented in the request, the same state value will be returned during the callback. It is used to prevent the CSRF attack. The value of the state is defined by Online Service, and it should be a secure random value.
error_code	String	Required (Conditional)	Return error code when exception.

- **Example Callback**

Allow

```
// Line breaks are for legibility only.
GET
https://<call_back_endpoint>
?businessID=b2c99aa83b0049e9ba370c5341681225
&code=0ad186353c424c64897fcc00445c9ba1
&state=eddd527b6
```

Deny

```
// Line breaks are for legibility only.
GET
https://<call_back_endpoint>
?error_code=D20001
&state=eddd527b6
```

12.5.6 Request accessToken & Tokenised ID

- **API Description**

Name	Description
Service Full Name	Request access token and tokenised ID with authCode
URI (as in RESTFUL API)	https://<iAM_Smart_domain>/api/v1/auth/getToken
Request Type	POST
Service Version	1.0.0
Description of Service	Online service uses this API to retrieve the access token and Tokenised ID (openID). An authorisation code is necessary during the process. The accessToken and openID will be used to call

	corresponding “iAM Smart” services subsequently.

- **Request Parameters**

Parameter	Type	Presence	Description
code	String	Required	The authorisation code is received from the authorisation server. One time use only and will be expired in 1 minute.
grantType	String	Required	the value MUST be set to authorization_code.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/auth/getToken
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
  "code": "xxxa42e76bf4cb0846a68e6d83d6096",
  "grantType": "authorization_code"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
accessToken	String	Required	accessToken value can only be used **once**.
tokenType	String	Required	Token type, support "Bearer" only
issueAt	Long	Required	The accessToken issue time is expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.

`expiresIn`	Long	Required	The lifetime in milliseconds of the token. The value may vary for different Online Services.
`openID`	String	Required	Tokenised ID, uniquely generated for each user of each online service website or mobile application.
`lastModifiedDate`	Long	Required	The datetime of the user complete registration at "iAM Smart" System. The value will be updated when either of the following is valid:- (1) If any one of the following verified data are changed. English name Chinese name (* not applicable if it was marked as unverified during registration) Gender Date of birth (2) User re-register "iAM Smart" after "iAM Smart" de-registration. The modification time will be expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
`userType`	String	Required	`default` or `sign` `default`: "iAM Smart" user `sign`: "iAM Smart+" user (digital signing capability)
`scope`	String	Required	The scope of the token. Please refer to the corresponding section specified in each API function.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
```

```
    "accessToken": "0ad186353c424c64897fcc00445c9ba1",
    "tokenType": "Bearer",
    "issueAt": 1557053922938,
    "expiresIn": 14400000,
    "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
    "lastModifiedDate": 1560849218006,
    "userType": "sign",
    "scope": "eidapi_sua"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D40004",
  "message": "authCode not exist or expired",
}
```

12.5.7 Obtain Anonymous Step-up Authentication Result

- **API Description**

Name	Description
Service Full Name	Obtain Anonymous Step-up Authentication Result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/anonymous/sua/getResult`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to retrieve anonymous step-up authentication result.

- **Request Parameters**

Parameter	Type	Presence	Description
`accessToken`	String	Required	`accessToken` value
`openID`	String	Required	Tokenised ID value

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/
header format.
POST
https://<iAM_Smart_domain>/api/v1/anonymous/sua/getResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencoded Base64 string of the rateLimitFactor
rateLimitFactor: {"key":"suaMethod","value":["FR","NFC"]}
// Unencrypted Request Body
{
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
`isPassed`	Boolean	Required	Step-up Authentication result. `true` - same person, otherwise `false`.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
// Decrypted Body
{
   "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
   "code": "D00000",
   "message": "SUCCESS",
   "content": {
    "isPassed": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
```

```
    "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
    "code": "D20002",
    "message": "empty parameter {accessToken}"
}
```

12.5.8 Open "iAM Smart" Mobile App for Anonymous Step-up Authentication (appv2)

- **Using URL Scheme**

Name	Description
Service Full Name	Open "iAM Smart" Mobile App for Anonymous Step-up Authentication
URI (as in RESTFUL API)	<"iAM Smart" app URL scheme>://v2_anon_sua
Service Version	V2.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to specific Step-up Authentication page in "iAM Smart" Mobile App. The URL schemes are supported on iOS and Android versions of "iAM Smart" Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
ticketID	String	Required	TicketID is a unique identifier provided by "iAM Smart" System. Online service retrieves ticketID while requesting the Anonymous Step-up Authentication function. It is an ASCII string with a length of less than or equal to 36 chars.
source	String	Required	App_Link (iOS) or App_Package (Android)
clientRedirectURI	String	Required (iOS only)	Callback redirect URI. The value should be URL encoded and registered in the self-service portal. Encoded and unencoded commas (",") are not accepted.
packageName	String	Required (Android only)	Package name of the Online Service App for callback use. Submission of Package name in the application form is required.

`activityClass`	String	Required (Android only)	Activity class name of the Online Service App for callback use.
`activityParams`	String	Optional (Android only)	Optional params used during callback.
`state`	String	Required	If the `state` parameter is presented in the request message, the same state value will be returned to Online Service during the callback. It is used to prevent the CSRF attack. The value of `state` is defined by Online Service, and it should be a secure random string of any length less than or equal to 36 (UUID string length). Only ASCII letters, numbers, underscore and hyphens are accepted.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://v2_anon_sua
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
&source=App_Package
&packageName=com.onlineservice.myapp
&activityClass=callback_activity
&activityParams=callback_param
```

13.CDEG INTEGRATION

13.1 Overview

Online services can invoke the “CDEG Integration” API to to query a user's consent status with a specified CDEG data provider or to update the corresponding consent record as long as online service holds a valid accessToken of the user.

13.2 Prerequisite

- Online service must undergo the authentication process (i.e. Request accessToken & Tokenised ID) in Section 3 in order to obtain the accessToken and openID as the input of CDEG Integration API.

13.3 Scope

Online service shall submit the application for using the following scope values to “iAM Smart” Support team in advance and provide the required scope values based on the business requirement specified in the API request. In addition, online service can check the approved scopes in the Self-Service Portal (ESP).

Scope	Description
`eidapi_auth`	Scope for CDEG Integration

13.4 Use Cases and Scenarios

13.4.1 Obtain User's Consent Result

The sequence diagram below shows how online service calls "iAM Smart" API to check if a user has given consent to a CDEG data provider.

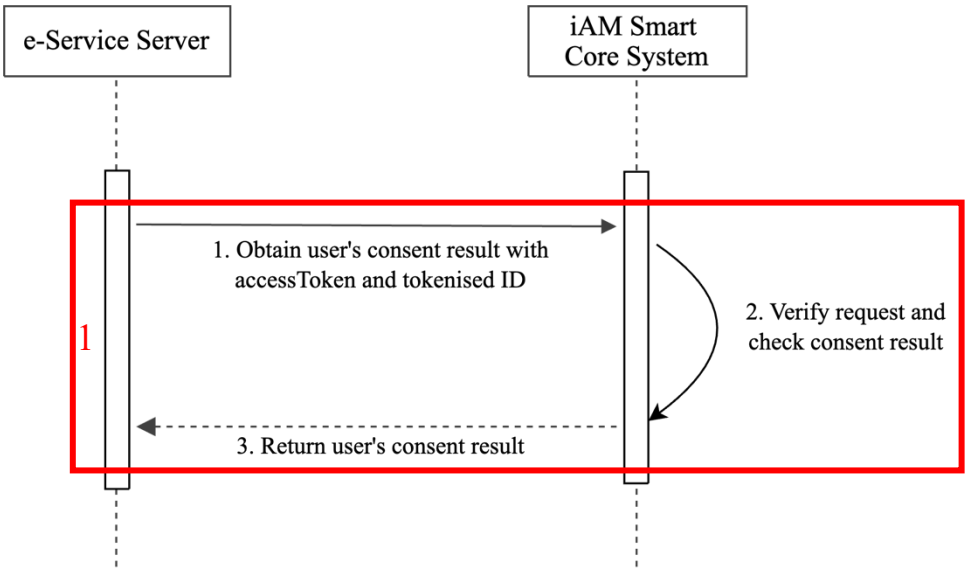

Figure-48 Obtain User's Consent Result

APIs interactions between Online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Obtain User's Consent Result	13.5.1

13.4.2 Request User's Consent(Online Service Website in Different Device)

The sequence diagram below shows how online service calls "iAM Smart" API to request a user to give consent to a CDEG data provider *when Online*

Service website and the "iAM Smart" Mobile App are running in different devices.

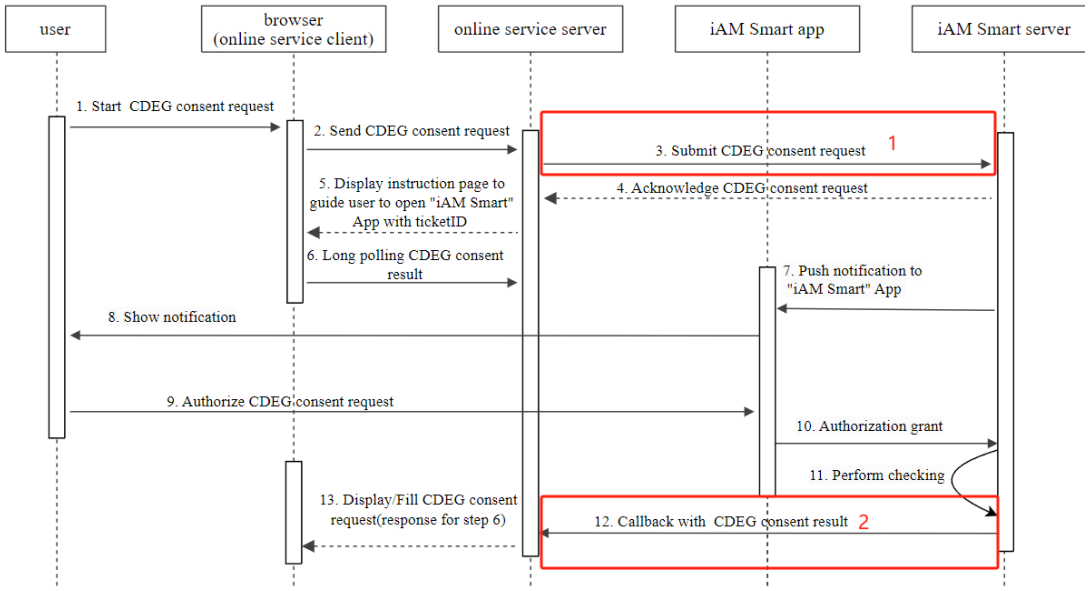

Figure-49 Request User's Consent in different devices

APIs interactions between Online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Submit CDEG consent request	13.5.2
2	Callback with CDEG consent result	13.5.4

13.4.3 Request User's Consent(Online Service Website in Same Device)

The sequence diagram below shows how online service calls "iAM Smart" API to request a user to give consent to a CDEG data provider *when Online*

Service website and the "iAM Smart" Mobile App are running in the same devices

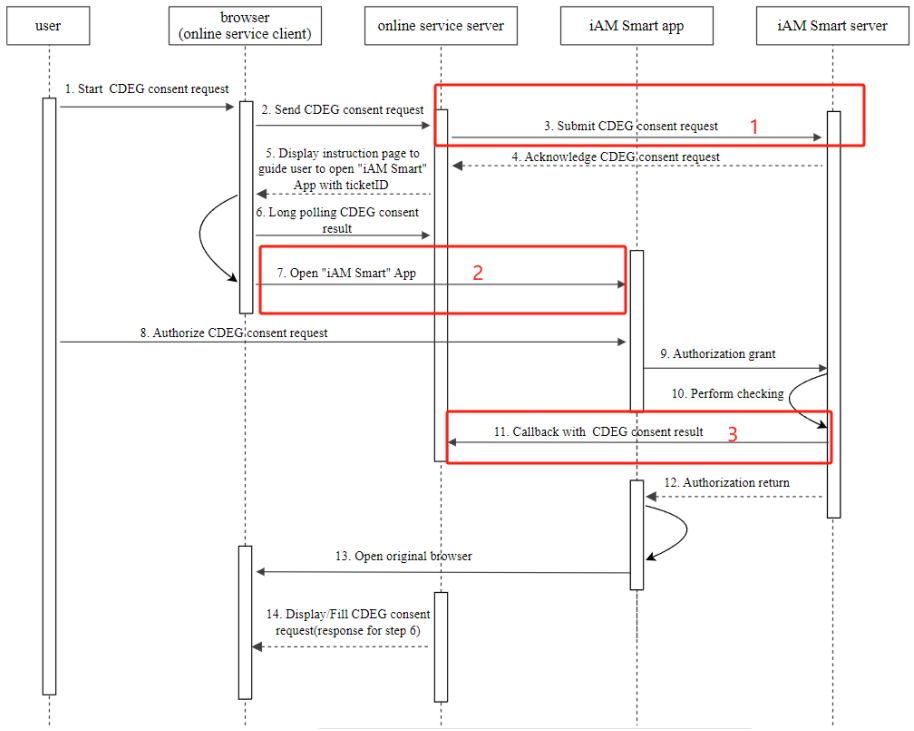

Figure-50 Request User's Consent in same devices

APIs interactions between Online service servers and "iAM Smart" System are marked in red. Technical details of respective APIs can be found in the following sections.

No.	API Name	API Reference (Section)
1	Obtain *User's Consent Result* CDEG consent request	13.5.2
2	Open "iAM Smart" App	13.5.3
3	Callback with CDEG result	13.5.4

13.5 API Implementation Details

13.5.1 Obtain User's Consent Result

- **API Description**

Name	Description
Service Full Name	Obtain user's consent result
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/cdeg/getCo`

	nsentResult
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to check if a user has given consent to a CDEG data provider. API access control, including scope (eidapi_auth) check and API access right check (CDEG Consent), will be applied when calling this API.

- **Request Parameters**

Parameter	Type	Presence	Description
providerCode	String	Required	The identity of the CDEG data provider
accessToken	String	Required	accessToken value
openID	String	Required	Tokenised ID value Maximum Length: 64

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4.1 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/cdeg/getConsentResult
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
   "providerCode": "WSD",
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
consentResult	Boolean	Required	If the user has granted consent to "All Government B/Ds" or to the specified

			provider, then "true" will be returned to the online service. Otherwise, "false" will be returned.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "consentResult": true
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4.2
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {providerCode}"
}
```

13.5.2 Submit CDEG Consent Request

- **API Description**

Name	Description
Service Full Name	Request user's consent
URI (as in RESTFUL API)	`https://<iAM_Smart_domain>/api/v1/cdeg/requestConsent`
Request Type	POST
Service Version	V1.0.0
Description of Service	Online service can use this API to request a user to give consent to a CDEG data provider. API access control, including scope (`eidapi_auth`) check and API access right check (`CDEG Consent`), will be applied when calling this API. Invoke this API only if the `consentResult` from the "Obtain User's Content Result" API returns `false`.

- **Request Parameters**

Parameter	Type	Presence	Description
providerCode	String	Required	The identity of the CDEG data provider
accessToken	String	Required	accessToken value
openID	String	Required	Tokenised ID value Maximum Length: 64
redirectURI	String	Required	Callback redirect URI. The value should be URL encoded and registered in the self-service portal and application form.
businessID	String	Required	businessID is a unique identifier for Online Service to differentiate different request. It should be ASCII string with length less than or equal to 36 chars.
source	String	Required	Request initiator. The supported source values can be found in Appendix B of this specification.

- **Example Request**

```
// Line breaks are for legibility only.
// Please refer to Section 2.4 for generating shared common parameters/ header format.
POST
https://<iAM_Smart_domain>/api/v1/cdeg/requestConsent
// Request Headers
clientID: "edae2e2529ff46228af1e4d18c8405d1"
signatureMethod: "HmacSHA256"
signature: "5X42Y1B7MEd8Mm%2BonwjiQz9VCZkkrntADskXsYntavU%3D"
timestamp: 1557048906183
nonce: "e893647dc4204eb9b7b8eddd527b687c"
// Unencrypted Request Body
{
   "providerCode": "WSD",
   "accessToken": "xxxa42e76bf4cb0846a68e6d83d6096",
   "openID": "liR14%2BvX%2F5hSum5uf4ERczu0KcDnIJA5BM7FoM1ag9c%3D",
   "redirectURI":
"https%3A%2F%2Frp_domain%2Frp_context%2Fcall_back_endpoint",
   "businessID": "b2c99aa83b0049e9ba370c5341681225",
```

```
    "source": "Android_Chrome"
}
```

- **Response Parameters**

Parameter	Type	Presence	Description
authByQR	Boolean	Required	If the user completes authentication by scanning the QR code (i.e. different device), then "true" will be returned to online service.
ticketID	String	Required (Conditional)	ticketID is a unique identifier provided by “iAM Smart” System, online service retrieve ticketID while requesting user’s consent function. It is ASCII string with length less than or equal 36 chars. The value valid for 18 minutes.

- **Example Success Response**

```
// The descriptions of txID, code, and message are in Section 2.4
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
    "authByQR": false,
    "ticketID": "bbb8aae57c104cda40c93843ad5e6db8"
  }
}
```

- **Example Error Response**

```
// The descriptions of txID, code, and message are in Section 2.4
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D20002",
  "message": "empty parameter {providerCode}"
}
```

13.5.3 Open “iAM Smart” Mobile App for Requesting User’s Consent

- **Using URL Scheme**

Name	Description
Service Full Name	Open “iAM Smart” Mobile App for requesting User’s Consent
URI (as in RESTFUL API)	`<"iAM Smart" app URL scheme>://request-consent`
Service Version	V1.0.0
Description of Service	The URL scheme makes use of deep linking to redirect users to requesting user’s consent action in “iAM Smart” Mobile App. The URL schemes supported on iOS and Android versions of “iAM Smart” Mobile App.

- **Request Parameters**

Parameter	Type	Presence	Description
`ticketID`	String	Required	`ticketID` is a unique identifier provided by “iAM Smart” System, online service retrieve `ticketID` while requesting user’s consent function. It is ASCII string with length less than or equal 36 chars.

- **Example Scheme**

```
// Line breaks are for legibility only.
<"iAM Smart" app URL scheme>://request-consent
?ticketID=bbb8aae57c104cda40c93843ad5e6db8
```

13.5.4 Callback With CDEG Consent Result

- **API Description**

Name	Description
Service Full Name	Callback to Receive Requesting User’s Consent Result
URI (as in RESTFUL API)	`https://<rp_domain>/<rp_context>/<call_back_endpoint>`
Request Type	POST
Service Version	V1.0.0
Description of Service	This callback returns the requesting User’s consent result to online service upon user consent. Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **API Specific Timeout**

Title	Timeout value	Description
Callback	18 minutes	Online service could treat the request as failed if it doesn't get callback within 18 minutes.

- **Callback Parameters**

Parameter	Type	Presence	Description
`businessID`	String	Required	`businessID` is a unique identifier for online service to differentiate different requests. Online services can use the `businessID` to relate the callback message with the original request. Maximum Length: 36
`state`	String	Optional	The same state value will be returned if the state parameter is in the request message. It is used to prevent the CSRF attack. The value of the state is defined by Online Service, and it should be a secure random value. Maximum Length: 36
`consentResult`	Boolean	Required	If the user has granted consent to “All Government B/Ds” or to the specified provider, then "true" will be returned to the online service. Otherwise, "false" will be returned.

- **Example Callback**

```
// Line breaks are for legibility only.
POST
https://<rp_domain>/<rp_context>/<call_back_endpoint>
// Callback Body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D00000",
  "message": "SUCCESS",
  "content": {
   "businessID": "3e47be25-66a6-43fb-89f6-7e2dd138aff8",
   "state": "unesidkd",
   "consentResult": true
```

```
  }

}
```

APPENDICES

A. "iAM Smart" Profile Field and "e-ME" Field Schema

Field	Adopt Common Schema	Sub-Field	Type (Chi/Eng)	Max Length (Chi/Eng)
idNo	Yes	Identification	String	8
		CheckDigit	String	1
prefix	Yes		String	35
enName	Yes	UnstructuredName	String	40
chName	Yes	ChineseName	String	6
chNameVerified	N/A		String	6
birthDate	N/A		String	8
gender	Yes		String	1
maritalStatus	Yes		String	1
homeTelNumber	Yes	CountryCode	String	3
		SubscriberNumber	String	15
officeTelNumber	Yes	CountryCode	String	3
		SubscriberNumber	String	15
mobileNumber	Yes	CountryCode	String	3
		SubscriberNumber	String	15
emailAddress	N/A		String	128
residentialAddress (Standard/Village Address)	Yes	Region	String	3
		DcDistrict	String	3
		Sub-district	String	40
		BuildingName	String	85/195
		EstateName	String	25/80
		PhaseNo	Decimal	2
		PhaseName	String	25/80
		StreetName	String	24/67
		BuildingNoFrom	String	7
		BlockDescriptor	String	14/35
		BlockNo	String	5/15
		FloorNum	String/Decimal	3
		FloorDescription	String	10/20

		UnitDescriptor	String	15/16
		UnitNo	String	23
residentialAddress (Lot Address)	Yes	DdType	String	4/5
		DdNo	String	4
		LotType	String	8/15
		LotNo	String	13
		LotSection1	String	4
		LotSubsection1	String	4
		LotSection2	String	4
		LotSubsection2	String	3
		LotSection3	String	4
		LotSubsection3	String	3
		LotExtendPortion Code	String	1
residentialAddress (Free Format)	No	LanguageCode	String	2
		AddressLine1	String	250
		AddressLine2	String	250
		AddressLine3	String	250
postalAddress (PostBoxAddress)	Yes	PoBoxNo	Decimal	6
		PostOffice	String	15/50
		PostOfficeRegion	String	5/30
educationLevel	Yes		String	1
addressDocInfo	N/A	enProviderName / tcProviderName / scProviderName	String	255
		enUserName / tcUserName	String	40
		addressLine1 (enServiceAddress / tcServiceAddress / enPostalAddress / tcPostalAddress)	String	80
		addressLine2 (enServiceAddress / tcServiceAddress / enPostalAddress / tcPostalAddress)	String	80

		addressLine3 (enServiceAddress / tcServiceAddress / enPostalAddress / tcPostalAddress)	String	80
		addressLine4 (enServiceAddress / tcServiceAddress / enPostalAddress / tcPostalAddress)	String	80
		lastRetrievalDate	Long	13
addressDocFile (callbackContentType “application/json”)	N/A	docFile	String	N/A
		docHash	String	64
		billDate	Long	13
addressDocFile (callbackContentType is “multipart/form-data”)	N/A	docHash	String	64
		billDate	Long	13
docFile (callbackContentType is “multipart/form-data”)	N/A	N/A	Binary	N/A

B. Supported Value at Source Parameter

"iAM Smart" System supports different types of browsers and Online Service App calling methods. Online service should detect its user agent value for a browser use case and pass the respective source value to the "iAM Smart" System.

Platform	Supported Browser / Online Service App calling method	Source
For Online Service **Web/App** in Different Device and For Online Service **Web** in Same Device		
Android Browser	Chrome	Android_Chrome
	Firefox	Android_Firefox
	Edge	Android_Edge
	Samsung built-in browser	Android_Samsung
	Huawei built-in browser	Android_Huawei
	Xiaomi built-in browser	Android_Xiaomi
	In-app browser[6]	Android_ IMS_InAppBrowser
iOS Browser	Safari	iOS_Safari
	Chrome	iOS_Chrome
	Firefox	iOS_Firefox
	Edge	iOS_Edge
	In-app browser[6]	iOS_IMS_InAppBrowser
Desktop Browser	Chrome, IE, Edge, Firefox, Safari	PC_Browser

For Online Service **App** in Same Device With "iAM Smart" App API v2		
Online Service Mobile App	To be invoked by Universal Link (iOS)	App_Link
	To be invoked by App_Package (Android)	App_Package

For Online Service **App** in Same Device With "iAM Smart" App API v1		
Online Service Mobile App	To be invoked by URL Scheme	App_Scheme
	To be invoked by Universal Link (iOS) / App Link (Android)	App_Link

[6] Online Service that is running in the in-app browser and calling iAM Smart APIs that require the source parameter should set this source value accordingly. More details can be found in the "iAM Smart In-App Browser Integration Technical Reference" document.