

“iAM Smart” Developer Guide for Online Services

Version: 2.2.2

JANUARY 2025

© The Government of the Hong Kong Special Administrative Region
of the People's Republic of China

The contents of this document remain the property of and may not be reproduced in whole or in part
without the express permission of the Government of
the Hong Kong Special Administrative Region of the People's Republic of China

Document Revision History

Ver. No.	Release Date	Section Affected	Summary of Changes
1.2.1	05 August 2020	Initial version	
1.2.2	10 November 2020	*, 3.5	Update terms New section for Direct Access.
1.2.3	22 Oct 2021	Section 1.1.3	Added description of the new billing address information in the e-ME profile
1.2.4	13 Apr 2022	Section 1.3	Add the description of Universal Link (iOS) /App Link (Android) for e-Service App
1.2.5	15 Jul 2022	Section 3.2.3	Update Error Code, D20012
		Section 3.5, 3.8	Update Form Filling scope, requests, response, and callback to V2. Online Service can request both eMEFields and profileFields with these V2 APIs.
1.2.6	14 Sep 2022	Section 1.1.10	Added the description of the support to CCIC holders
		Section 3.4.6	Added the description and workflow to verify whether iAM Smart user is a CCIC holder
1.2.7	5 Dec 2022	Section 3.9	Added Anonymous Signing
1.2.8	25 July 2024	Terms And Conditions	Updated organisation name
2.2.1	1 Nov 2024	Section 1.1 1.1.1, 1.1.2, 1.1.3	Updated the description of Overview, Introudction of API Functions and “iAM Smart” Account Version and User Profiles and Authorisation Scope and Access Token
		Section 1.1.10, 1.1.11	Added the description of Service Catalogue and Direct Login
		Section 3.10.1	Added the description of Direct Login v2 for “iAM Smart” user to login to Online Service from Service Catalogue in a simple and swift manner
		Section 1.1.1	Added Bulk Digital Signing

Ver. No.	Release Date	Section Affected	Summary of Changes
		Section 1.1.2	Removed duplicated paragraph
		Section 3.10.1	Added supported browser list for Direct Login v2
		Section 3.11 - 3.15 Section B.3	Added Bulk Digital Signing WorkFlow Description Added Implementation Reference for Bulk Digital Signing
		Section B.3	Updated APP URL scheme from “hk.gov.ogcio” to “hk.gov.iamsmart”
		Whole document	The terms “government”, “government and related organisations” and “GRO” can be used interchangeably in the whole document
		Section 1.3, 3.10.4	Update Direct Login Workflow description, pre-requisites, requests, response, and callback to V2.
		Terms And Conditions Section 1.1, 3.10.1	Update OGCIO to DPO
		Section 3.14,3.15	Delete the redundant parameter “accessToken”
2.2.2	6 January 2025	Appendix D	Added common pitfalls of App-to-App Direct Login v2

Table of Contents

Document Revision History	i
Table of Contents	i
Terms And Conditions	1
Definitions.....	2
1. Introduction.....	3
1.1 OVERVIEW	3
1.1.1 Introduction of API functions	4
1.1.2 “iAM Smart” Account Version and User Profiles	5
1.1.3 Authorisation Scope and Access Token	7
1.1.4 Authentication for Online Service Login and Authorisation for Anonymous “iAM Smart” APIs	9
1.1.5 Callback API and Transient Input Data	10
1.1.6 Data Protection for “iAM Smart” API.....	10
1.1.7 Identification Code and Result Notification for Digital Signing	11
1.1.8 Re-authentication.....	12
1.1.9 Consular Corps Identity Card (“CCIC”).....	12
1.1.10 Service Catalogue and Direct Login	13
1.1.11 Not Applicable.....	14
1.1.12 Bulk Digital Signing	14
1.2 ASSUMPTIONS AND CONSTRAINTS.....	15
1.3 ONLINE SERVICE REGISTRATION TO “IAM SMART” SYSTEM	16
1.4 POTENTIAL BUSINESS CASES USING “IAM SMART”	16
1.4.1 Not Applicable.....	18
1.4.2 User Authentication	18
1.4.3 Not Applicable.....	18
1.4.4 Digital Signing with Service Login	18
1.4.5 Re-authentication with Service Login	20
1.4.6 Anonymous Form Filling	21
1.4.7 Anonymous Digital Signing	21
1.4.8 Bulk Digital Signing with Service Login	23
1.4.9 Anonymous Bulk Digital Signing.....	24
2. System Workflow Using “iAM Smart”	26
3. Implementation Procedure	26
3.1 PREREQUISITE	26
3.2 ONLINE SERVICE AND “IAM SMART” SYSTEM INTERACTION DESIGN	27
3.2.1 Interaction between Online Service and “iAM Smart” Mobile App	27

3.2.2	Process of Common API Request and Response Parameters	29
3.2.3	Process of Common Errors Returned from “iAM Smart” APIs	32
3.3	API DATA ENCRYPTION AND DECRYPTION.....	36
3.3.1	Workflow for Encryption and Decryption.....	37
3.3.2	Encryption & Decryption Scope and Examples	41
3.3.3	Proper Handling of Encryption and Decryption	41
3.4	WORKFLOWS FOR AUTHENTICATION.....	42
3.4.1	Scenario 1: Authentication (Online Service Website in Different Device)	42
3.4.2	Scenario 2: Authentication (Online Service Website in Same Device).....	48
3.4.3	Scenario 3: Authentication (Online Service App in Different Device)	52
3.4.4	Scenario 4: Authentication (Online Service App in Same Device)	56
3.4.5	Workflows for verifying CCIC user	59
3.5	WORKFLOWS FOR FORM FILLING AFTER AUTHENTICATION API	62
3.5.1	Scenario 1: Form Filling (e-Service Website/App in Different Device)	62
3.5.2	Scenario 2: Form Filling (e-Service Website in Same Device)	67
3.5.3	Scenario 3: Form Filling (e-Service App in Same Device)	72
3.6	WORKFLOWS FOR FORM FILLING WITHOUT SERVICE LOGIN (AKA ANONYMOUS FORM FILLING).....	76
3.6.1	Scenario 1: Anonymous Form Filling (Online Service Website in Different Device) 76	
3.6.2	Scenario 2: Anonymous Form Filling (Online Service Website in Same Device)...	84
3.6.3	Scenario 3: Anonymous Form Filling (Online Service App in Different Device) ...	91
3.6.4	Scenario 4: Anonymous Form Filling (Online Service App in Same Device)	98
3.7	WORKFLOWS FOR DIGITAL SIGNING WITH SERVICE LOGIN	105
3.7.1	Scenario 1: Digital Signing (Online Service Website/App in Different Device) ...	105
3.7.2	Scenario 2: Digital Signing (Online Service Website in Same Device)	117
3.7.3	Scenario 3: Digital Signing (Online Service App in Same Device)	123
3.8	WORKFLOWS FOR DIGITAL SIGNING WITHOUT SERVICE LOGIN (AKA ANONYMOUS DIGITAL SIGNING)	129
3.8.1	Scenario 1: Anonymous Digital Signing (Online Service Website in Different Device).....	129
3.8.2	Scenario 2: Anonymous Digital Signing (Online Service Website in Same Device) 140	
3.8.3	Scenario 3: Anonymous Digital Signing (Online Service App in Different Device) 147	
3.8.4	Scenario 4: Anonymous Digital Signing (Online Service App in Same Device)...	153
3.9	WORKFLOWS FOR RE-AUTHENTICATION WITH SERVICE LOGIN	161
3.9.1	Scenario 1: Re-authentication (Online Service Website/App in Different Device)	161

3.9.2	Scenario 2: Re-authentication (Online Service Website in Same Device)	166
3.9.3	Scenario 3: Re-authentication (Online Service App in Same Device)	171
3.10	WORKFLOWS FOR DIRECT LOGIN & DIRECT ACCESS.....	175
3.10.1	Scenario 1: Direct Login with Online Service Website (aka Direct Login v2)	175
3.10.2	Scenario 2: Direct Access with Online Service Website	180
3.10.3	Scenario 3: Direct Login with Online Service Website (Fallback Mechanism)	181
3.10.4	Scenario 4: Direct Login with Online Service App (Direct Login v2)	182
3.10.5	Scenario 5: Direct Login with Online Service App (Direct Login v1)	192
3.11	WORKFLOWS FOR BULK DIGITAL SIGNING WITH SERVICE LOGIN	198
3.11.1	Scenario 1: Bulk Digital Signing (Online Service Website/App in Different Device)	198
3.11.2	Scenario 2: Bulk Digital Signing (Online Service Website/App in Same Device)	210
3.12	WORKFLOWS FOR BULK DIGITAL SIGNING WITHOUT SERVICE LOGIN (AKA ANONYMOUS BULK DIGITAL SIGNING)	216
3.12.1	Scenario 1: Anonymous Bulk Digital Signing (Online Service Website in Different Device).....	216
3.12.2	Scenario 2: Anonymous Bulk Digital Signing (Online Service Website in Same Device).....	227
3.12.3	Scenario 3: Anonymous Bulk Digital Signing (Online Service App in Different Device).....	232
3.12.4	Scenario 4: Anonymous Bulk Digital Signing (Online Service App in Same Device)	238
3.13	WORKFLOWS FOR CALLBACK BULK DIGITAL SIGNING RESULT	245
3.13.1	Workflows for Callback Bulk Digital Signing Result	245
3.14	WORKFLOWS FOR ENQUIRE BULK DIGITAL SIGNING RESULT	248
3.14.1	Workflows for Enquire Bulk Digital Signing Result.....	248
3.15	WORKFLOWS FOR CANCEL BULK DIGITAL SIGNING REQUEST.....	250
3.15.1	Workflows for Cancel Bulk Digital Signing Request.....	250
Appendices.....		252
A.	NOT APPLICABLE	252
B.	NOT APPLICABLE	252
C.	NOT APPLICABLE	252
D.	COMMON PITFALLS OF APP-TO-APP DIRECT LOGIN V2錯誤! 尚未定義書籤。	
D.1	Unable to launch Android App via Package Name	錯誤! 尚未定義書籤。
D.2	Unable to obtain the authCode and code_verifier from Android Intent	錯誤! 尚未定義書籤。

TERMS AND CONDITIONS

The Government of the HKSAR (“HKSARG”) has the sole discretion to amend or vary the Developer Guide from time to time.

The contents of the Developer Guide remain the property of, and shall not be reproduced in whole or in part without the express permission of HKSARG. Information provided by the Developer Guide and all the associated intellectual property rights are retained by HKSARG.

Online service Providers and their contractors (and sub-contractors, if applicable) (hereafter referred as users) may use the Developer Guide for the purpose of designing, developing, testing and running of their applications when adopting iAM Smart. By using the Developer Guide, users agree not to sue HKSARG and agree to indemnify, defend and hold harmless HKSARG, its officers and employees from any and all third party claims, liability, damages and/or costs (including but not limited to, legal fees) arising from the use of the Developer Guide.

HKSARG will not be liable for any direct, indirect, incidental, special or consequential damages of any kind resulting from the use of or inability to use the information provided by the Developer Guide.

Users agree to abide the usage terms and conditions specified here before releasing the Developer Guide to their contractors (and sub-contractors, if applicable).

DEFINITIONS

The following terms and abbreviations are used in this document.

Terms / Abbreviations	Definition
“iAM Smart” System	The “iAM Smart” backend services to provide “iAM Smart” APIs and trigger Online Service callback APIs to return results to Online Service applications.
Online Service server	The Online Service backend services make use of “iAM Smart” APIs and provide Online Service callback APIs to receive results from “iAM Smart” System.
Online Service client terminal	The browser (PC or mobile) and/or Online Service mobile application that provide user interface to “iAM Smart” user and backend interface to Online Service server.
Online Service Website	It is the webpage generated by the Online Service server and displayed by a browser (Desktop PC or mobile).
“iAM Smart” e-Cert	Digital certificate issued by Recognized Certification Authority for “iAM Smart” account with digital signing function enabled. The usage and maintenance of the “iAM Smart” e-Cert are managed by “iAM Smart” System for the “iAM Smart” account.
Content Encryption Key or CEK	Symmetric encryption key provided by “iAM Smart” System for API data encryption and decryption.
Key Encryption Key or KEK	The public key certificate of Online Service for encrypting the CEK from the “iAM Smart” System.
Cross-Site Request Forgery or CSRF Attack	CSRF is an attack that forces an end user to execute unwanted actions on a web application in which he/she is currently authenticated.
“iAM Smart” BDSS	The “iAM Smart” Bulk Digital Signing System to support the provision of digital signing for the “iAM Smart” users to perform digital signing on multiple documents with only one digital signing cycle.

1. INTRODUCTION

1.1 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 government information and online services without the need of account registration.

Online Services shall follow both “iAM Smart” API Specification and “iAM Smart” Developer Guide for “iAM Smart” Adoption.

1.1.1 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 (aka Form Filling)**
Online services can make use of the Form Filling 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.
- **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 provide 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 a “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”.

1.1.2 “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+” with mobile phones supporting NFC. Without NFC supported mobile phone, “iAM Smart” account will be registered.

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+”

- Remote registration for “iAM Smart+” requires applicant to use personal mobile phone supporting NFC to read the HKIC and take selfies for identity verification purpose. The whole process can be completed online using mobile phone.
- 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>).

“iAM Smart” User Profiles

An “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. The “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 address¹, 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

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.

also available in “iAM Smart” User Profiles, however, only the “iAM Smart+” version has verified the user’s “Chinese Name”². Depending on the “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 conducted³.

The account information in “profileFields” and “eMEFields” are the same for same user (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 request. In addition, unlike “profileFields” which is for the purpose of identity verification, Online Service can allow “iAM Smart” user to modify online form data filled with “eMEFields”.

1.1.3 Authorisation Scope and Access Token

“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

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.

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.

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

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.

Get Access Token

Online Service should invoke “iAM Smart” API “Request QR Page” with the required authorisation scope(s). With successful authorisation of the “iAM Smart” user, “iAM Smart” System will return an Authorisation Code (“authCode”) with the required authorisation scopes (i.e., access rights for “iAM Smart” APIs) to Online Service via the Online Service client terminal (e.g., browser). Online Service then exchanges the authCode with “iAM Smart” System for the Access Token (“accessToken”), the Tokenised ID and other metadata of the “iAM Smart” user. The accessToken is bound to the authorisation scope(s) and Online Service must present the accessToken and relevant parameters to access the “iAM Smart” APIs. Online Service should retain the accessToken for access to “iAM Smart” APIs within its validity period and destroy the accessToken when it expired, become invalid ^{Note 1} or the “iAM Smart” user left the Online Service (e.g., logout).

The process to get accessToken by Online Service is summarised below:

Step	Description
1	<p>Online Service requests for authCode with required authorisation scope(s) by using Online Service client terminal (e.g., browser or Online Service App) to invoke the “iAM Smart” API “Request QR Page”. Depending on the API request parameters, either:</p> <ul style="list-style-type: none"> - A webpage with QR code will be shown in Online Service client terminal [A]; or - “iAM Smart” Mobile App installed in the device of the Online Service client terminal will be launched [B].

Step	Description
2	<p>“iAM Smart” user logs in his/her “iAM Smart” Mobile App. To synchronise “iAM Smart” System with his/her “iAM Smart” Mobile App:</p> <ul style="list-style-type: none"> - For [A], “iAM Smart” user will use the “iAM Smart” Mobile App to scan the QR code. - For [B], the synchronisation will be established upon login of “iAM Smart” Mobile App.
3	“iAM Smart” user gives authorisation in “iAM Smart” Mobile App for Online Service’s request of authCode.
4	“iAM Smart” System returns authCode to Online Services using Online Service callback API via the same Online Service client terminal in Step 1.
5	Online Service invokes the “iAM Smart” API “Request accessToken & Tokenised ID” to get accessToken using the authCode.
6	“iAM Smart” System returns the accessToken, Tokenised ID and other metadata of the “iAM Smart” user to Online Service.
7	Online Service uses the accessToken and relevant parameters to access “iAM Smart” APIs.

Note 1: The accessToken obtained for invoking “iAM Smart” APIs for anonymous form filling and anonymous digital signing on document hash can be used only once. The accessToken obtained for invoking other “iAM Smart” APIs is valid for multiple use within its lifetime as indicated in the response of “iAM Smart” API “Request accessToken & Tokenised ID”.

Details of authorisation scope can be found in the “iAM Smart” API Specification.

1.1.4 Authentication for Online Service Login and Authorisation for Anonymous “iAM Smart” APIs

Authentication for Online Service Login

To complete the “Get Access Token” process as stipulated in Section 1.1.3, “iAM Smart” user has to authenticate his/her identity using 2FA (i.e., the mobile device bounded with his/her “iAM Smart” account and a local biometric authentication with the “iAM Smart” Mobile App) to log in “iAM Smart” System to give authorisation. As such, successful completion of this “Get Access Token” process resulting in return of accessToken and Tokenised ID to Online Service should be accepted as successful authentication of the “iAM Smart” user to the Online Service.

Details of authentication can be found in Section 3.4.

Authorisation for Anonymous “iAM Smart” APIs

For Online Service to access “iAM Smart” APIs for anonymous form filling and anonymous digital signing using “iAM Smart”, unlike the above Online Service Login process, the successful completion of “Get Access Token” process means that “iAM Smart” user has authorised a one-time access to the relevant “iAM Smart” API. The relevant accessToken should be valid for only one usage. Online Service must go through the same “Get Access Token” process for every access to the same or different anonymous “iAM Smart” APIs.

Details of authorisation for anonymous “iAM Smart” API can be found in Section 3.6 and 3.8.

1.1.5 Callback API and Transient Input Data

Due to the possible unexpected delay (e.g., network latency) and/or action time of the “iAM Smart” user to perform authorisation in the “iAM Smart” Mobile App (e.g., to determine which data items in “e-ME profile” are allowed for retrieval by Online Service), Online Service may experience timeout if an instant API response design is used. Therefore, asynchronised API response is adopted in “iAM Smart” APIs. Online Service is required to implement its own callback APIs to support the return of asynchronised API response from the “iAM Smart” System after invoking the relevant “iAM Smart” APIs. Online Service should also manage “iAM Smart” user's transient input data before invoking any “iAM Smart” APIs.

1.1.6 Data Protection for “iAM Smart” API

All “iAM Smart” and Online Service callback APIs are protected by HTTPS protocol during transmission. Additional API data encryption on API POST request and response body is implemented with encryption key being generated and renewed in “iAM Smart” System using a dedicated “iAM Smart” API “Request/Revoke Symmetric Content Encryption Key”. The data encryption key has its validity and registered Online Service must request/renew the key by itself and make use the key to protect the data when invoking the relevant “iAM Smart” APIs.

1.1.7 Identification Code and Result Notification for Digital Signing

For “iAM Smart” user to digitally sign a document in Online Service, Online Service should first generate the hash from the document (or document digest for a PDF document) to be signed (“Document Hash”). The Document Hash will be sent to “iAM Smart” System for digital signing using the private key of “iAM Smart” user's “iAM Smart” e-Cert. A signature bundle including the digital signature, “iAM Smart” e-Cert, etc., will be returned to Online Service for further computing of the signed document. When “iAM Smart” user has logged in the Online Service, both Online Service and “iAM Smart” System will compute and display a 4-digit identification code using the Document Hash and the hash of Tokenised ID of the “iAM Smart” user. For anonymous digital signing, the 4-digit identification code will be computed using the Document Hash, the hash of HKIC number and the hash of client ID of Online Service. “iAM Smart” User should be requested to verify the two identification codes are the same before authorising the digital signing request. After receiving and verifying the signature bundle using the “iAM Smart” e-Cert, Online Service should notify the digital signing result to “iAM Smart” System by calling the “Online Service Acknowledges Digital Signing Result” “iAM Smart” API.

Details of the computation algorithm of the 4-digit identification code can be found in Section 3.7 and 3.8.

For bulk digital signing, Online Service should first generate the hash from the documents (and/or document digest for PDF documents) to be signed (“Document Hash”). The Document Hash will be sent to “iAM Smart” System for Bulk Digital Signing using the private key of “iAM Smart” user's “iAM Smart” e-Cert. A signature bundle including the digital signature, “iAM Smart” e-Cert, etc., will be returned to Online Service for further computing of the signed documents. When “iAM Smart” user has logged in the Online Service, both Online Service and “iAM Smart” System will compute and display a 6-digit identification code using the Document Hash and the hash of Tokenised ID of the “iAM Smart” user. For anonymous digital signing, the 6-digit identification code will be computed using the Document Hash, the hash of HKIC number and the hash of client ID of Online Service. “iAM Smart” User should be requested to verify the two identification codes are the same before authorising the digital signing request. After receiving and verifying the signature bundle using the “iAM Smart” e-Cert, Online Service should notify the digital signing result to “iAM Smart” System by calling the “Online Service Acknowledges Bulk Digital Signing Result” “iAM Smart” API.

Details of the computation algorithm of the 6-digit identification code can be found in Section 3.11.1.1.

1.1.8 Re-authentication

Re-authentication provides an alternative for Online Service to re-confirm the “iAM Smart” user’s identity before completing a critical transaction (e.g., confirm submission of his tax return). After an “iAM Smart” user has authenticated in Online Service using “iAM Smart”, Online Service can leverage the “Request Re-authentication” “iAM Smart” API to request the same “iAM Smart” user to re-authenticate himself/herself to “iAM Smart” System via “iAM Smart” Mobile App.

1.1.9 Consular Corps Identity Card (“CCIC”)

“iAM Smart” System supports the holders of Consular Corps Identity Card (“CCIC”) to register for “iAM Smart+” accounts with effect from November 2022. A dedicated registration team is responsible for the registration of “iAM Smart+” accounts for CCIC holders.

CCIC is one kind of Hong Kong Identity Card issued by Immigration Department (“ImmD”) since 20 February 2019 and the registration data of the CCIC holders are recognised as part of the Registration of Persons (“ROP”) data. CCICs are issued to consuls, consular staff, the head and members of the Office of the European Union in Hong Kong, their spouses and dependent children of age 11 or above.

More information about CCIC can be found on ImmD website: (<https://www.immd.gov.hk/eng/services/hkid/ccic.html>).

There are two ways for Online Services to verify whether the specific “iAM Smart” user is a CCIC holder:

1. Since “J” prefix of identity card number is only allotted for the holders of CCIC, Online Services may invoke the “GetProfile” or form filling API and check against the prefix of the identity card number of the “iAM Smart” user so as to distinguish whether it is a CCIC holder.
2. Online Services can also invoke the “verifyIsCcic” API to verify whether the “iAM Smart” user is a CCIC holder after the user performs the authentication process and obtain the accessToken and Tokenised ID. Details of verification of CCIC holder can be found in

Section 0.

1.1.10 Service Catalogue and Direct Login

“iAM Smart” Platform provides a Single Portal for Online Services to enable users to access Online Service directly with one single login through “iAM Smart”. Online Service Catalogue within “iAM Smart” Mobile App offers a list of Online Services which have adopted the features of “iAM Smart”. With Direct Login function, “iAM Smart” users can choose the desired Online Service to launch its web application for logging into the Online Service directly (i.e., without the need of providing further login credentials from user). Authentication is performed at the background with minimal user intervention for better user experience. Currently, Direct Login function is only available for “iAM Smart” enabled web service, but not mobile app service.

Direct Login

To provide a better user experience to “iAM Smart” user, “iAM Smart” provides a simplified Direct Login process, allowing user to initiate the login request from “iAM Smart” service catalogue (i.e., Direct Login v2). The implementation details can be found in Section 3.10.1 of this developer guide.

Direct Access

Similar to Direct Login workflow, “iAM Smart” users can choose the desired Website or App from Service Catalogue to launch its web application with external browser if their business process does not involve “Authentication”. Special approval is required to use Direct Access, and Online Service using the Direct Access function needs to provide sound justification to the Support Team.

1.1.11 Not Applicable

1.1.12 Bulk Digital Signing

The iAM Smart Bulk Digital Signing System (BDSS) is to support the provision of digital signing for the “iAM Smart” users to perform digital signing on multiple documents with only one digital signing cycle.

BDSS is composed of the following segmentations:

1. Bulk Digital Signing Module – provide functions including batch submission & handling and bulk digital signing for digital signing process which is initiated by the Online Service to submit the request for bulk digital signing with “iAM Smart” authorisation by the user. This bulk digital signing supports mix of hash and/or PDF digital signings, normal and anonymous digital signing flow.
2. Integration with Online Service – provide the RESTful APIs for integration with the Online Service.
3. Integration with “iAM Smart” Mobile App – integrate with the existing “iAM Smart” Mobile App for user authorisation of the bulk digital signing request, push notification and related notification function.
4. Integration with “iAM Smart” support functions – implement the system support functions including but not limited to statistical reports, billing reports, audit trails and audit log

1.2 ASSUMPTIONS AND CONSTRAINTS

1. “iAM Smart” System adopts HTTPS to protect the data in transmission. Additional API data encryption is adopted for API POST request and response body.
2. Application data in the API POST request and response is in JSON format. Online Service Applications are responsible for processing of the returned data and reflecting the result to end users if required.
3. List of browsers supported by “iAM Smart” System is shown in Appendix B of the “iAM Smart” API Specification. Online Service should detect the browser’s user agent value and pass respective source value in the list to “iAM Smart” System when necessary.
4. For Online Service not using Spring Boot, Spring Cloud development platform nor Microservices style for development, Online Service should follow HTTPS standard protocols, “iAM Smart” API Specification and “iAM Smart” Developer Guide to develop the interfaces with “iAM Smart” System.
5. Please also refer to “iAM Smart” API Specifications and other technical documents for other assumptions and possible constraints.

1.3 ONLINE SERVICE REGISTRATION TO “IAM SMART” SYSTEM

Online Service Providers are required to register their Online Services in “iAM Smart” System for accessing “iAM Smart” APIs. Registration information including application name, digital certificate for encipherment, redirect URL, etc., are required to be submitted by Online Services for the registration. A client ID (“clientID”) and a secret key (“clientSecret”) will be generated by “iAM Smart” System to the Online Service for accessing “iAM Smart” APIs upon the completion of the registration.

Online Service Provider is also required to provide Universal Link (iOS) / App Link (Android) and Package Name (Android) that offer better user experience and greater security for “iAM Smart” Mobile App to launch Online Service App from Service Catalogue. Custom URL scheme is not accepted unless sound justification can be provided.

1.4 POTENTIAL BUSINESS CASES USING “IAM SMART”

Potential business cases using “iAM Smart” are listed in the following sections for reference. For each case, possible high-level steps and the corresponding “iAM Smart” APIs and callback APIs to be used and/or implemented are provided (depending on the interface of Online Service and “iAM Smart” Mobile App as stipulated in Section 3.2.1).

While the involved “iAM Smart” APIs and callback APIs are listed below, details of the “iAM Smart” APIs and callback APIs can be found in the “iAM Smart” API Specification.

“iAM Smart” APIs:

#	“iAM Smart” API	Type
1	Request Symmetric Content Encryption Key	Data encryption / decryption
2	Revoke Symmetric Content Encryption Key	Data encryption / decryption
3	Request QR Page	Authentication
4	Request accessToken & Tokenised ID	Authentication
5	Open “iAM Smart” Mobile App for Authentication	Authentication
6	Open “iAM Smart” Mobile App for Getting Context	Interface with “iAM Smart” Mobile App for business operation
7	Request Form Filling (Profile)	User Registration/Update
8	Request Form Filling	Business operation
9	Request Digital Signing	Business operation
10	Request Re-authentication	Business operation

#	“iAM Smart” API	Type
11	Online Service Acknowledges Digital Signing Result	Business operation
12	Request PDF Digital Signing	Business operation
13	Request Anonymous Form Filling	Business operation
14	Obtain Anonymous Form Filling Result	Business operation
15	Request Anonymous Digital Signing	Business operation
16	Obtain Anonymous Digital Signing Result	Business operation
17	Request Anonymous PDF Digital Signing	Business operation
18	Obtain Anonymous PDF Digital Signing Result	Business operation
19	Verify CCIC User	Business operation
20	Obtain User Profiles information	Business operation
21	Request Bulk Digital Signing	Business operation
22	Request Anonymous Bulk Digital Signing	Business operation
23	Request BSQC Token	Business operation
24	Online Service Acknowledges Bulk Digital Signing Result	Business operation
25	Enquire Bulk Digital Signing Status	Business operation
26	Cancel Bulk Digital Signing Request	Business operation

Callback APIs implemented by Online Services:

#	Callback API (Implemented by Online Service)	Type
A	Callback with authCode to Online Service App	Authentication
B	Callback with authCode to Online Service Server	Authentication
C	Callback to Receive “iAM Smart” Profile	Business operation
D	Callback to Receive Form Filling Information	Business operation
E	Callback to Receive Digital Signing Result	Business operation
F	Callback to Receive Re-authentication Result	Business operation
G	Callback to Receive PDF Digital Signing Result	Business operation
H	Callback with AuthCode to Online Service Server (Direct Login V2)	Authentication
I	Callback to Receive BSQC Token	Business operation
J	Callback to Receive Bulk Digital Signing Result	Business operation

1.4.1 Not Applicable

1.4.2 User Authentication

User authentication is similar to user registration. For “iAM Smart” user who has linked his/her “iAM Smart” account (i.e., Tokenised ID) to Online Service user repository, Online Service can authenticate the “iAM Smart” user with the following steps:

Step	Description	“iAM Smart” API	Callback API
1a	Online Service provides a link for login by “iAM Smart” in the application.		
1b	Alternatively, “iAM Smart” user can login the Online Service via “iAM Smart” service catalogue		
2a	Online Service requests for authCode with required authorisation scopes (e.g., “iAM Smart” authentication, form filling authorisation, etc.).	3, 5	A, B
2b	Alternatively, if “iAM Smart” user login the Online Service via “iAM Smart” service catalogue, Online Service would receive the authCode accordingly.		H
3	Online Service gets accessToken and Tokenised ID of “iAM Smart” user and checks if it has linked with any user account in the Online Service.	4	
4	User authentication completes if linkage found in Step 3 and Online Service permits “iAM Smart” user's login.		
5	Online Service keeps the accessToken for subsequent access to “iAM Smart” APIs.		

1.4.3 Not Applicable

1.4.4 Digital Signing with Service Login

The accessToken of the “iAM Smart” user that possessed by Online Service should include the authorisation scope of digital signing authorisation. Online Service can request for “iAM Smart” digital signing with the following steps:

Step	Description	“iAM Smart” API	Callback API
1	Online Service provides digital signing using “iAM Smart” in Online Service application.		
2	Online Service requests for authCode with required authorisation scopes (e.g., “iAM Smart” authentication, Digital Signing authentication).	3, 5	A, B
3	Online Service gets accessToken and Tokenised ID of “iAM Smart” user.	4	
4	Online Service calculates hash of the document (or digest for pdf document) to be signed and requests digital signing from “iAM Smart” System.	9,12	
5	“iAM Smart” System notes receipt of Online Service's request.		
6	Online Service generates a 4-digit identification code using hash of document (or digest for pdf document) and hash of Tokenised ID.		
7a	When Online Service client terminal and “iAM Smart” Mobile App are in different device, <i>Online Service client terminal shows the 4-digit identification code with instructions to inform “iAM Smart” user to verify the identification code and complete the “iAM Smart” digital signing in “iAM Smart” Mobile App and polls Online Service server for response from “iAM Smart” System.</i>		
7b	When Online Service client terminal and “iAM Smart” Mobile App are in same device, <i>Online Service client terminal shows a 4-digit identification code and invokes “iAM Smart” Mobile App and polls Online Service server for response from “iAM Smart” System.</i>	6	
8	Online Service server receives signing response from “iAM Smart” System via Online Service callback API and computes the signed document.		E, G
9	Online Service server verifies and acknowledges digital signing result to “iAM Smart” System.	11	
10	Online Service server informs Online Service client terminal to stop polling.		
11	Online Service saves the accessToken for subsequent access to “iAM Smart” APIs.		

1.4.5 Re-authentication with Service Login

The accessToken of the “iAM Smart” user that possessed by Online Service should include the authorisation scope of Re-authentication authorisation. Online Service application can request for re-authentication with the following steps:

Step	Description	“iAM Smart” API	Callback API
1	Online Service provides re-authentication using “iAM Smart” in Online Service application.		
2	Online Service requests for authCode with required authorisation scopes (e.g., “iAM Smart” authentication, Re-authentication authorisation).	3, 5	A, B
3	Online Service gets accessToken and Tokenised ID of “iAM Smart” user.	4	
4	Online Service requests for re-authentication from “iAM Smart” System.	10	
5	“iAM Smart” System notes the receipt of Online Service's request.		
6a	When Online Service client terminal and “iAM Smart” Mobile App are in different devices, <i>Online Service client terminal shows instruction to inform “iAM Smart” user to complete re-authentication in “iAM Smart” Mobile App and polls Online Service server for response returned from “iAM Smart” System.</i>		
6b	When Online Service client terminal and “iAM Smart” Mobile App are in the same device, <i>Online Service client terminal invokes “iAM Smart” Mobile App and polls Online Service server for response returned from “iAM Smart” System.</i>	6	
7	Online Service server receives Re-authentication response from “iAM Smart” System via Online Service callback API and proceeds its processing.		F
8	Online Service server informs Online Service client terminal to stop polling.		
9	Online Service saves accessToken for subsequent access to “iAM Smart” APIs.		

1.4.6 Anonymous Form Filling

Anonymous form filling describes the situation that user initiates a form filling request without prior authentication to Online Service using “iAM Smart”. Online Service can request anonymous form filling with the following steps:

Step	Description	“iAM Smart” API	Callback API
1	Online Service provides a link for anonymous form filling in the application.		
2	Online Service requests “Profile Field” and “e-ME Field” for the purpose of identity verification and form filling respectively from “iAM Smart” System.	13	
3	“iAM Smart” System notes receipt of Online Service's request.		
4a	When Online Service client terminal and “iAM Smart” Mobile App are in different device, <i>Online Service redirects client terminal to “iAM Smart” QR/App broker page.</i>	3	
4b	When Online Service client terminal and “iAM Smart” Mobile App are in the same device, <i>Online Service client terminal invokes “iAM Smart” Mobile App, Online Service App polls Online Service server for response from “iAM Smart” System.</i>	3, 6	
5	Online Service receives authCode from “iAM Smart” System via Online Service callback API.		A, B
6	Online Service gets accessToken and Tokenised ID of “iAM Smart” user.	4	
7	Online Service server obtains authorised information of “eMEFields” from “iAM Smart” System using accessToken and Tokenised ID and processes the form filling.	14	
8a	When Online Service client terminal is browser, <i>Online Service server redirects the browser to view the form filling result.</i>		
8b	When Online Service client terminal is Online Service App, <i>Online Service server informs Online Service App to stop polling and Online Service App shows the form filling result.</i>		

1.4.7 Anonymous Digital Signing

Anonymous digital signing describes the situation that user initiates a digital signing request without prior authentication to Online Service using “iAM Smart”. Online Service can request anonymous digital signing with the following steps:

Step	Description	“iAM Smart” API	Callback API
1	Online Service provides a link for anonymous digital signing in the application.		
2	Online Service calculates hash of the document (or digest for pdf document) to be signed and hash of the HKIC number, and requests digital signing from “iAM Smart” System.	15, 17	
3	“iAM Smart” System notes receipt of Online Service's request.		
4	Online Service generates a 4-digit identification code using hash of document (or digest for pdf document), hash of the HKIC number and hash of clientID of the Online Service.		
5a	When Online Service client terminal and “iAM Smart” Mobile App are in different devices, <i>Online Service client terminal shows the 4-digit identification code. For browser, Online Service redirects the browser to “iAM Smart” QR/App broker page. For Online Service App, it polls Online Service server for response from “iAM Smart” System.</i>	3	
5b	When Online Service client terminal and “iAM Smart” Mobile App are in the same device, <i>Online Service client terminal shows a 4-digit identification code. For browser, Online Service redirects the browser to “iAM Smart” QR/App broker page. For Online Service App, it invokes “iAM Smart” Mobile App and polls Online Service server for response from “iAM Smart” System.</i>	3, 6	
6	Online Service receives authCode from “iAM Smart” System via Online Service callback API.		A, B
7	Online Service gets accessToken and Tokenised ID of “iAM Smart” user.	4	
8	Online Service server obtains digital signing result from “iAM Smart” System using accessToken and Tokenised ID and computes the signed document.	16, 18	
9	Online Service server verifies and confirms digital signing result to “iAM Smart” System.	11	
10a	When Online Service client terminal is browser, <i>Online Service server redirects the browser to view digital signing result.</i>		
10b	When Online Service client terminal is Online Service App, <i>Online Service server informs Online Service App to stop polling and Online Service App shows the digital signing result.</i>		

1.4.8 Bulk Digital Signing with Service Login

The accessToken of the “iAM Smart” user that possessed by Online Service should include the authorisation scope of bulk digital signing authorisation. Online Service can request for “iAM Smart” bulk digital signing with the following steps:

Step	Description	“iAM Smart” API	Callback API
1	Online Service provides bulk digital signing using “iAM Smart” in Online Service application.		
2	Online Service requests for authCode with required authorisation scopes (e.g., “iAM Smart” authentication, Digital Signing authentication).	3, 5	A, B
3	Online Service gets accessToken and Tokenised ID of “iAM Smart” user.	4	
4	Online Service calculates hash of the documents (and/or digest for pdf documents) to be signed and requests bulk digital signing from “iAM Smart” System.	21	
5	“iAM Smart” System notes receipt of Online Service's request.		
6	Online Service generates a 6-digit identification code using hash of documents (and/or digest for pdf documents) and hash of Tokenised ID.		
7a	When Online Service client terminal and “iAM Smart” Mobile App are in different device, <i>Online Service client terminal shows the 6-digit identification code with instructions to inform “iAM Smart” user to verify the identification code and complete the “iAM Smart” bulk digital signing in “iAM Smart” Mobile App.</i>		
7b	When Online Service client terminal and “iAM Smart” Mobile App are in same device, <i>Online Service client terminal shows a 6-digit identification code and invokes “iAM Smart” Mobile App.</i>	6	
8	Online Service server receives BSQC Token and signing response from “iAM Smart” System via Online Service callback API.		I, J
9	Online Service server verifies and acknowledges bulk digital signing result to “iAM Smart” System if request completed.	24	
10	Online Service saves the accessToken and BSQC Token for subsequent access to “iAM Smart” APIs.		

Step	Description	“iAM Smart” API	Callback API
11	Online Service uses the BSQC Token to enquire bulk digital signing status or cancel bulk digital signing requests.	25, 26	

1.4.9 Anonymous Bulk Digital Signing

Anonymous bulk digital signing describes the situation that user initiates a bulk digital signing request without prior authentication to Online Service using “iAM Smart”. Online Service can request anonymous bulk digital signing with the following steps:

Step	Description	“iAM Smart” API	Callback API
1	Online Service provides a link for anonymous bulk digital signing in the application.		
2	Online Service calculates hash of the documents (and/or digest for pdf documents) to be signed and hash of the HKIC number, and requests bulk digital signing from “iAM Smart” System.	22	
3	“iAM Smart” System notes receipt of Online Service's request.		
4	Online Service generates a 6-digit identification code using hash of documents (and/or digest for pdf documents), hash of the HKIC number and hash of clientID of the Online Service.		
5a	When Online Service client terminal and “iAM Smart” Mobile App are in different devices, <i>Online Service client terminal shows the 6-digit identification code. For browser, Online Service redirects the browser to “iAM Smart” QR/App broker page.</i>	3	
5b	When Online Service client terminal and “iAM Smart” Mobile App are in the same device, <i>Online Service client terminal shows a 6-digit identification code. For browser, Online Service redirects the browser to “iAM Smart” QR/App broker page. For Online Service App, it invokes “iAM Smart” Mobile App.</i>	3, 6	
6	Online Service receives authCode from “iAM Smart” System via Online Service callback API.		A, B
7	Online Service gets accessToken and Tokenised ID of “iAM Smart” user.	4	
8	Online Service server obtains BSQC Token from “iAM Smart” System using accessToken and Tokenised ID.	23	

Step	Description	“iAM Smart” API	Callback API
9	Online Service server receives signing response from “iAM Smart” System via Online Service callback API.		J
10	Online Service server verifies and confirms bulk digital signing result to “iAM Smart” System if request completed.	24	
11	Online Service saves the accessToken and BSQC Token for subsequent access to “iAM Smart” APIs.		
12	Online Service uses the BSQC Token to enquire bulk digital signing status or cancel bulk digital signing requests.		

2. SYSTEM WORKFLOW USING “IAM SMART”

There are two types of system workflows using “iAM Smart”:

1. “Authentication to Online Services using iAM Smart” for Online Service to authenticate “iAM Smart” user who gives authorisation to Online Service to get the accessToken to access the required “iAM Smart” APIs (e.g., user authentication).
2. “Online Service business operations using iAM Smart” for Online Service to access the required “iAM Smart” APIs for completing its business operations (e.g., form filling).

For details, please refer to corresponding system workflows described in Section 3.

3. IMPLEMENTATION PROCEDURE

3.1 PREREQUISITE

- (a) Work out the usage of “iAM Smart” for the Online Service application (e.g., form filling, digital signing, etc.) and determine the authorisation scopes (e.g., form filling authorisation for form filling) required
- (b) Design the lifecycle management of the API data encryption/decryption key (e.g., get, renewal)
- (c) Apply the digital certificates for encipherment as KEK (i.e., for API data encryption/decryption key)
- (d) Study the specifications of relevant “iAM Smart” APIs and Online Service callback APIs
- (e) Prepare the redirect URIs and/or URL scheme (or Universal Link/App Link) (for invoking Online Service App to return authorisation code to Online Service server) for each Online Service callback endpoint to be implemented for the Online Service application
- (f) Work out and enable the bi-directional network connectivity between Online Service server and “iAM Smart” System
- (g) Register Online Service application to “iAM Smart” System and get the corresponding client ID, client secret and/or URL scheme (or Universal Link/App Link) (for invoking “iAM Smart” Mobile App)
- (h) Configure client authentication for “iAM Smart” System in Online Service callback endpoint (optional)
- (i) Study the relevant system workflows and start implementation

3.2 ONLINE SERVICE AND “IAM SMART” SYSTEM INTERACTION DESIGN

3.2.1 Interaction between Online Service and “iAM Smart” Mobile App

The “iAM Smart” API is designed to provide better user experience when two forms of Online Service client terminal, Online Service App and Online Service Website (i.e., browser) are interacting with “iAM Smart” Mobile App in the same device and different devices. There are three scenarios for the interactions: (1) *for authentication using “iAM Smart”* (2) *for authorisation of anonymous “iAM Smart” API access* (e.g., Anonymous Form Filling) and (3) *for performing Online Service business operations with authentication using “iAM Smart”* (e.g., Form Filling by “iAM Smart”) as follows:

1. For authentication using “iAM Smart”, Online Service should invoke the “iAM Smart” API “Request QR Page”. By default, this API returns a webpage with a QR code for scanning by “iAM Smart” Mobile App in different device.
 - When Online Service detects that the Online Service client terminal is a browser running in personal computer (“PC”), the API request parameter “brokerPage” can be kept as “false” and QR Code webpage will be shown.
 - When Online Service detects that the Online Service client terminal is a browser running in mobile platform, the API request parameter “brokerPage” can be set to “true”. The broker page of “iAM Smart” System will show a button to invoke the “iAM Smart” Mobile App. If “iAM Smart” Mobile App is not installed in the same device, QR Code webpage will be shown.
 - When Online Service client terminal is Online Service App, it should detect the existence of “iAM Smart” Mobile App in the device using program code. If the “iAM Smart” Mobile App is installed in the same device, Online Service App should invoke “iAM Smart” Mobile App using URL Scheme with the API parameters. Otherwise, Online Service App should make use the device browser to invoke the “iAM Smart” API with “brokerPage” set to “false” such that QR Code webpage will be shown.
2. For authorisation of anonymous “iAM Smart” API access, Online Service will first invoke the relevant anonymous “iAM Smart” APIs and get the “ticketID”. Online Service will then invoke the “iAM Smart” API “Request QR Page” using the “ticketID” to get the accessToken and Tokenised ID for obtaining the result of the relevant anonymous “iAM Smart” API. Same detection for client terminal by Online Service as in authentication using “iAM Smart”, is applied.

3. To perform Online Service business operations with authentication using “iAM Smart” such as “Form Filling”, “iAM Smart” System will return the immediate API response with parameters “authByQR” and “ticketID” (optional) to Online Service server before the final result is returned to Online Service using Online Service callback API. Online Service should determine the next action based on “authByQR”:
 - When it is “false”, Online Service client terminal (both browser and Online Service App) should invoke the “iAM Smart” Mobile App using URL Scheme with the “ticketID”. Online Service client terminal should keep polling Online Service server for the result returned to relevant Online Service callback API.
 - Otherwise, Online Service client terminal should show instructions to inform “iAM Smart” user to complete the request in “iAM Smart” Mobile App (“iAM Smart” System will also notify “iAM Smart” Mobile App using push message) and keep polling Online Service server for the result returned to relevant Online Service callback API.

The table below summarises the scenarios:

Scenario	Online Service App/“iAM Smart” Mobile App	Online Service Website/“iAM Smart” Mobile App
Authentication / Authorisation for anonymous “iAM Smart” API (Same device)	<ul style="list-style-type: none"> Determine “iAM Smart” Mobile App existence by program code Invoke “iAM Smart” Mobile App using URL scheme with API parameters 	<ul style="list-style-type: none"> Determine browser platform by program code Invoke “Request QR Page” with “brokerPage” set to “true”
Authentication / Authorisation for anonymous “iAM Smart” API (Different device)	<ul style="list-style-type: none"> Determine “iAM Smart” Mobile App existence by program code Invoke “Request QR Page” using device browser with “brokerPage” set to “false” 	<ul style="list-style-type: none"> Determine browser platform by program code: <ul style="list-style-type: none"> If browser in PC, invoke “Request QR Page” with “brokerPage” set to “false” If browser in mobile, invoke “Request QR Page” with “brokerPage” set to “true”
Business operations with authentication using “iAM Smart” (Same device)	<ul style="list-style-type: none"> Determine by “authByQR” is “false” Invoke “iAM Smart” Mobile App using URL scheme with “ticketID” 	
Business operations with authentication using “iAM Smart”	<ul style="list-style-type: none"> Determine by “authByQR” is “true” Show instructions to inform “iAM Smart” user to complete the request in “iAM Smart” Mobile App and keep polling Online Service server for result 	

(Different device)	returned to relevant Online Service callback API
--------------------	--

Specific for the situation when ***“iAM Smart” Mobile App and Online Service App are in the same device,***

- (a) In the scenario (2) when invoking “iAM Smart” Mobile App (i.e., “iAM Smart” API: Open “iAM Smart” Mobile App for Getting Context), Online Service App is required to provide “source” and “redirectURI” parameters. Online Service can specify “source” parameter as “App_Scheme” (for URL Scheme) or “App_Link” (for iOS Universal Link/Android App Link) depending on its implementation method and the information submitted at Online Service registration. The “source” provides information for how “iAM Smart” Mobile App can invoke Online Service App to return Online Service the authCode for authorisation of anonymous “iAM Smart” API access (i.e., “iAM Smart” API: Callback with authCode to Online Service App):
- If Online Service specify “source” = “App_Scheme”, then “iAM Smart” System will consider it as Online Service App and using URL Scheme to invoke the Online Service App. Online Service should specify the corresponding URL Scheme in “redirectURI” and “iAM Smart” System will verify it with the URL Scheme submitted at Online Service registration.
 - If Online Service specify “source” = “App_Link”, then “iAM Smart” System will consider it as Online Service App and using Universal Link/App Link to invoke the Online Service App. Online Service should specify the corresponding Universal Link/App Link in “redirectURI” and “iAM Smart” System will verify it with the Universal Link/App Link submitted at Online Service registration.
- (b) In the scenario (3) when Online Service backend invokes the required “iAM Smart” API (e.g., “iAM Smart” API: Request Form Filling), Online Service backend is required to provide “source” parameter. Online Service can specify “source” parameter as “App_Scheme” (for URL Scheme) or “App_Link” (for iOS Universal Link/Android App Link) depending on its implementation method and the corresponding scheme or link information submitted to “iAM Smart” System during Online Service registration. “iAM Smart” System will query such scheme or link information using the “source” provided for “iAM Smart” Mobile App to invoke Online Service App.

3.2.2 Process of Common API Request and Response Parameters

There are common parameters defined in the HTTP request header, HTTP request body of “iAM Smart” POST APIs. Common parameters are also defined in the “iAM Smart” POST API response body and Online Service callback API request body. This section describes

how these common parameters should be prepared and processed by Online Service. Relevant information can be found in the “iAM Smart” API Specification.

Common parameters in the HTTP request header and body of “iAM Smart” POST APIs:

Request	Parameters	Description
Request Header	clientID	The Client Identity provided by “iAM Smart” System during Online Service registration.
	signatureMethod	Name of signature algorithm used to compute the POST request signature Currently only support HmacSHA256 and its value must be set to “HmacSHA256”, case sensitive.
	timestamp	The timestamp is a number expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT. It must be greater than or equal to the timestamp of previous request submitted by Online Service. “iAM Smart” System compares the timestamp with system time and treats it as valid when the difference is within 60 seconds.
	nonce	A unique, non-repetitive random string for each request submitted by Online Service. It is used with timestamp to prevent Replay Attack. Its value cannot be repeated within 60 seconds.
	signature	It is a HmacSHA256 digital signature signed by Online Service using its client secret (paired with the clientID) provided by “iAM Smart” System during Online Service registration. The object to be signed includes the parameters concatenating in the following sequence: <i>clientID + signatureMethod + timestamp + nonce + encrypted request body</i> All parameters are case sensitive.
Request Body	content	Its value is created using API data encryption. It is BASE64 encoded with the following components concatenating in the following sequence: Content value: <i>4 bytes IV length + IV + ciphertext</i> IV is the initialisation vector of the encryption algorithm “AES/GCM/NoPadding” used in API data encryption. The ciphertext is the encrypted textual content of request body of the “iAM Smart” POST API. Details of API data encryption can be referred to Section 3.3.

Common parameters in the “iAM Smart” POST API response body and Online Service callback API request body:

Response/ Callback	Parameters	Description
Response/ Callback Body	txID	It is a unique transaction ID allocated by “iAM Smart” System for the Online Service request. Online Service is suggested to record this transaction ID for future reference or problem diagnostics.
	code	It is the return code of the processing result of the “iAM Smart” API invoked by Online Service. For details, please refer to “iAM Smart” API Specification.
	message	It is the text description of the “code” parameter.
	secretKey	It exists only in Online Service callback API request body. Its value is the encrypted CEK of Online Service (in AES256 format) using Online Service’s KEK based on RSA algorithm. It is BASE64 encoded. Details of CEK and KEK can be referred to Section 6.1 of “iAM Smart” API Specification.
	content	Its value is created using API data encryption. It is BASE64 encoded with the following components concatenating in the following sequence: Content value: <i>4 bytes IV length + IV + ciphertext</i> IV is the initialisation vector of the encryption algorithm “AES/GCM/NoPadding” used in API data encryption. The ciphertext is the encrypted textual content of the JSON data inside “content” of the “iAM Smart” POST API response body and Online Service callback APIs request body. Details of API data encryption can be referred to Section 3.3.

Common parameters in JSON data of the “iAM Smart” APIs request body:

Request/ Callback	Parameters	Description
Request Body	businessID	It is a unique identifier generated by Online Service for identifying the business request of Online Service for invoking the “iAM Smart” API. It will be returned to Online Service via the corresponding Online Service callback API. For Online Service, it is used to match the result returned via Online Service callback API and thus should not be repeated. It is recommended to use 36-byte UUID number (ASCII character set) for this parameter.
	accessToken	It is the accessToken Online Service retained after authentication of the “iAM Smart” user and must be still valid. accessToken can be used multiple of times before expiry except that requested through the anonymous “iAM Smart” APIs which can only be used once.

Request/ Callback	Parameters	Description
	openID	It is the Tokenised ID of the “iAM Smart” user. Online Service should ensure the accessToken and Tokenised ID are in pair and belongs to the target “iAM Smart” user for the request.
	source	It can be “App_Scheme” (for URL Scheme), “App_Link” (for iOS Universal Link/Android App Link), or the user agent’s name value of the browser. Please refer to Appendix B of the “iAM Smart” API Specification for the list of source values. It is used by “iAM Smart” System to determine how to process the interaction with the Online Service client terminal after receiving the “iAM Smart” API call. Please refer to Section 3.2.1 for details.
	redirectURI	It is the callback URI of Online Service. It is used by “iAM Smart” System to return the result via Online Service callback API and its value must match with the value provided by Online Service during registration to “iAM Smart” System.
	state	It is an optional parameter to prevent CSRF attack. The value of “state” is defined and managed by Online Service. It is recommended to use 36-byte UUID number (ASCII character set) or less with random value. It will be returned to Online Service via the corresponding callback API and Online Service is suggested to verify this value to prevent CSRF attack.

3.2.3 Process of Common Errors Returned from “iAM Smart” APIs

There are two tiers of errors returned from “iAM Smart” System: HTTP error and application error returned from “iAM Smart” APIs (given in parameter “code” or “error_code” in the POST response, Online Service callback API and URL Scheme). For application error, the parameter “message” presents the error description.

Common application errors are listed below:

Error Code	Error Description	Parameters in POST request	Suggested Action
Common Error Code			
D20000	unknown exception	N/A	Unknown error happened in “iAM Smart” System. Retry later or contact “iAM Smart” System administrator if problem persists.

Error Code	Error Description	Parameters in POST request	Suggested Action
D20001	parameter { %s } is missing	All parameters in request	Check parameter in { %s } is provided
D20002	empty parameter { %s }	All parameters in request	Check parameter in { %s } provided has correct value and format
D20003	invalid parameter { %s }	All parameters in request	Check parameter in { %s } provided is defined in the corresponding API
D20004	duplicated request	“timestamp”, “nonce”	Check same “timestamp” and “nonce” are used in different requests submitted
D20005	unsupported signature method	“signatureMethod”	Check signature method is supported by “iAM Smart” System and its value must be “HmacSHA256”, case-sensitive
D20006	signature verification failed	“signature”	1. Check correct client secret is used to calculate signature 2. Check any missing fields or incorrect field sequence when generating signature
D20007	unsupported source	“source”	For Online Service Website, check browser user agent value is one of the browsers supported by “iAM Smart” System (Please refer to Appendix B of “iAM Smart” API Specification for details.) For Online Service App, check the source is “App_Scheme” or “App_Link” (Please refer to Appendix B of “iAM Smart” API Specification for details).
D20008	unregistered redirectURI	“redirectURI”	Check value is registered with “iAM Smart” System
D20009	accessToken not exist or expired	“accessToken”	1. Check correctness of accessToken 2. Check accessToken is not expired
D20010	openID not exist	“openID”	Check validity of openID
D20011	duplicated businessID	“businessID”	Check uniqueness of businessID
D20012	insufficient scope	“scope”	Check authorisation scope(s) requested is/are sufficient for invoking the corresponding “iAM Smart” API
D20012	Forbidden	N/A	Check your endpoint is requested in the application form.

Error Code	Error Description	Parameters in POST request	Suggested Action
D20015	Scope mismatch	N/A	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	N/A	Check whether Online Service public key certificate (i.e., KEK) has been registered with “iAM Smart” System or it is expired
D30002	content encryption key not exist or expired	N/A	1. Check validity of CEK 2. Check CEK is not expired 3. Check value of “content” is BASE64 encoded
D30003	encryption exception	N/A	Unknown error when performing API data encryption in “iAM Smart” System: please contact “iAM Smart” System administrator if problem persists
D30004	decryption exception	N/A	API data decryption failed: check validity of encrypted “content”, encryption process and validity of initialisation vector IV

A list of common HTTP errors which may be returned from “iAM Smart” System:

Error Code	Error Description	Suggested Action
302 Found	The requested resource resides temporarily under a different URI.	Browser directs or initiates request to new address.
400 Bad Request	The request could not be understood by the server due to malformed syntax.	Check syntax and correctness of the HTTP request and its parameters
401 Unauthorised	The request has not been applied because it lacks valid authentication credentials for the target resource.	No authorisation: please contact “iAM Smart” System Administrator
403 Forbidden	The server understood the request but is refusing to fulfil it.	1. “iAM Smart” System verifies the request header anti-replay parameter to be empty or illegal: check parameters and re-request 2. “iAM Smart” System request is detected as Replay: check if timestamp and nonce parameters meets the requirements

Error Code	Error Description	Suggested Action
		<p>3. “iAM Smart” System verifies the signature verification parameter to be empty or illegal: check the parameters and re-request</p> <p>4. “iAM Smart” System signature verification fails: check if the signature rule is incorrect and re-request after modification</p> <p>5. “iAM Smart” System business data decryption fails: check if the encryption rule is incorrect and re-request after modification</p>
404 Not Found	The server has not found anything matching the Request-URI.	Check if interface address is correct
429 Too Many Requests	The user has sent too many requests in a given amount of time.	“iAM Smart” System is busy: resubmit request after a period of time (e.g., 30s) or contact “iAM Smart” System administrator if problem persists
500 Internal Server Error	The server encountered an unexpected condition which prevents it from fulfilling the request.	Retry later, or contact “iAM Smart” System administrator
503 Service Unavailable	The server is currently unable to handle the request due to a temporary overloading or maintenance of the server.	Retry later, or contact “iAM Smart” System administrator

3.3 API DATA ENCRYPTION AND DECRYPTION

HTTPS will be used to secure communication channels between Online Services and “iAM Smart” System. However, there are chances that the data transferred by HTTPS could be eavesdropped or manipulated by malicious service providers in some special situations. To protect the data in transit, an additional layer of data encryption will be applied to all APIs POST request and response body (except the API that Online Service request for getting the data encryption key). The response body of all Online Service callback POST APIs invoked by “iAM Smart” System will also be encrypted using the same key and method as described in this section. The process is as below:

1. Online Service requests for data encryption key from “iAM Smart” System;
2. “iAM Smart” System generates an AES256 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;

(Note: For the callback to Online Service, considering that CEK may time out when “iAM Smart” System received the request and consequently be regenerated, the encrypted CEK (probably new) will be returned to Online Service in the Online Service callback API. Online Service should use the CEK in callback body for data decryption, but does not retain this CEK.)

6. If the generated key is expired, Online Service have to request a new key and repeat the above process.

3.3.1 Workflow for Encryption and Decryption

The following diagram describes the detailed workflow 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:

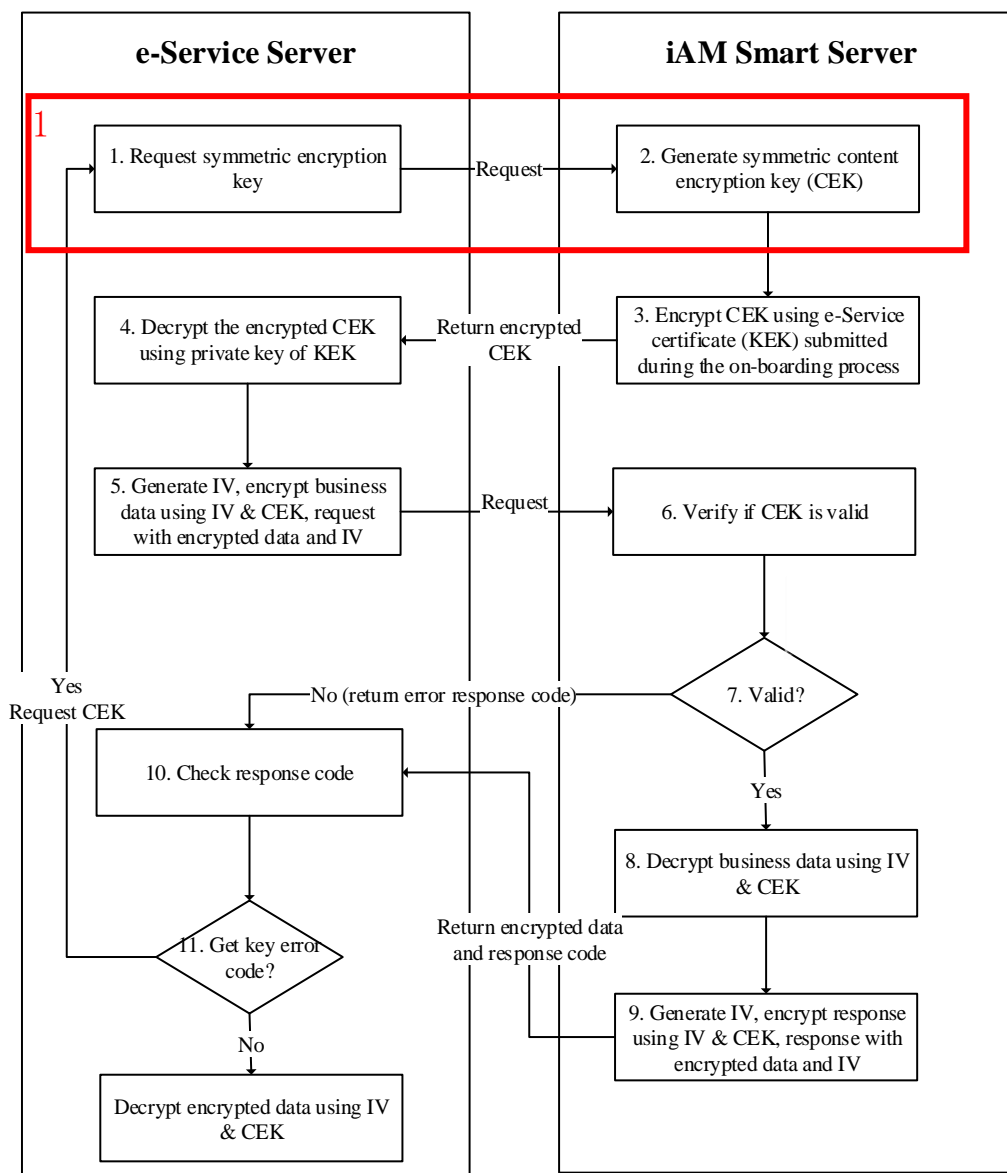


Figure-1 Application of Content Encryption Key and Procedures of Encryption and Decryption (Perform by Online Service)

Step 1. Online Service requests for data encryption key from “iAM Smart” System through invoking “iAM Smart” API;

“iAM Smart” API (POST: Request Symmetric Content Encryption Key);

Step 2. “iAM Smart” System generates an AES256 content encryption key (“CEK”) for the Online Service and saves it in “iAM Smart” System;

- Step 3. “iAM Smart” System encrypts the CEK using the public key certificate A (“KEK”) provided by the Online Service in registration to “iAM Smart” System based on RSA algorithm, and returns the encrypted CEK, key expiry time to the Online Service;
- Step 4. Online Service decrypts the encrypted CEK using the private key of its KEK and saves it and its key expiry time;
- Step 5. Before Online Service calling APIs of “iAM Smart” System, Online Service should check whether its CEK is still valid and use it to encrypt all the JSON data of the POST request body as ciphertext using encryption algorithm “AES/GCM/NoPadding”. This algorithm requires an initialisation vector (“IV”) which is provided by Online Service. The ciphertext, IV and length of IV are then concatenated in the following sequence and BASE64 encoded to formulate the value of JSON name/value pair called “content”:

Content value: *4 bytes IV length + IV + ciphertext*

- Step 6-7. After receiving the Online Service request, “iAM Smart” System first checks whether the KEK, CEK of the Online Service exist or expired and returns with corresponding error code if required;
- Step 8. “iAM Smart” System extracts the initialisation vector IV from the Online Service request and decrypts the JSON data of the POST request body using Online Service's CEK and IV. “iAM Smart” System then validates the request parameters and returns error response to Online Service if required;
- Step 9. After “iAM Smart” System has processed the request, it generates its initialisation vector IV and encrypts all the JSON data of POST response (except “txID”, “code” and “message”) as ciphertext using the valid Online Service's CEK (i.e., CEK is not expired) and IV by the same encryption algorithm (“AES/GCM/NoPadding”):
- For “iAM Smart” POST API response, if CEK does not exist or expires, corresponding error code will be returned.
 - For Online Service callback, if CEK is expired, “iAM Smart” System will re-generate a new CEK for the Online Service to perform the encryption.

The ciphertext of encrypted POST response / callback JSON data, IV and length of IV are then concatenated in the following sequence and BASE64 encoded to formulate the value of JSON name/value pair called “content”:

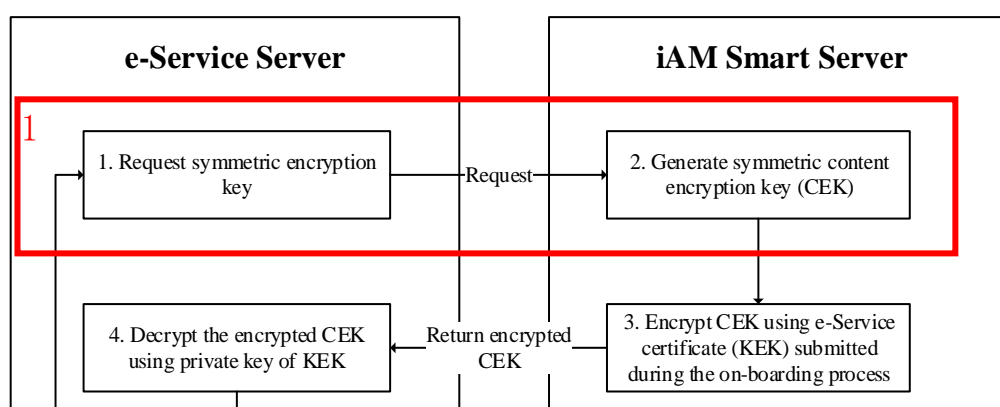
Content value: *4 bytes IV length + IV + ciphertext*

For Online Service callback, the CEK used for this encryption (e.g., new CEK) will also be encrypted using Online Service's KEK and BASE64 encoded, and

will be returned to Online Service as part of the POST callback body as JSON name/value pair called “secretKey”;

Step 10-11. For the response received, the Online Service should first check whether the return code contains any key-related error code such as “key does not exist or has been expired”. For “iAM Smart” POST API response, if there are no such key-related errors, Online Service should use its CEK and IV (extracted from “content”) to decrypt the response data. For Online Service callback, Online Service should decrypt the CEK from “secretKey” parameter in the callback body using its KEK’s private key and then use this CEK and IV (extracted from “content”) to decrypt the callback data. Online Service should request for a new CEK when it is expired.

3.3.1.1 Implementing (1) POST: Request Symmetric Content Encryption Key



Pre-conditions

- Online Service can access the private key of its KEK.
- Online Service has registered to “iAM Smart” System and uploaded its KEK.

Post-conditions

- Online Service decrypts CEK received from “iAM Smart” System using the private key of its KEK and saves the CEK and its key expiry time.

Error conditions

- For common errors, please refer to Section 3.2.3.

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Online Service should ensure the KEK registered to “iAM Smart” System is valid and keeps the private key of KEK. The KEK used for this “iAM Smart” API should be bound to the same clientID allocated to Online Service by “iAM Smart” System.
- If the current CEK for Online Service is not expired, “iAM Smart” System will return the same CEK to Online Service with “issueAt” and “expiresIn” unchanged.
- For common parameters in request and response, please refer to Section 3.2.2.
- Response parameter “secretKey”: It is a CEK generated by “iAM Smart” System for the Online Service and is encrypted using the RSA algorithm and the latest Online Service's KEK. Online Service should use the private key of its KEK to decrypt “secretKey” to get the CEK.
- Response parameter “pubKey”: It is the latest Online Service’s KEK that is used to encrypt the CEK to form value of “secretKey”. Online Service may make use “pubKey” to locate the corresponding private key.
- Response parameter “issueAt”: It is the issue time of CEK expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.
- Response parameter “expiresIn”: It is the validity period of CEK expressed in milliseconds.

3.3.1.2 Implementing (2) POST: Revoke Symmetric Content Encryption Key

This API allows Online Service to revoke symmetric Content Encryption Key when necessary.

Pre-conditions

- Nil

Post-conditions

- Nil

Error conditions

- For common errors, please refer to Section 3.2.3.

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Each Online Service can only request for one CEK. When CEK was revoked, Online Service is required to request for a new CEK.

3.3.2 Encryption & Decryption Scope and Examples

Please refer to “iAM Smart” API Specification.

3.3.3 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 6.3.1.1 of “iAM Smart” API Specification 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 6.3.1.1 of “iAM Smart” API Specification 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 6.3.1.1 of “iAM Smart” API Specification)
- “iAM Smart” API Online Service has to use the old private key to decrypt the CEK until the expiry time. Online service shall match the public key returned from the “Request Symmetric Content Encryption Key” API response to determine which private key should be used for the decryption
- 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 6.3.1.1 of “iAM Smart” API Specification 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 6.3.1.2 and 6.3.1.1 respectively of “iAM Smart” API Specification or to request a new CEK after its expiry through the same APIs.

3.4 WORKFLOWS FOR AUTHENTICATION

3.4.1 Scenario 1: Authentication (Online Service Website in Different Device)

The sequence diagram below shows how 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.

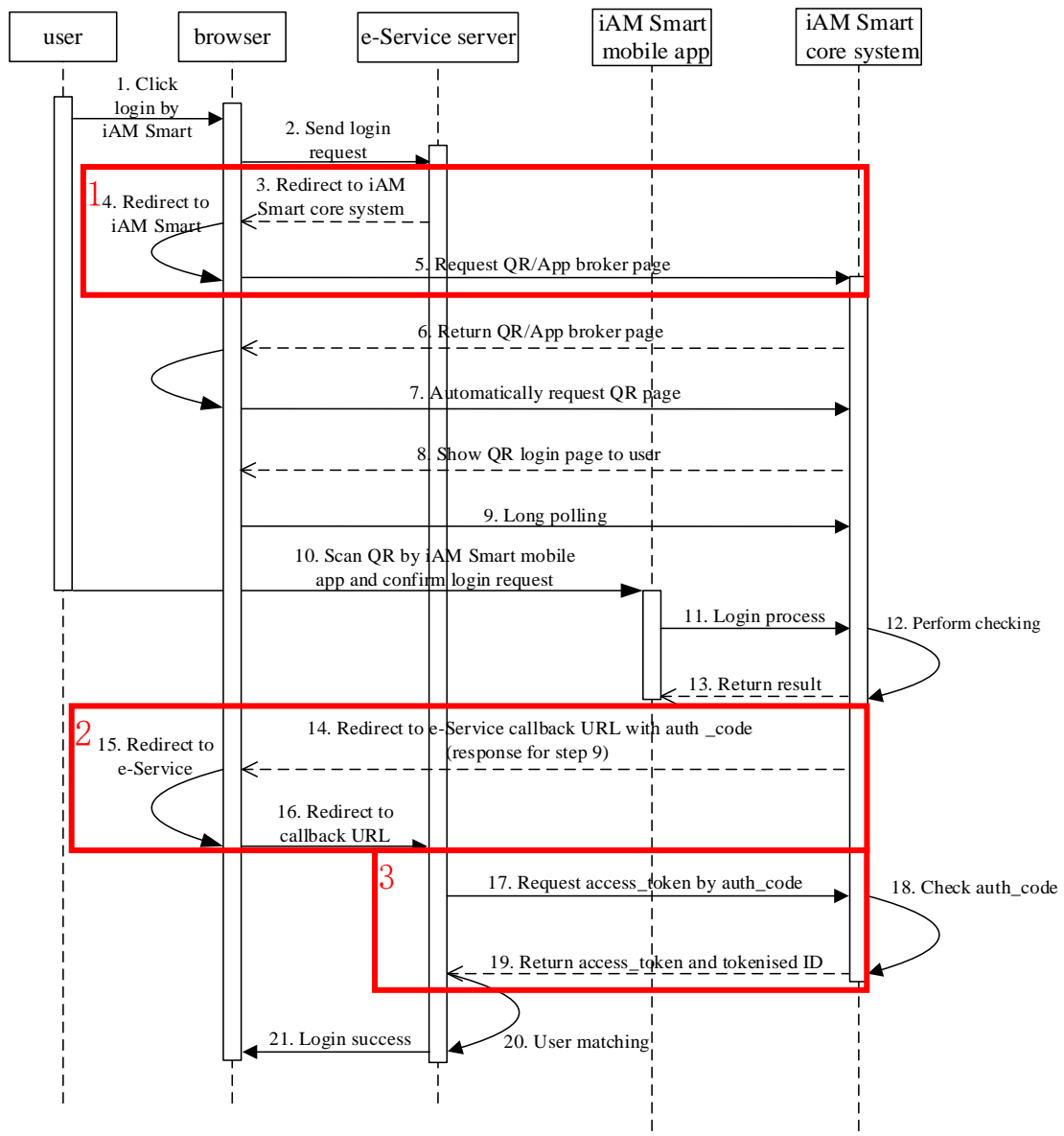


Figure-2 Authentication (Online Service Website in Different Device)

- Step 1. User clicks “Login by iAM Smart” button on Online Service Website;
- Step 2. Online Service Website initiates login request to Online Service Server with the browser's user agent value (for use as value for request parameter “source”);
- Step 3-5. Online Service Server prepares the request parameters such as “clientID”, “redirectURI”, “source”, “scope”, etc. and constructs the GET request to invoke the “iAM Smart” API using browser redirection;

“iAM Smart” API (GET: Request QR page)

- Step 6-8. “iAM Smart” System returns the broker page (if Online Service has set “brokerPage” to “true”) and after the broker page fails to find the “iAM Smart” Mobile App, it will request QR page to be displayed in browser. If Online Service does not request for broker page (i.e., no broker page will be returned), the browser will directly display QR Code page;
- Step 9. QR Code page of “iAM Smart” System polls “iAM Smart” System for the QR Code status;
- Step 10-11. “iAM Smart” user logs in “iAM Smart” Mobile App to scan the QR Code and confirms Online Service's login request;
- Step 12-13. “iAM Smart” System verifies the validity of QR code and other necessary information, and responses the login result to “iAM Smart” Mobile App (e.g., login success and inform “iAM Smart” user to proceed in Online Service, invalid / expired QR code, etc.);
- Step 14-16. QR Code page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e., response for the polling in Step 9) which includes authCode or any error code (e.g., “iAM Smart” user rejects the login request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

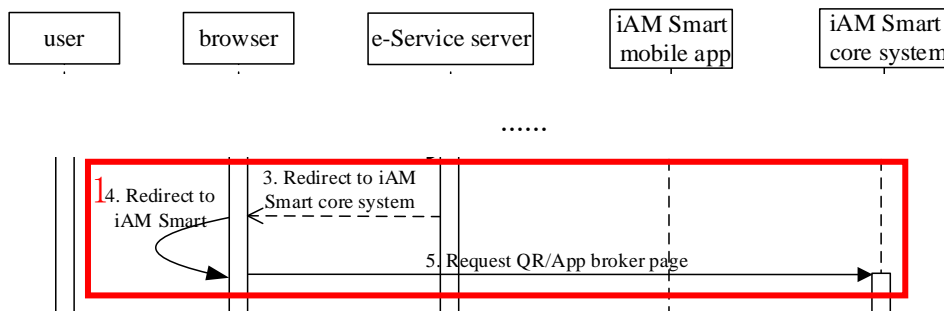
Online Service Callback API (GET: Callback with authCode to Online Service Server)

- Step 17-19. When Online Service Server gets the authCode, it should invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

“iAM Smart” API (POST: Request accessToken & Tokenised ID)

- Step 20. Online Service Server then performs user matching using the Tokenised ID with its user repository and determines the result of this login request;
- Step 21. The Online Service Website shows the login result (e.g., successful when Tokenised ID matched with a user account of Online Service).

3.4.1.1 Implementing (1) GET: Request QR page



Pre-conditions

- Online Service should determine all the authorisation scopes required for this authentication request.
- Online Service should detect the browser's user agent name.
- Online Service Website can use the optional “brokerPage” parameter (Details please refer to Section 3.2.1), or determine whether the browser and “iAM Smart” Mobile App are on different devices or not by using its program code (e.g., the browser's agent name, client configure check). For browser type supported by “iAM Smart” System, please refer to Appendix B of “iAM Smart” API Specification.
- Online Service should determine if it would generate the optional “state” request parameter to prevent CSRF attack. The “state” will be returned to Online Service for checking through the Online Service callback API for receiving the authorisation code from “iAM Smart” System in Step 16.

Post-conditions

- The browser will show the QR Code page of “iAM Smart” System with a “Return” button to allow user to return to Online Service.

Error conditions

- This “iAM Smart” API does not return JSON response (i.e., no application error will return). For common errors, please refer to Section 3.2.3.

Request Parameters

- Please refer to “iAM Smart” API Specification.

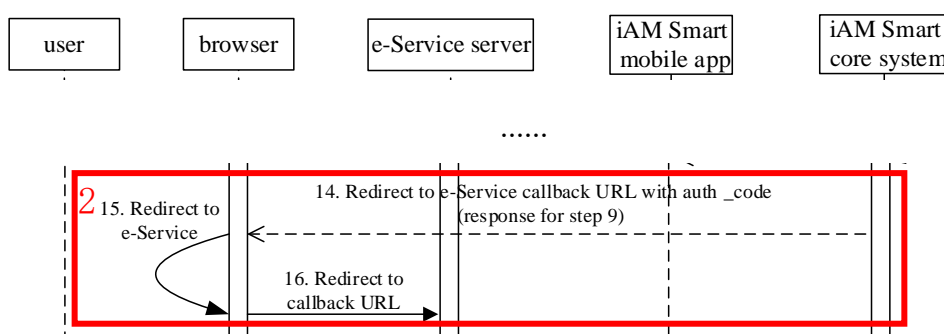
Notes

- For common parameters, please refer to Section 3.2.2.
- Request parameter “responseType”: Value must be 'code'.
- Request parameter “source”: Value should be the browser's user agent value.
- Request parameter “redirectURI”: This is Online Service URL for receiving the authorisation code. The value should be URL encoded. For details, please referred to

“iAM Smart” API Specification “Callback with authCode to Online Service Server”. The value must be the same as provided during Online Service registration.

- Request parameter “scope”: All authorisation scopes required for this authentication request should be specified. Multiple values should be separated by space (e.g., eidapi_auth ediapi_eMe eidapi_sign). The value should be URL encoded.
- Request parameter “ticketID”: Not required in this scenario.
- Request parameter “lang”: Display language of the QR code page.
- Request parameter “state”: The value should be URL encoded.
- Request parameter “brokerPage”: Value should be set to “false” in this scenario. For details, please refer to Section 3.2.1.

3.4.1.2 Implementing (2) GET: Callback with authCode to Online Service Server



Pre-conditions

- “iAM Smart” user has scanned the QR Code using “iAM Smart” Mobile App and authorised Online Service's authentication request.
- “iAM Smart” user can also reject the authentication request in “iAM Smart” Mobile App.

Post-conditions

- authCode will be expired in 1 minute and Online Service can only use the authCode once for requesting the accessToken from “iAM Smart” System.

Error conditions

- This Online Service callback API is a HTTP GET request. If parameter “error_code” is returned, it means the authentication request is failed.

Error Code	Error Description	Suggested Action
error_code - D40000	User cancelled authentication request	Inform user the authentication request is cancelled
error_code - D40001	User rejected authentication request	Inform user the authentication request is rejected
error_code - D40002	Failed to authenticate	Inform user the authentication request is failed and retry later

The error_code D40003 (authentication request timeout) does not appear in this scenario. For the QR code timeout or request confirmation timeout in the “iAM Smart” Mobile App, message will be prompted in the corresponding user interface.

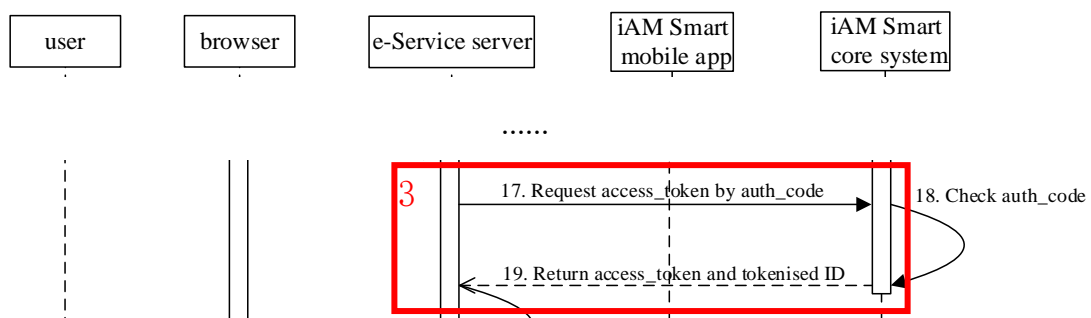
Callback Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Callback parameter “businessID”: Not required in this scenario.
- Callback parameter “code”: It is the authCode for requesting the accessToken from “iAM Smart” System. Its validity is 1 minute and can only be used once.
- If “iAM Smart” user rejects the authentication request or other errors occur, “error_code” instead of “code” will be returned.

3.4.1.3 Implementing (3) POST: Request accessToken & Tokenised ID



Pre-conditions

- Online Service should request the accessToken using a valid authCode. authCode is valid for 1 minute and should not be used more than once.
- API data encryption is required.

Post-conditions

- API data decryption is required.
- Online Service should save the accessToken and Tokenised ID for subsequent access to other “iAM Smart” APIs.
- The validity period of accessToken is given in the response parameter “expiresIn”.
- The accessToken can be used multiple of times before expiry.
- Depending on its business needs, Online Service may also keep and check other response parameters such as “userType”, “scope” to determine subsequent actions to access to other “iAM Smart” APIs.

Error conditions

- For common errors, please refer to Section 3.2.3. Other error for this “iAM Smart” API includes:

Error Code	Error Description	Suggested Action
code - D40004	authCode not exist or expired	Check authCode, or redo authentication

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Request parameter “code”: It is the authCode for requesting the accessToken from “iAM Smart” System. Its validity is 1 minute and can only be used once.
- Request parameter “grantType”: Value must be “authorization_code”.
- Response Parameters “accessToken”, “tokenType”, “issueAt”, “expiresIn”: These are the accessToken information of the “iAM Smart” user. The accessToken is issued by “iAM Smart” System at time specified by “issueAt” and valid for the period specified in “expiresIn”. The “tokenType” must be “Bearer”.
- Response parameter “openID”: It is the Tokenised ID of the “iAM Smart” user for the Online Service. The Tokenised ID of an “iAM Smart” user for different Online Services are different. Online Service can use it to verify if the “iAM Smart” user has linked with any user account in Online Service user repository.
- Response parameter “lastModifiedDate”: It is the last update timestamp of “iAM Smart” user's CFD information in his/her “iAM Smart” Profile.
- Response parameter “userType”: It is the type of the “iAM Smart” account owned by the “iAM Smart” user. It is either “default” (i.e., default “iAM Smart”) or “sign” (i.e., “iAM Smart” with digital signing function) and only “userType” with value “sign” is given an “iAM Smart” e-Cert to carry out digital signing operation.
- Response parameter “scope”: It is the authorisation scope of the accessToken and it should match with the authorisation scope requested by Online Service in Step 5.

3.4.2 Scenario 2: Authentication (Online Service Website in Same Device)

The sequence diagrams below show how an Online Service website leverages the “iAM Smart” System to perform user authentication when the “iAM Smart” Mobile App is installed in the same mobile device.

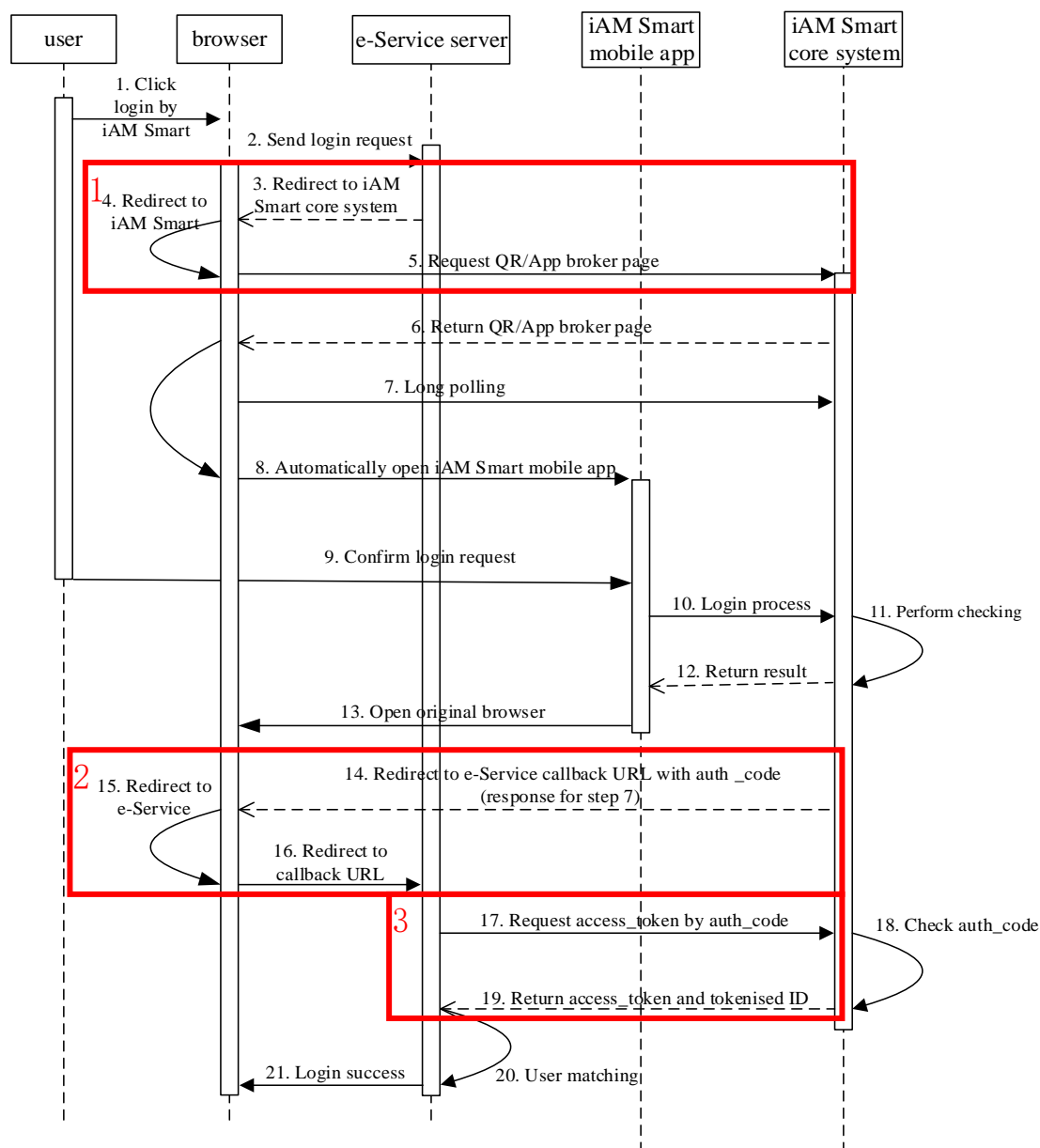


Figure-3 Authentication (Online Service Website in Same Device)

- Step 1. User clicks “Login by iAM Smart” button on Online Service Website;
- Step 2. Online Service Website initiates login request to Online Service Server with the browser's user agent value (for use as value for request parameter “source”);
- Step 3-5. Online Service Server prepares the request parameters such as “clientID”, “redirectURI”, “source” (set as browser's user agent value), “scope”, “brokerPage”

(set as “true”). etc. and constructs the GET request to invoke the “iAM Smart” API using browser redirection;

“iAM Smart” API (GET: Request QR Page)

- Step 6. “iAM Smart” System returns the broker page to the Online Service Website;
- Step 7. Broker page of “iAM Smart” System polls “iAM Smart” System for further action;
- Step 8-9. Broker page invokes the “iAM Smart” Mobile App in the same device automatically and sends it the relevant parameters. “iAM Smart” user logs in the “iAM Smart” Mobile App;
- Step 10. “iAM Smart” user confirms the Online Service login request in “iAM Smart” Mobile App and “iAM Smart” Mobile App submits the login request to “iAM Smart” System;
- Step 11-13. “iAM Smart” System verifies the validity of necessary information, and responses the login result to “iAM Smart” Mobile App. “iAM Smart” Mobile App invokes the original Online Service browser (i.e., using the browser's user agent value submitted);
- Step 14-16. Broker page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e., response for the polling in Step 7) which includes authCode or any error code (e.g., “iAM Smart” user rejects the login request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

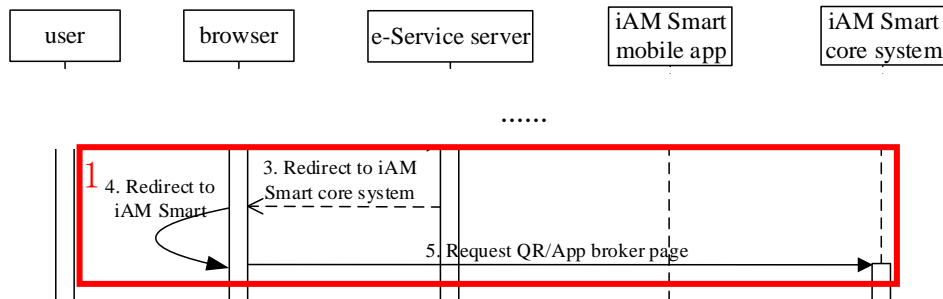
Online Service Callback API (GET: Callback with authCode to Online Service Server)

- Step 17-19. When Online Service Server gets the authCode, it should invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

“iAM Smart” API (POST: Request accessToken & Tokenised ID)

- Step 20. Online Service Server performs user matching using the Tokenised ID with its user repository and determines the result of this login request;
- Step 21. The Online Service Website shows the login result (e.g., successful when Tokenised ID matched with a user account of Online Service).

3.4.2.1 Implementing (1) GET: Request QR Page



Pre-conditions

- Please refer to Section 3.4.1.1.

Post-conditions

- Online Service browser will display the broker page returned by “iAM Smart” System and “iAM Smart” Mobile App is launched automatically.

Error conditions

- Please refer to Section 3.4.1.1.

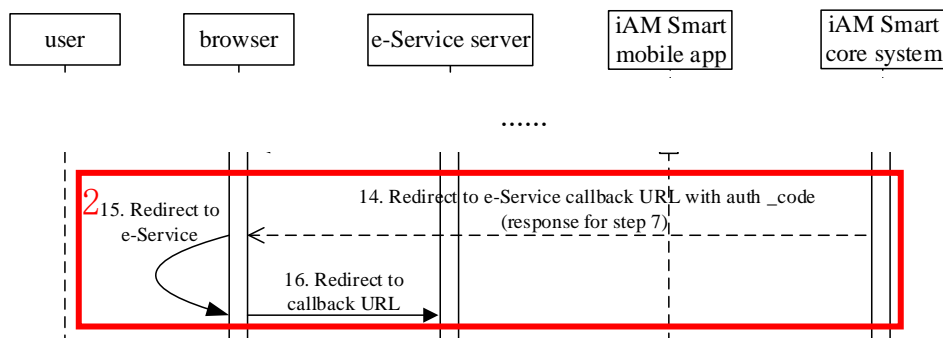
Request Parameters

- Please refer to Section 3.4.1.1.

Notes

- Please refer to Section 3.4.1.1 except the following parameter:
 - Request parameter “brokerPage”: Value should be set to “true” in this scenario. Broker page invokes the “iAM Smart” Mobile App in the same device automatically. For details, please refer to Section 3.2.1.

3.4.2.2 Implementing (2) GET: Callback with authCode to Online Service Server



Pre-conditions

- “iAM Smart” user has authorised the authentication request of Online Service in “iAM Smart” Mobile App.
- “iAM Smart” user can also reject the authentication request in “iAM Smart” Mobile App.

Post-conditions

- Please refer to Section 3.4.1.2.

Error Conditions

- Please refer to Section 3.4.1.2.

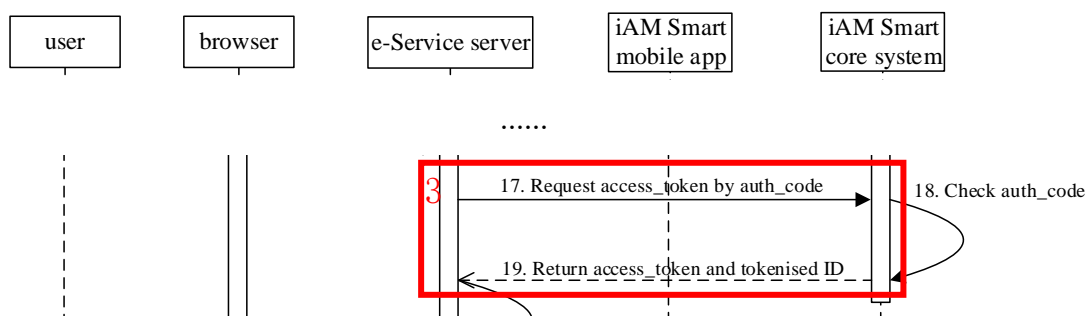
Callback Parameters

- Please refer to Section 3.4.1.2.

Notes

- Please refer to Section 3.4.1.2.

3.4.2.3 Implementing (3) POST: Request accessToken & Tokenised ID



Please refer to Section 3.4.1.3.

3.4.3 Scenario 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 in a separated mobile device.

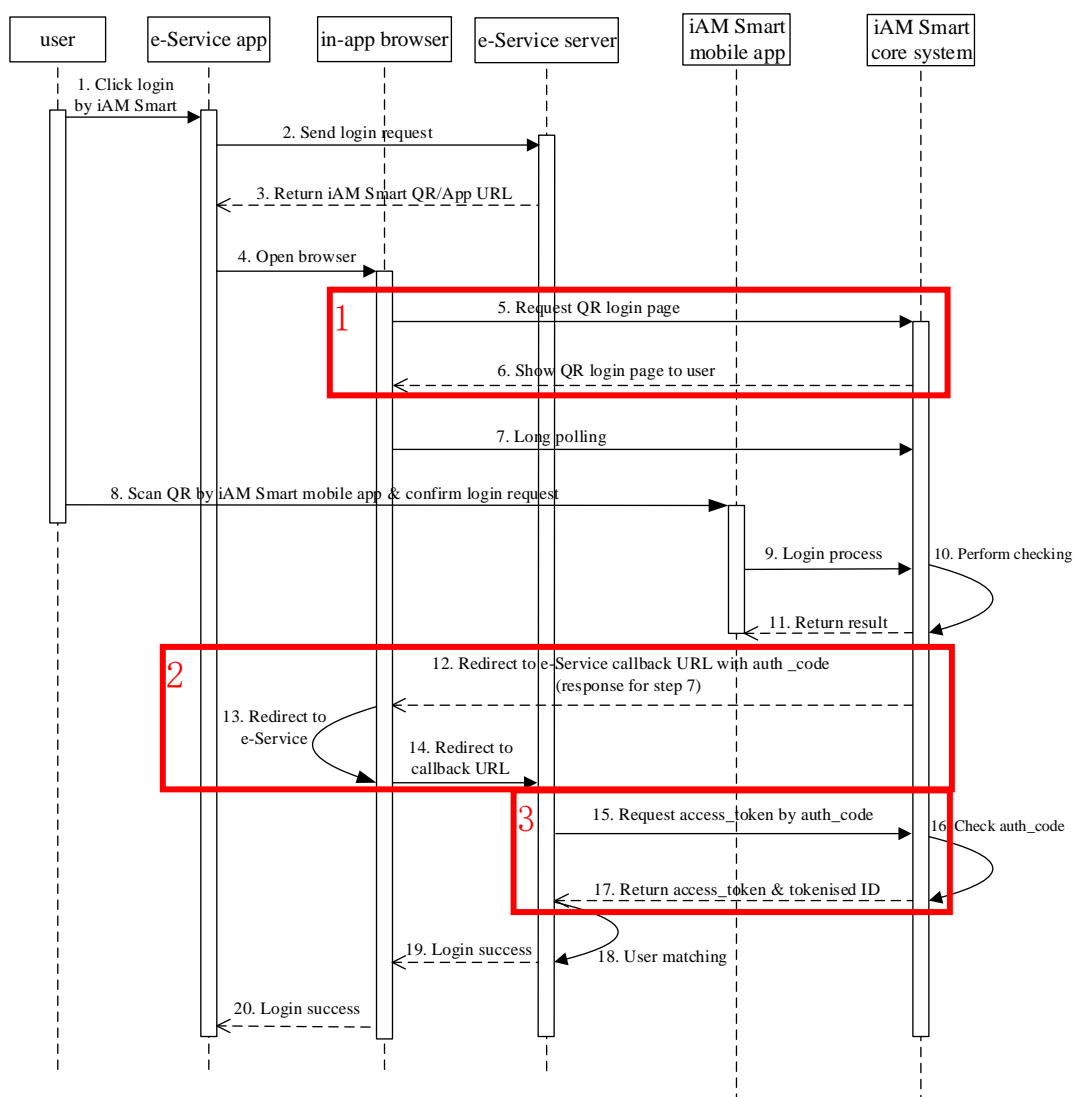


Figure-4 Authentication (Online Service App in Different Device)

- Step 1. User clicks “Login by iAM Smart” button in Online Service App;
- Step 2-3. Online Service App determines there is no “iAM Smart” Mobile App in the device using program code, requests Online Service Server to prepare the request parameters including “clientId”, “redirectURI”, “scope”, “source” (set as in-app browser’s user agent value), “brokerPage” (set as “false”), etc. and constructs the GET request to invoke the “iAM Smart” API by Online Service App;
- Step 4. Online Service App invokes in-app browser (Safari for iOS, Chrome for Android) to submit the GET request and keeps synchronise with Online Service server for the status of the login request;

Step 5-6. Browser opens the GET request to invoke the “iAM Smart” API and QR Code page will be shown;

“iAM Smart” API (GET: Request QR page)

Step 7. QR Code page of “iAM Smart” System polls “iAM Smart” System for the QR Code status;

Step 8-9. “iAM Smart” user logs in “iAM Smart” Mobile App to scan the QR Code and confirms Online Service's login request;

Step 10-11. “iAM Smart” System verifies the validity of QR code and other necessary information, and responses the login result to “iAM Smart” Mobile App (e.g., login success and inform “iAM Smart” user to proceed in Online Service, invalid / expired QR code, etc.);

Step 12-14. QR Code page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e., response for the polling in Step 7) which includes authCode or any error code (e.g., “iAM Smart” user rejects the login request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

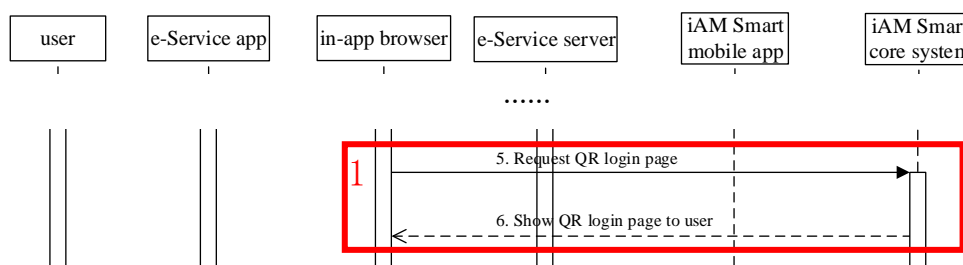
Step 15-17. When Online Service Server gets the authCode, it should invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

“iAM Smart” API (POST: Request accessToken & Tokenised ID)

Step 18. Online Service Server then performs user matching using the Tokenised ID with its user repository and determines the result of this login request;

Step 19-20. The Online Service App shows the login result (e.g., successful when Tokenised ID matched with a user account of Online Service).

3.4.3.1 Implementing (1) GET: Request QR page



Pre-conditions

- Online Service should determine all the authorisation scopes required for this authentication request.
- Online Service should determine the “iAM Smart” Mobile App is not in the same device of Online Service App using program code.
- Online Service should invoke in-app browser (Safari for iOS, Chrome for Android) to submit the GET request, and transfer the browser's user agent name to “iAM Smart”.
- Online Service should determine if it would generate the optional “state” request parameter to prevent CSRF attack. The “state” will be returned to Online Service for checking through the Online Service callback API for receiving the authorisation code from “iAM Smart” System in Step 14.

Post-conditions

- The browser will show the QR Code page of “iAM Smart” System with a “Return” button to allow user to return to Online Service App.
- Online Service App should keep synchronising with Online Service server for the status of the login request.

Error conditions

- This “iAM Smart” API does not return JSON response (i.e., no application error will be returned). For common errors, please refer to Section 3.2.3.

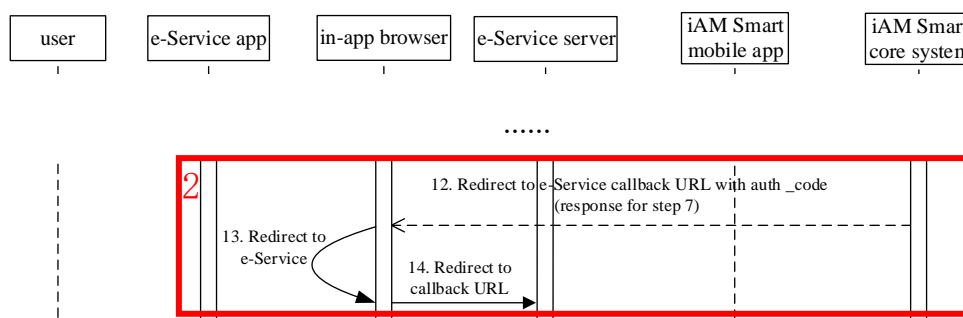
Request Parameters

- Please refer to “iAM Smart” API Specification.

Notes

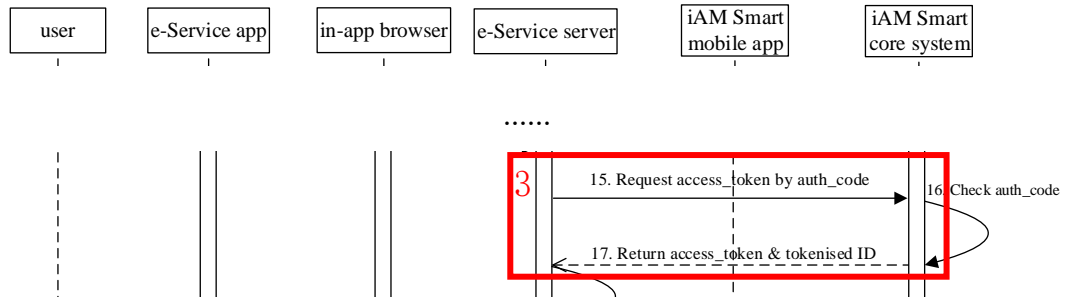
- Please refer to Section 3.4.1.1, except for the following parameter:
 - Request parameter “source”: Value should be the in-app browser's user agent value.

3.4.3.2 Implementing (2) GET: Callback with authCode to Online Service Server



Please refer to Section 3.4.1.2.

3.4.3.3 Implementing (3) POST: Request accessToken & Tokenised ID



Please refer to Section 3.4.1.3.

3.4.4 Scenario 4: Authentication (Online Service App in Same Device)

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.

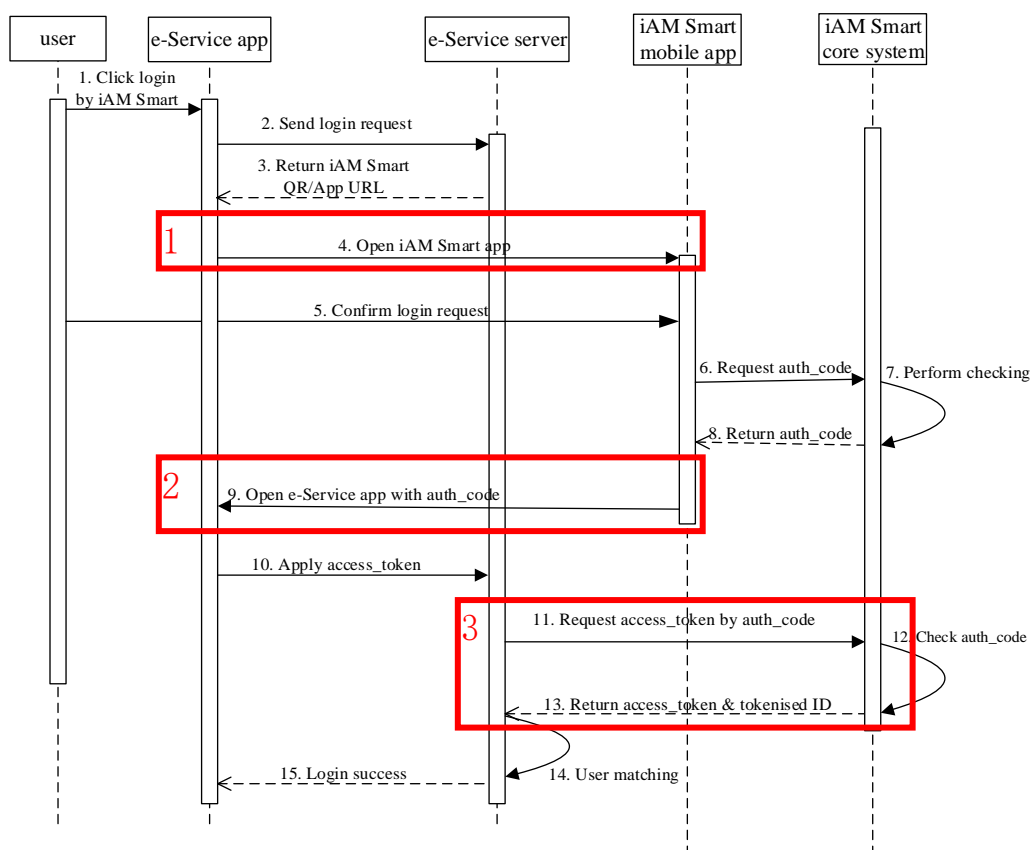
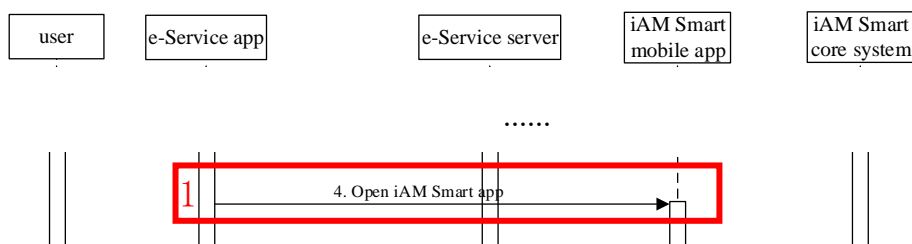


Figure-5 Authentication (Online Service App in Same Device)

- Step 1. User clicks “Login by iAM Smart” button on Online Service App;
- Step 2-3. Online Service App determines “iAM Smart” Mobile App is installed in the device using program code, requests Online Service Server to prepare the request parameters including “clientID”, “redirectURI” (set as Online Service App URL Scheme or Universal Link/App Link depending on “source” parameter), “scope”, “source” (set as “App_Scheme” or “App_Link”), etc. and constructs the URL Scheme to invoke “iAM Smart” Mobile App by Online Service App;
- Step 4. Online Service App invokes “iAM Smart” Mobile App using URL Scheme with request parameters (set <Context> as “auth” in URL Scheme);
- “iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Authentication)*

- Step 5-7. “iAM Smart” user logs in the “iAM Smart” Mobile App confirms the Online Service login request and “iAM Smart” Mobile App submits the login request to “iAM Smart” System;
- Step 8. “iAM Smart” System verifies the necessary information of the login request and returns login result to “iAM Smart” Mobile App;
- Step 9. “iAM Smart” Mobile App invokes Online Service App with required parameters using URL Scheme or Universal Link/App Link;
- Online Service Callback API (Callback with authCode to Online Service App)*
- Step 10. Online Service App sends authCode, etc. to Online Service Server;
- Step 11-13. When Online Service Server gets the authCode, it should invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;
- “iAM Smart” API (POST: Request accessToken & Tokenised ID)*
- Step 14. Online Service then performs user matching using the Tokenised ID with its user repository and determines the result of this login request;
- Step 15. Online Service App shows the login result (e.g., successful when Tokenised ID is matched with a user account of Online Service).

3.4.4.1 Implementing (1) URL Scheme: Open “iAM Smart” Mobile App for Authentication



Pre-conditions

- Online Service should determine all the authorisation scopes required for this authentication request.
- Online Service should determine the “iAM Smart” Mobile App is on the same device of Online Service App using program code.
- Online Service should determine if it would generate the optional “state” request parameter to prevent CSRF attack. The “state” will be returned to Online Service for checking through the Online Service callback API for receiving the authorisation code from “iAM Smart” System in Step 9.

Post-conditions

- “iAM Smart” Mobile App will be launched.
- Online Service App should keep synchronising with Online Service server for the status of the login request.

Error conditions

- Nil

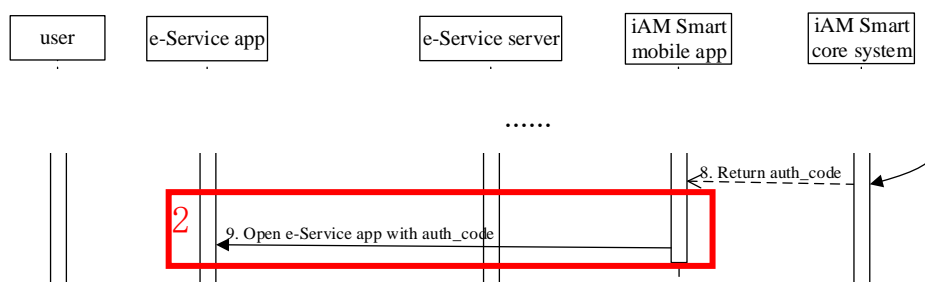
Request Parameters

- This “iAM Smart” API is a URL Scheme. Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.4.1.1, except for the following parameters:
 - Request parameter “source”: Value should be “App_Scheme” (for the Online Service App URL scheme), or “App_Link” (for the Universal Link/App Link).
 - Request parameter “redirectURI”: This is Online Service App URL scheme or Universal Link/App Link depending on the “source” parameter, which is used for receiving the authorisation code. The value should be URL encoded. For details, please refer to “iAM Smart” API Specification. The value must be the same as provided during Online Service registration.
- Set <Context> as “auth” in URL Scheme.

3.4.4.2 Implementing (2) Callback with authCode to Online Service App



Pre-conditions

- Please refer to Section 3.4.2.2.

Post-conditions

- Please refer to Section 3.4.2.2.

Error conditions

- This Online Service callback API is a URL Scheme, or Universal Link/App Link. If parameter “error_code” is returned, it means the authentication request is failed.

Error Code	Error Description	Suggested Action
error_code - D40000	User cancelled authentication request	Inform user the authentication request is cancelled
error_code - D40001	User rejected authentication request	Inform user the authentication request is rejected
error_code - D40002	Failed to authenticate	Inform user the authentication request is failed and retry later
error_code - D40003	Authentication request timeout	Inform user the authentication request is timeout, and provide way for user to do re-authentication

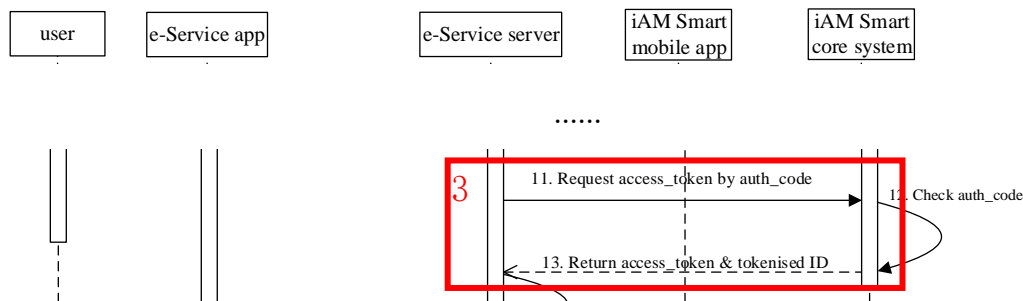
Callback Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.4.2.2.

3.4.4.3 Implementing (3) POST: Request accessToken & Tokenised ID



Please refer to Section 3.4.1.3.

3.4.5 Workflows for verifying CCIC user

3.4.5.1 Scenario : Verify whether the “iAM Smart” user is a CCIC holder

The sequence diagram below shows how Online Services invoke “verifyIsCcic” API to verify whether the “iAM Smart” user is a CCIC holder after performing authentication process.

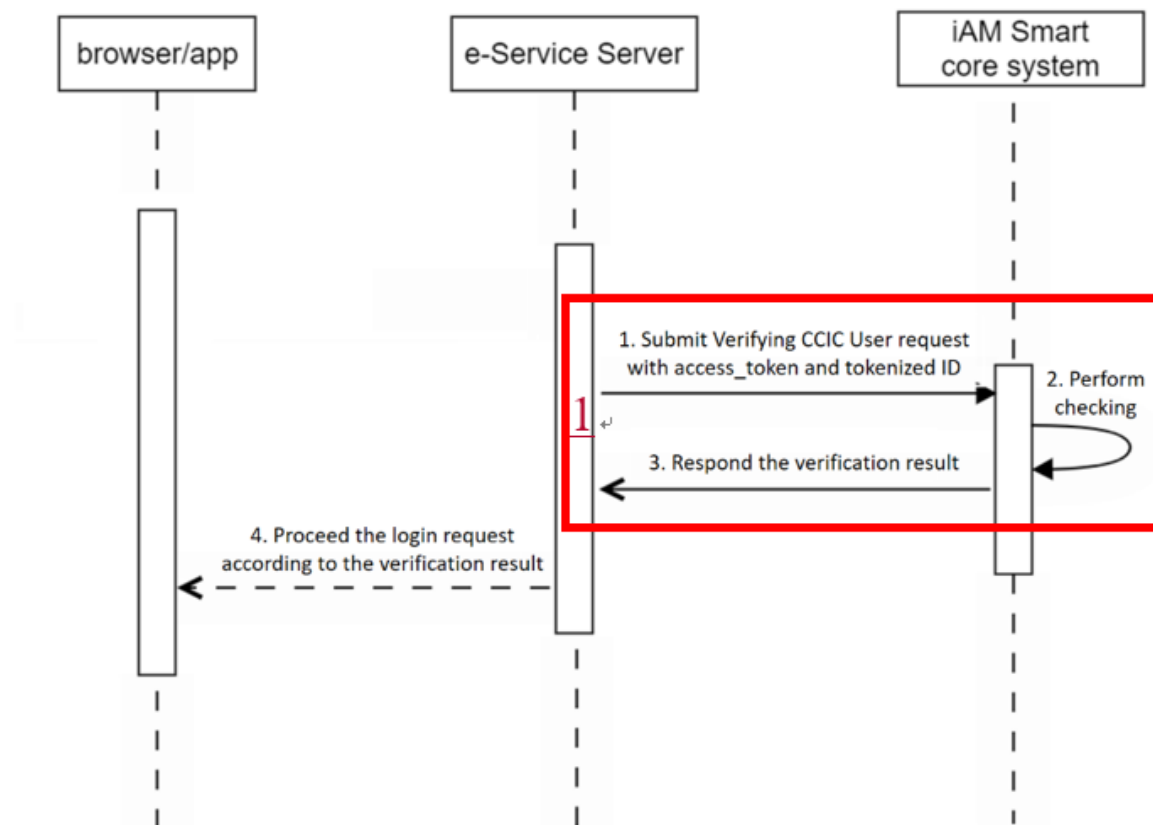


Figure-6 Verify whether the “iAM Smart” user is a CCIC holder

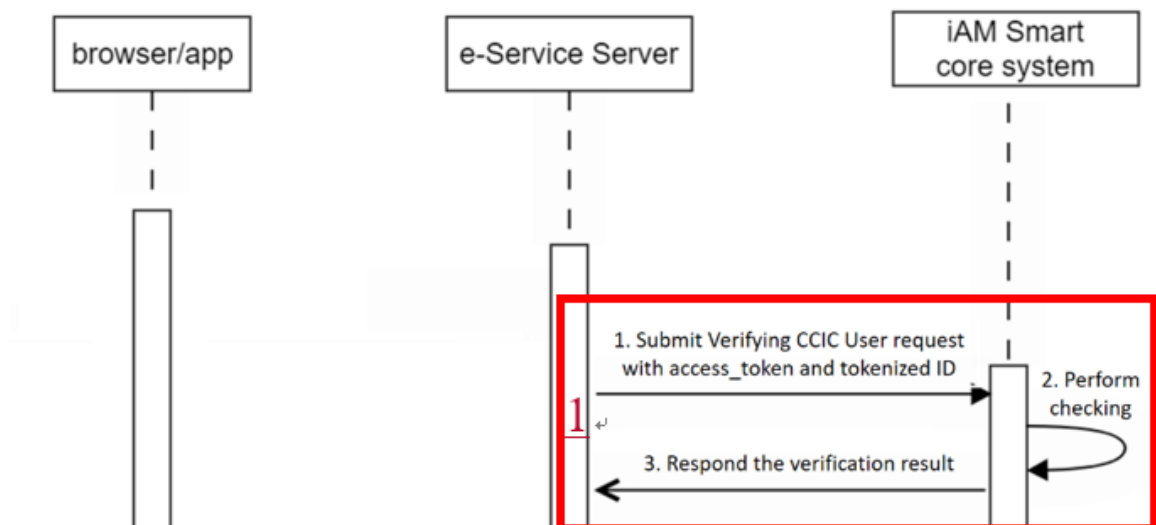
Step 1. Online Service Server initiates a request to invoke the “iAM Smart” *verifyIsCcic* API by sending user’s accessToken and openID to “iAM Smart” System. API data encryption is required;

“iAM Smart” API (POST: *verifyIsCcic*)

Step 2-3. “iAM Smart” core system receives the request and verifies the validity of the accessToken and openID, and then respond the result to Online Service Server. API data encryption is required;

Step 4. The Online Service Server will proceed subsequent login process or display error message according to the verification result.

3.4.5.1.1 Implementing (1) POST: VerifyIsCcic



Pre-conditions

- Online Service must possess a valid accessToken and OpenID of the “iAM Smart” user.
- Online Service must apply the authorisation scope “Authentication” and API Access Control “verifyIsCcic” in advance.
- API data encryption is required.

Post-conditions

- Online Service should check the response parameters “verifyResult” and determine the next action.
- API data decryption is required.

Error conditions

- Please refer to Section 3.2.3 for common errors.

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.

3.5 WORKFLOWS FOR FORM FILLING AFTER AUTHENTICATION API

3.5.1 Scenario 1: Form Filling (e-Service Website/App in Different Device)

The sequence diagram below shows how an authenticated user authorises an e-Service to use his/her e-ME profile and/or iAM Smart profile for form filling when e-Service and the iAM Smart Mobile App are running in different devices.

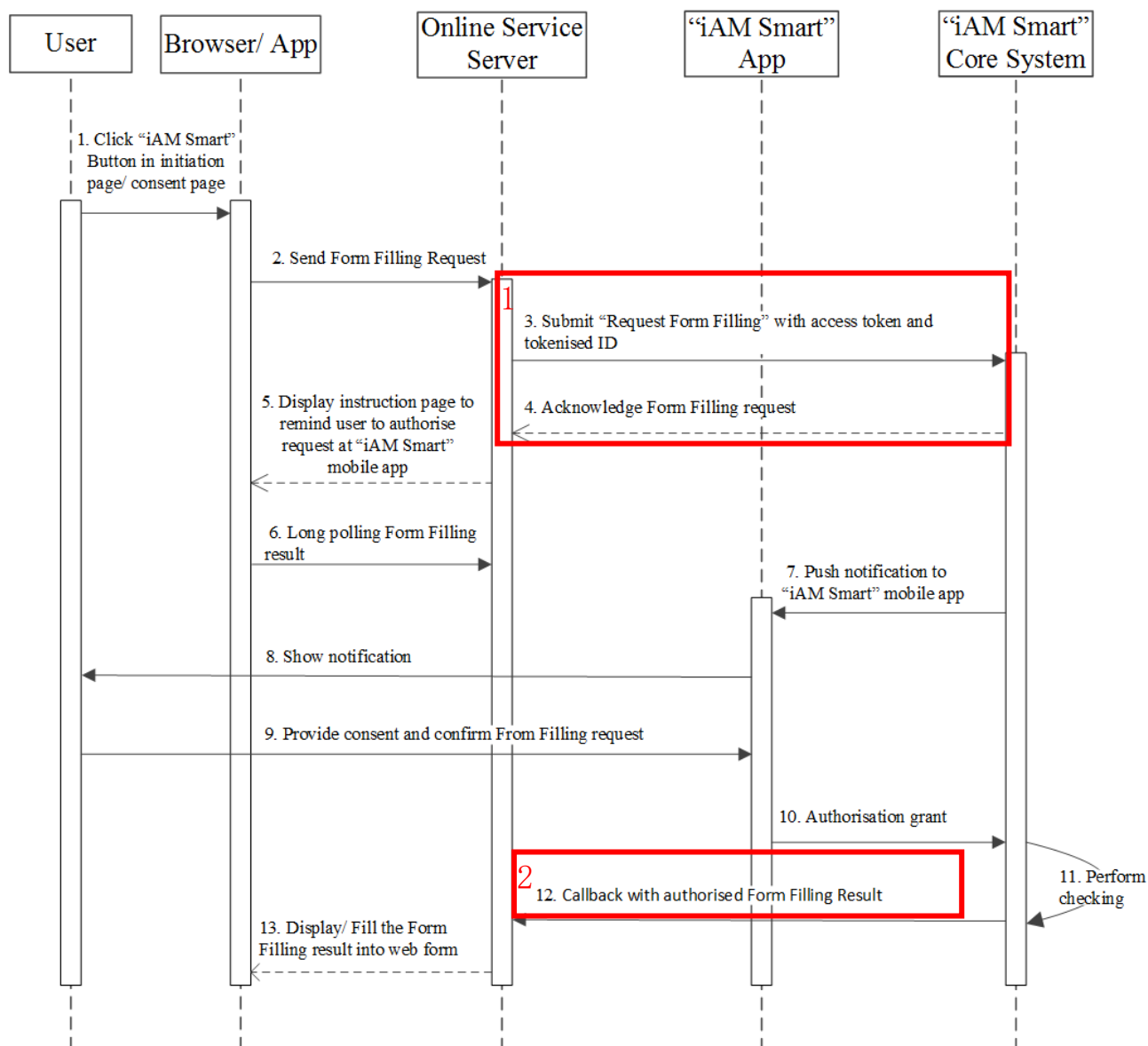


Figure-7 Form Filling (e-Service Website/App in Different Device)

- Step 1. User reads the consent page (if iAM Smart profile fields requested) and clicks the button in e-Service Website/App;
- Step 2. e-Service Website/App initiates form filling request to e-Service Server with browser's user agent name (for e-Service Website) or "App_Scheme"/"App_Link" (for e-Service App) in this scenario (i.e. for use as value for request parameter "source");

Step 3. e-Service Server initiates a Form Filling request to invoke the iAM Smart API with the “businessID” of this request, accessToken, Tokenised ID, required e-ME profile fields and/or iAM Smart profile fields, etc. The request parameter “source” will be the browser's user agent value or “App_Scheme”/“App_Link” in this scenario. API encryption is required;

iAM Smart API (POST: Request Form Filling)

Step 4. iAM Smart System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the Form Filling request to e-Service Server by returning POST response with parameter “authByQR” (set to “true”) in this scenario;

Step 5-6. e-Service Website/App shows instruction to inform iAM Smart user to process the Form Filling authorisation request in iAM Smart Mobile App. The e-Service Website/App should keep synchronising with the e-Service server for the request result (e.g. polling);

Step 7. iAM Smart System pushes a notification message to the iAM Smart Mobile App based on the Tokenised ID;

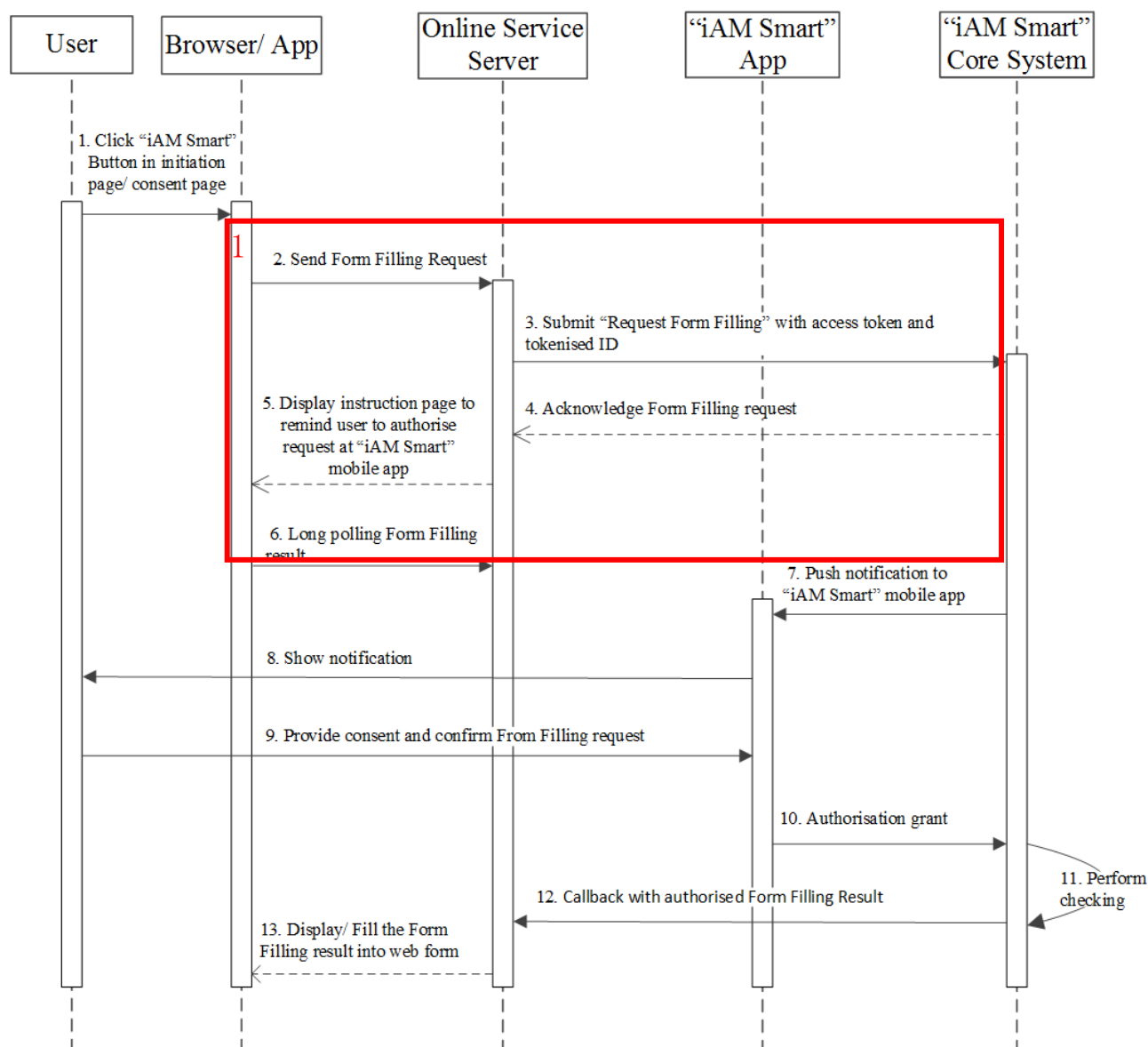
Step 8-10. iAM Smart user logs in iAM Smart Mobile App, reviews the Form Filling authorisation request (e.g. continue or reject the request) and authorises those selected e-ME profile fields and/or iAM Smart profile fields to e-Services (e.g. authorise HKIC number and english name in iAM Smart profile and residential address in e-ME profile(unchecked other e-ME profile fields));

Step 11-12. iAM Smart System invokes e-Service callback API to return the result with the “businessID” of the form filling request. API decryption is required;

e-Service Callback API (POST: Callback to Receive Form Filling Information)

Step 13. e-Service Server processes and matches the result using the “businessID” and shows the result in the corresponding e-Service Website/App.

3.5.1.1 Implementing (1) POST: Request Form Filling



Pre-conditions

- e-Service must possess a valid accessToken of the iAM Smart user with authorisation scope "Form Filling authorisation".
- e-Service Website/App and iAM Smart Mobile App are in different devices in this scenario.
- e-Service generates a "businessID" for this Form Filling request and uses this identifier to match the callback result returned from iAM Smart System.
- API data encryption is required.

Post-conditions

- e-Service server should check the response parameter "authByQR" and "ticketID" and determine the next action. For details, please refer to Section 3.2.1. "ticketID" will not be provided by iAM Smart System in this scenario.

- e-Service Website/App should show instructions to inform iAM Smart user to process the Form Filling request in iAM Smart Mobile App when “authByQR” is “true” is returned.
- e-Service Website/App should keep synchronising with e-Service server for the result returned from iAM Smart System.
- API data decryption is required.

Error conditions

- For common errors, please refer to Section 3.2.3. Other error of this API includes:

Error Code	Error Description	Suggested Action
code - D60002	Failed to request Form Filling	Inform user the Form Filling request is failed and retry later

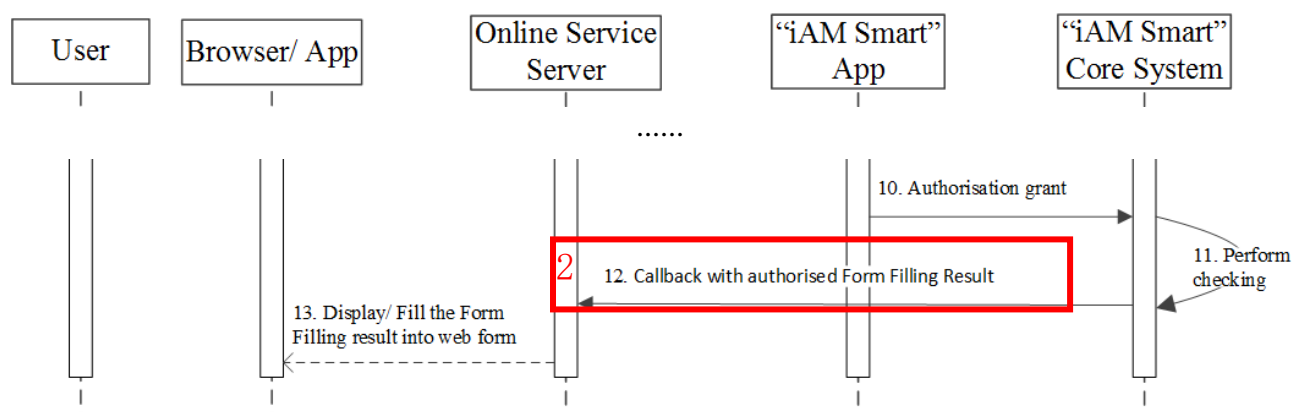
Request and Response Parameters

- Please refer to Section 6.3.7 of iAM Smart API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Request parameter “source”: Value should be matched with e-Service client terminal (e.g. “App_Scheme”/“App_Link” or browser’s user agent value).
- Request Parameter “redirectURI”: Value should equal to URI of “Callback to Receive Form Filling Information” e-Service callback API. It must be in the same value as provided during e-Service registration.
- Request parameters “formName”, “formNum”, “formDesc”: They are the information of the e-Service's form and will be shown in iAM Smart Mobile App for iAM Smart user's reference. e-Service should determine the language of these items.
- Request parameter “eMEFields”: It specifies the e-ME profile fields required by e-Service. The item name must match with the specification of this iAM Smart API. Otherwise, error code for the invalid parameter will be returned.
- Request parameter “profileFields”: It specifies the iAM Smart profile fields required by e-Service. The item name must match with the specification of this iAM Smart API. Otherwise, error code for the invalid parameter will be returned.
- Response parameter “authByQR”: The value returned by iAM Smart System will be “true” in this scenario.
- iAM Smart System will not return the response parameter “ticketID” in this scenario.

3.5.1.2 Implementing (2) POST: Callback to Receive Form Filling Information



Pre-conditions

- iAM Smart user can accept the request and authorise data fields to e-Service.
- iAM Smart user can also reject the form filling request.

Post-conditions

- API data decryption is required.
- e-Service Server should match the callback result with corresponding e-Service Website/App using the “businessID”.
- e-Service Server should check the requested data fields has authorised for the Anonymous Form Filling.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D60000	User cancelled Form Filling request	Inform user the Form Filling request is cancelled
code - D60001	User rejected Form Filling request	Inform user the Form Filling request is rejected
code - D60002	Failed to request Form Filling	Inform user the Form Filling request is failed and retry later
code - D60003	Form Filling request timeout	Inform user the Form Filling request is timeout, and provide way for user to retry

Callback Parameters

- Please refer to Section 6.4.4 of iAM Smart API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.

- e-ME profile fields that not authorised by iAM Smart user will not return in result (i.e. JSON name/value of that item will not exist in result).

3.5.2 Scenario 2: Form Filling (e-Service Website in Same Device)

The sequence diagram below shows how an authenticated user authorises an e-Service to use his/her e-ME profile and/or iAM Smart profile for form filling when e-Service website and the iAM Smart Mobile App are running in the same device.

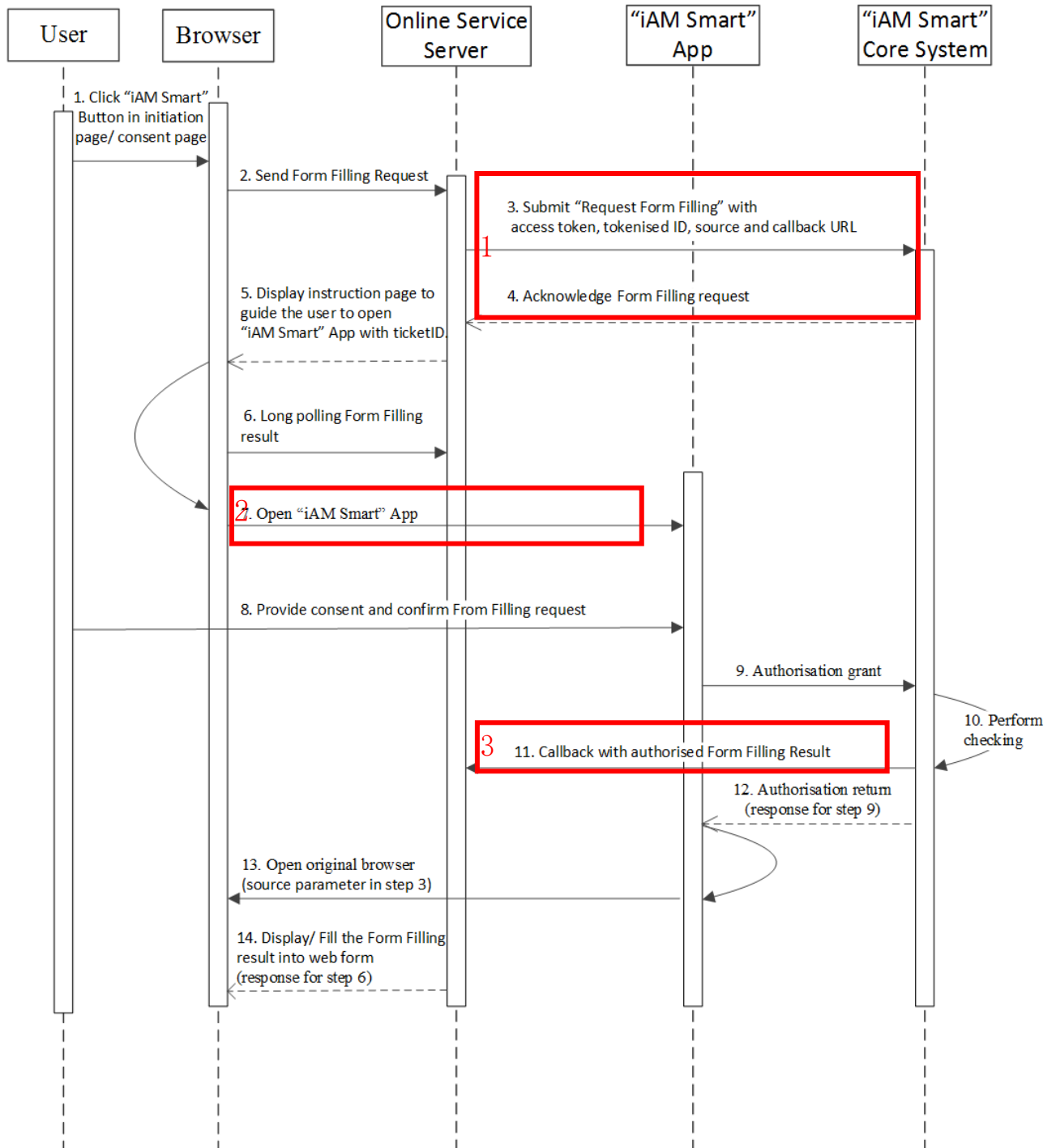
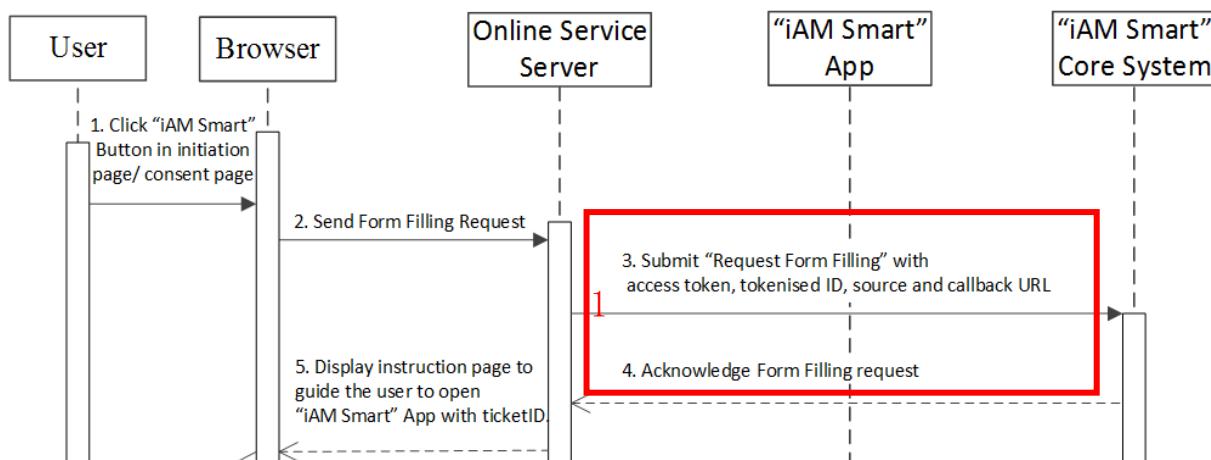


Figure-8 Form Filling (e-Service Website in Same Device)

- Step 1. User reads the consent page (if iAM Smart profile fields requested) and clicks the button in e-Service Website;
- Step 2. e-Service Website initiates form filling request to e-Service Server with browser's user agent name (i.e. for use as value for request parameter “source”);
- Step 3. e-Service Server initiates Form Filling request to invoke the iAM Smart API with the “businessID” of this request, accessToken, Tokenised ID, required e-ME profile fields and/or iAM Smart profile fields, etc. The request parameter “source” will be the browser's user agent value. API data encryption is required;
iAM Smart API (POST: Request Form Filling)
- Step 4. iAM Smart System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the Form Filling request to e-Service Server by returning POST response with parameter “authByQR” (set to “false”) and “ticketID” in this scenario;
- Step 5. After receiving the response, e-Service Server prepares and sends necessary parameters such as “authByQR”, “ticketID”, etc. to e-Service Website;
- Step 6-7. e-Service Website invokes iAM Smart Mobile App using URL Scheme with “ticketID” (set <Context> as “form-filling” in URL Scheme). Other parameters of this API are not used in this scenario. The e-Service Website should keep synchronising with the e-Service Server for the request result (e.g. polling). API data encryption is required;
iAM Smart API (URL Scheme: Open iAM Smart Mobile App for Getting Context)
- Step 8-10. iAM Smart user logs in iAM Smart Mobile App, reviews the Form Filling authorisation request (e.g. continue or reject the request) and authorises those selected e-ME profile fields and/or iAM Smart profile fields to e-Services (e.g. authorise HKIC number and english name in iAM Smart profile and residential address in e-ME profile(unchecked other e-ME profile fields));
- Step 11. iAM Smart System invokes e-Service callback API to return the result with the “businessID” of the Form Filling request. API decryption is required;
e-Service callback API (POST: Callback to Receive Form Filling Information)
- Step 12-13. iAM Smart System instructs iAM Smart Mobile App to invoke the original e-Service browser (i.e. using the browser's user agent value submitted in form filling request in Step 3);
- Step 14. e-Service server processes and matches the result using the “businessID” and shows the result in the corresponding e-Service Website.

3.5.2.1 Implementing (1) POST: Request Form Filling



Pre-conditions

- Please refer to Section 3.5.1.1 except:
 - e-Service Website and iAM Smart Mobile App are in the same device in this scenario.

Post-conditions

- Please refer to Section 3.5.1.1 except:
 - Response "authByQR" is "false", "ticketID" will be provided by iAM Smart System in this scenario.

Error conditions

- Please refer to Section 3.5.1.1.

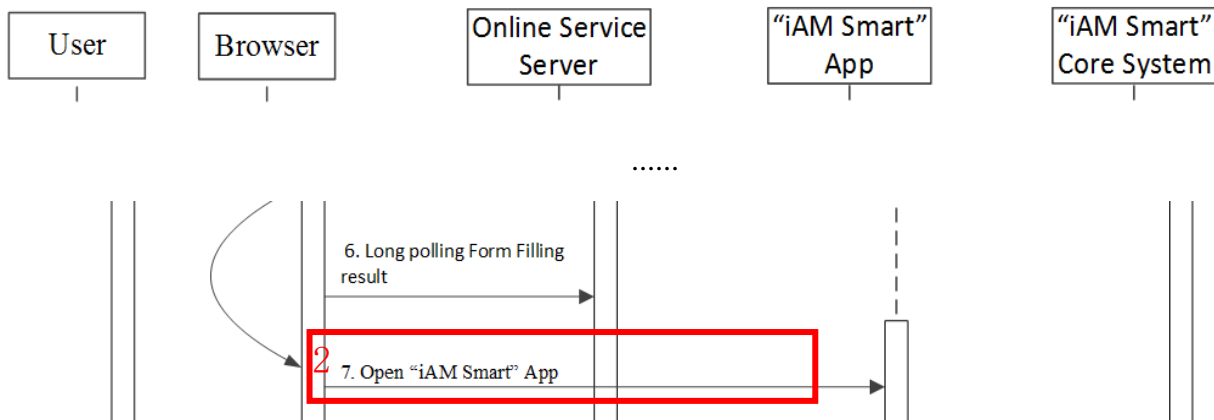
Request and Response Parameters

- Please refer to Section 6.3.7 of iAM Smart API Specification.

Notes

- Please refer to Section 3.5.1.1, except for the following parameters:
 - Request parameter "source": Value should be the browser's user agent value.
 - Response parameter "authByQR": The value is "false" in this scenario.
 - Response parameter "ticketID": It will be provided by iAM Smart System in this scenario. It is a 36-byte (or less) UUID number (ASCII character set).

3.5.2.2 Implementing (2) URL Scheme: Open iAM Smart Mobile App for Getting Context



Pre-conditions

- e-Service Website and iAM Smart Mobile App are in the same user device.
- e-Service has the "ticketID" provided by iAM Smart System for the form filling request;
- Other parameters of this API are not used in this scenario.

Post-conditions

- iAM Smart Mobile App will be launched.
- e-Service Website should keep synchronising with e-Service server for the callback response of the form filling request from iAM Smart System.

Error conditions

- Nil

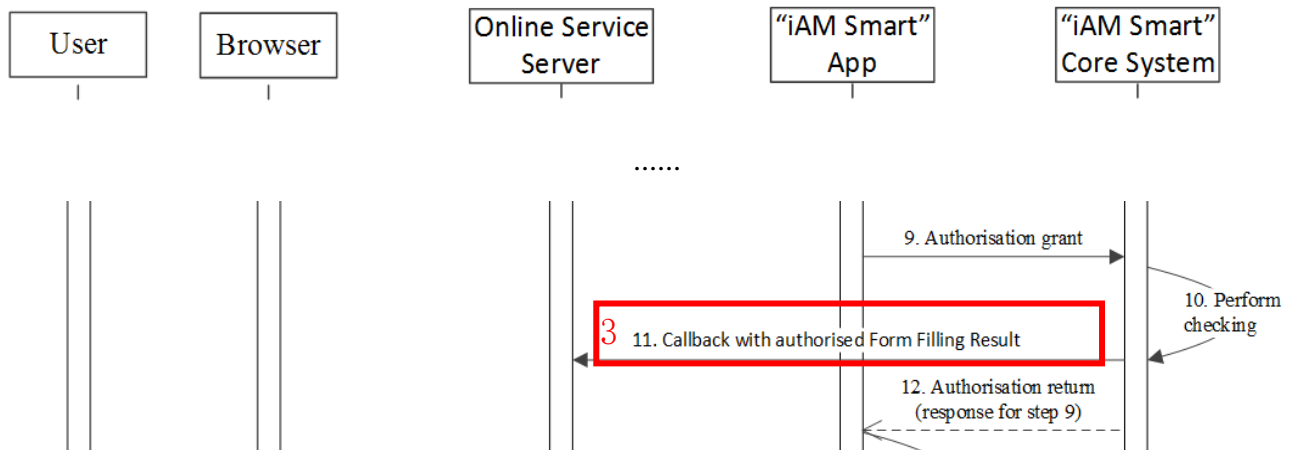
Request Parameters

- Please refer to Section 6.3.5 of iAM Smart API Specification.

Notes

- Set <Context> as "form-filling" in URL Scheme.

3.5.2.3 Implementing (3) POST: Callback to Receive Form Filling Information



Please refer to Section 3.5.1.2.

3.5.3 Scenario 3: Form Filling (e-Service App in Same Device)

The sequence diagram below shows how an authenticated user authorises an e-Service to use his/her e-ME profile and/or iAM Smart profile for form filling when e-Service App and the iAM Smart Mobile App are running in the same device.

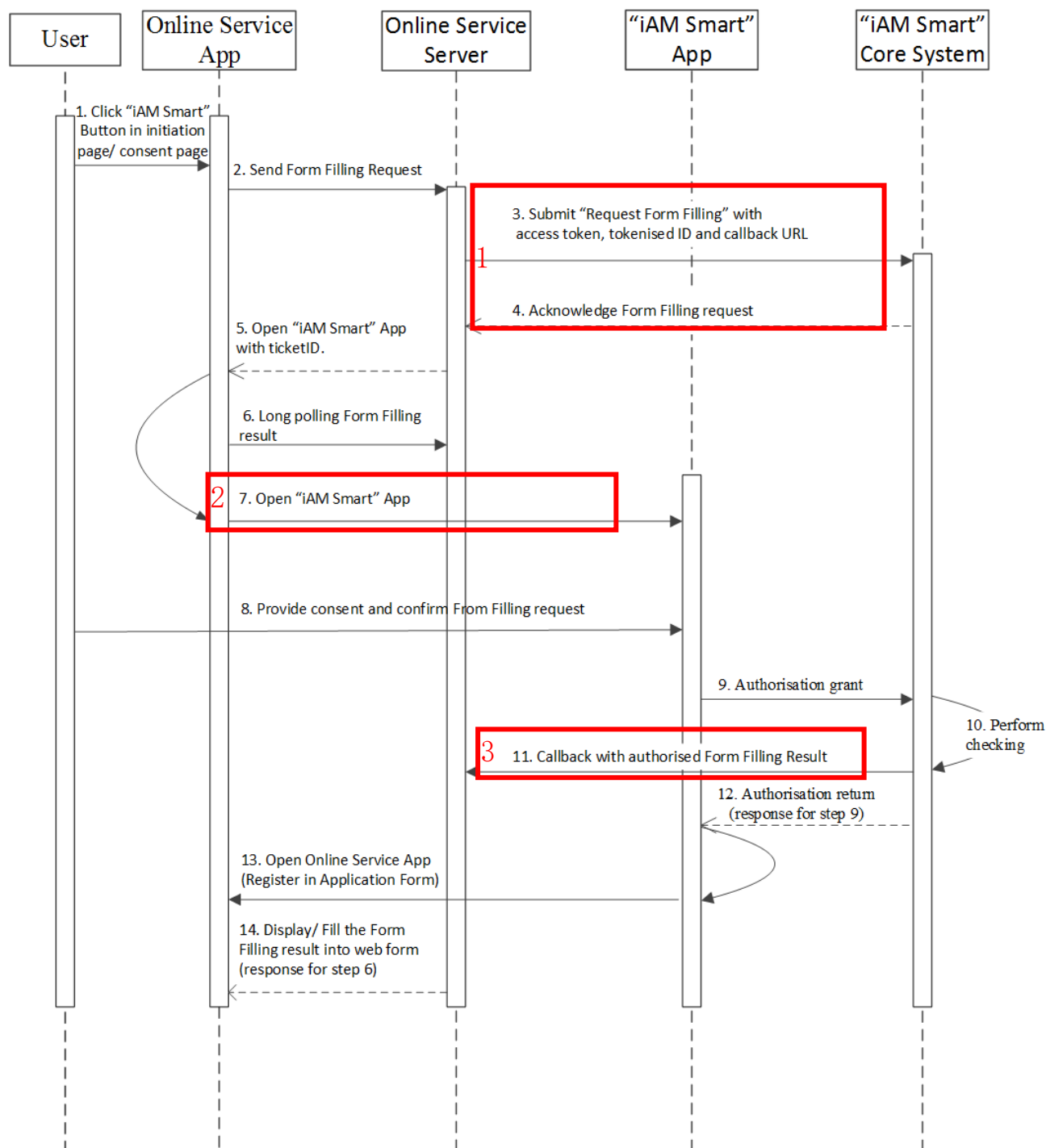
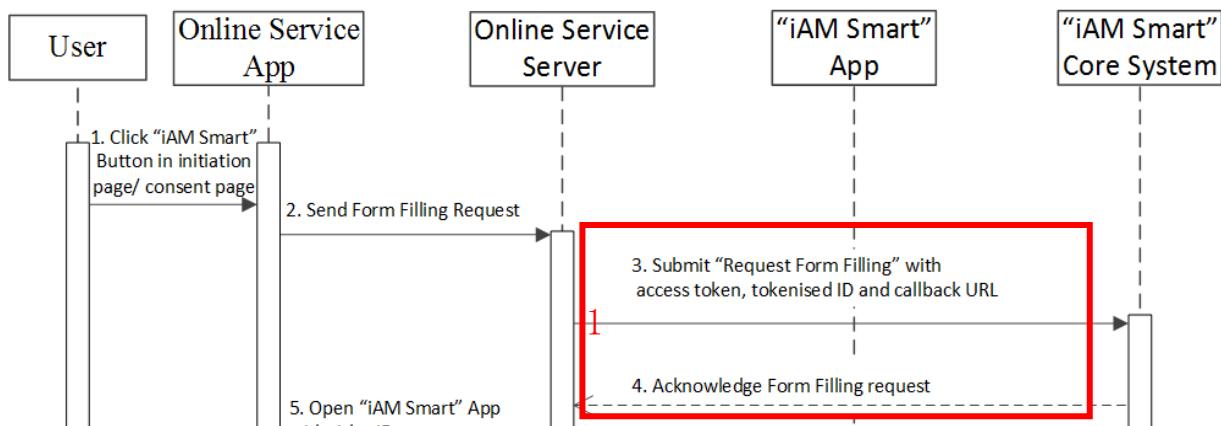


Figure-9 Form Filling (e-Service App in Same Device)

Step 1. User reads the consent page (if iAM Smart profile fields requested) and clicks the button in e-Service App;

- Step 2. e-Service App initiates form filling request to e-Service Server with “App_Scheme” or “App_Link” (for use as value for request parameter “source”);
- Step 3. e-Service Server initiates form filling request to invoke the iAM Smart API with the “businessID” of this request, accessToken, Tokenised ID, required e-ME items, etc. The request parameter “source” will be “App_Scheme” or “App_Link”. API data encryption is required;
- iAM Smart API (POST: Request Form Filling)*
- Step 4. iAM Smart System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the form filling request to e-Service Server by returning POST response with parameter “authByQR” (set to “false”) and “ticketID” in this scenario;
- Step 5. After receiving the response, e-Service Server prepares necessary parameters such as “authByQR”, “ticketID”, etc. to e-Service App;
- Step 6-7. e-Service App determines “authByQR” to be false, or uses program code to check if iAM Smart Mobile App exists in the same device, then invokes iAM Smart Mobile App using URL Scheme with “ticketID” (set <Context> as “form-filling” in URL Scheme). Other parameters of this API are not used in this scenario. The e-Service App should keep synchronising with the e-Service Server for the request result (e.g. polling);
- iAM Smart API (URL Scheme: Open iAM Smart Mobile App for Getting Context)*
- Step 8-9. iAM Smart user logs in iAM Smart Mobile App, reviews the Form Filling authorisation request (e.g. continue or reject the request) and authorises those selected e-ME profile fields and/or iAM Smart profile fields to e-Services (e.g. authorise HKIC number and english name in iAM Smart profile and residential address in e-ME profile (unchecked other e-ME profile fields));
- Step 10-11. iAM Smart System invokes e-Service callback API to return the result with the “businessID” of the form filling request. API decryption is required;
- e-Service Callback API (POST: Callback to Receive Form Filling Information)*
- Step 12-14. iAM Smart System instructs iAM Smart Mobile App to invoke the e-Service App using URL Scheme, or Universal Link/App Link (iAM Smart System queries the information registered at e-Service registration depending on the “source” submitted in Step 3).

3.5.3.1 Implementing (1) POST: Request Form Filling



Pre-conditions

- Please refer to Section 3.5.2.1 except:
 - e-Service App and iAM Smart Mobile App are in the same device in this scenario.

Post-conditions

- Please refer to Section 3.5.2.1.

Error conditions

- Please refer to Section 3.5.2.1.

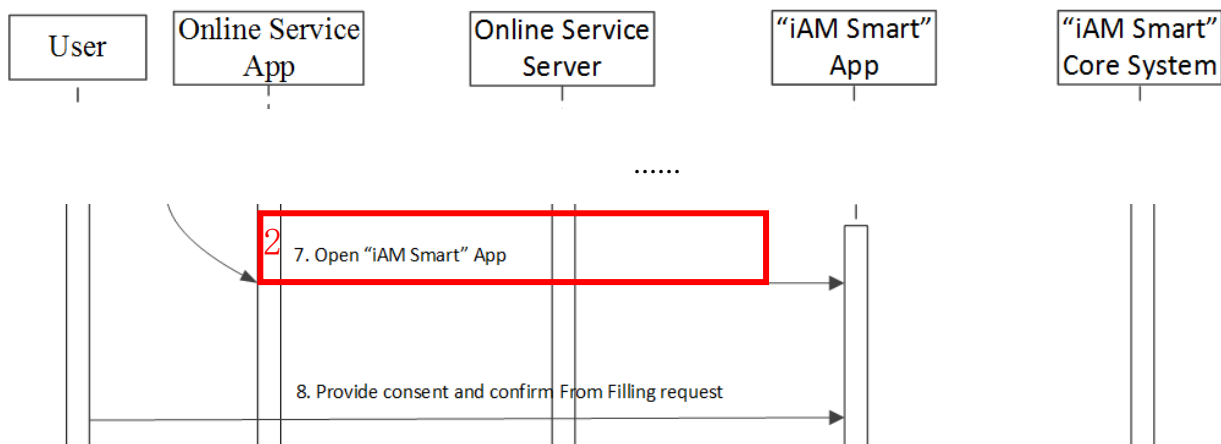
Request and Response Parameters

- Please refer to Section 6.3.7 of iAM Smart API Specification.

Notes

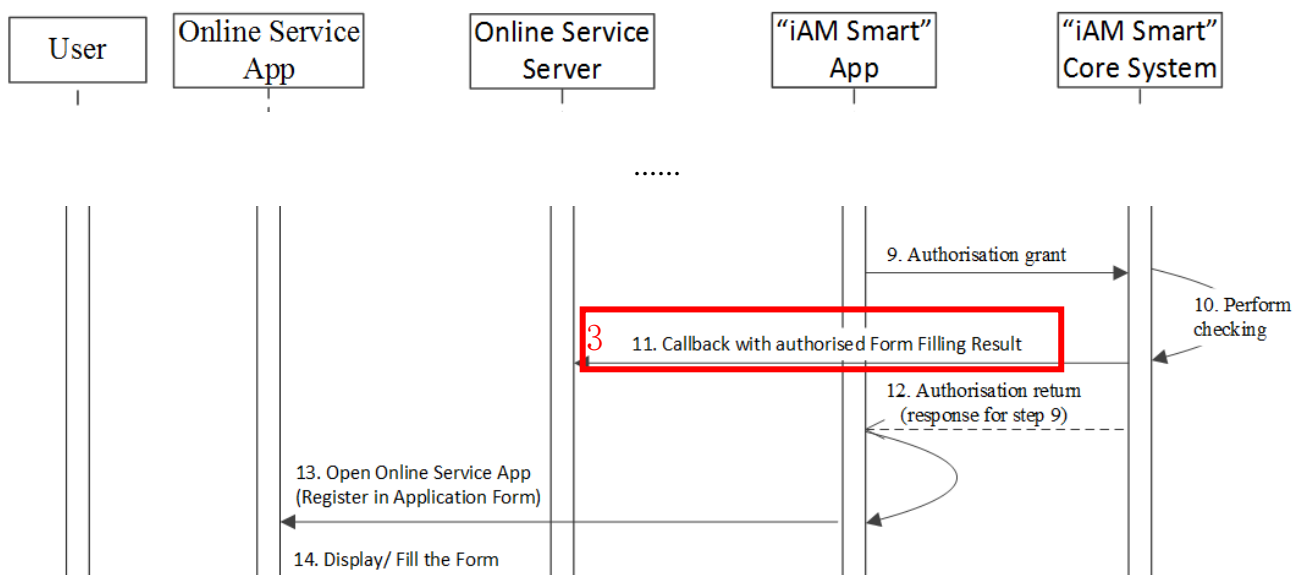
- Please refer to Section 3.5.2.1, except for the following parameter:
 - Request parameter "source": Value should be "App_Scheme" (for the e-Service App URL scheme), or "App_Link" (for the Universal Link/App Link).

3.5.3.2 Implementing (2) URL Scheme: Open iAM Smart Mobile App for Getting Context



Please refer to Section 3.5.2.2.

3.5.3.3 Implementing (3) POST: Callback to Receive Form Filling Information



Please refer to Section 3.5.1.2.

3.6 WORKFLOWS FOR FORM FILLING WITHOUT SERVICE LOGIN (AKA ANONYMOUS FORM FILLING)

3.6.1 Scenario 1: Anonymous Form Filling (Online Service Website in Different Device)

The sequence diagram below shows how an anonymous user authorises an Online Service to use his/her “eMEFields” for form filling when Online Service website and the “iAM Smart” Mobile App are running in different devices.

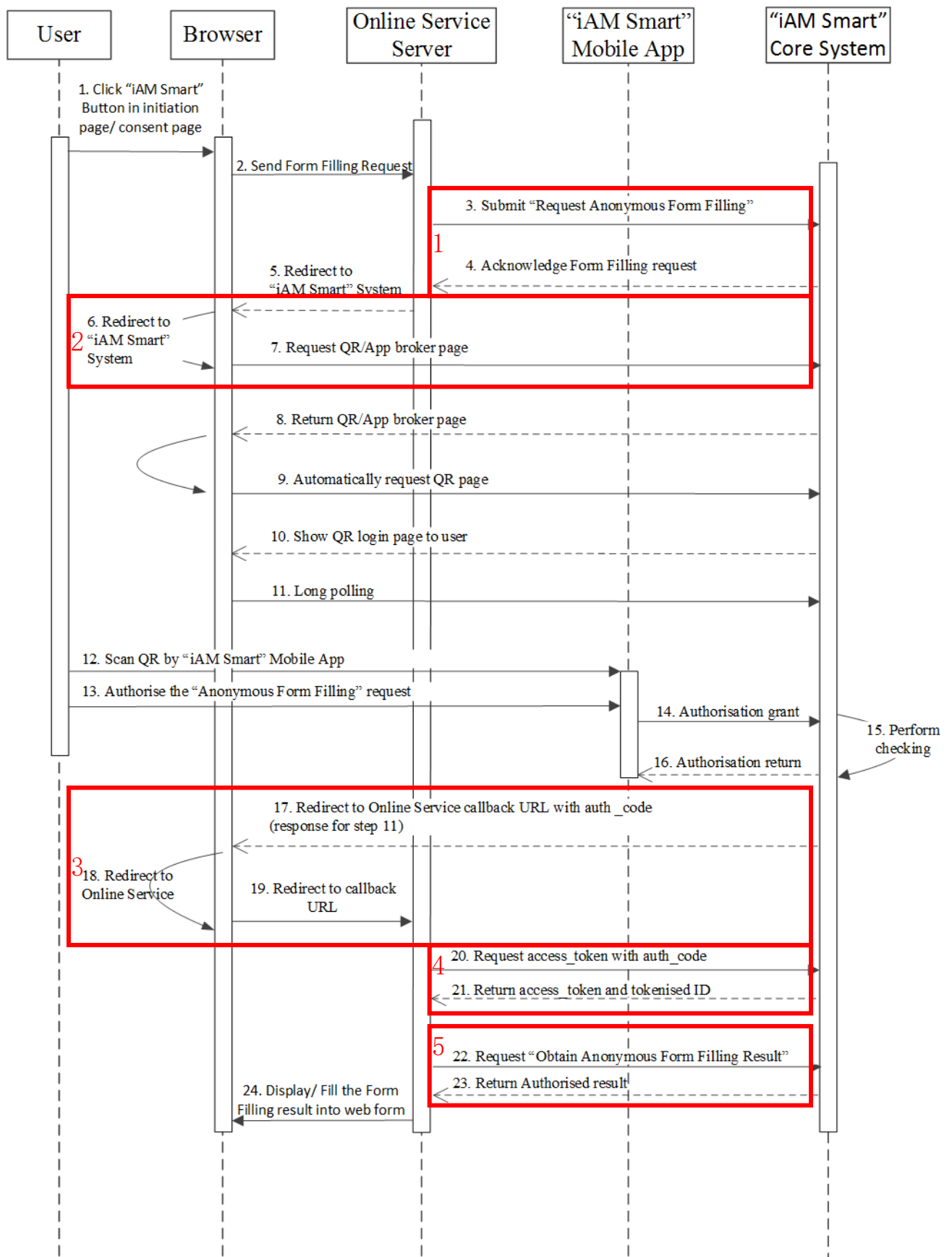


Figure-10 Anonymous Form Filling (Online Service Website in Different Device)

- Step 1. User reads the consent page (if “iAM Smart” profile fields requested) and clicks the button in Online Service Website;
- Step 2. Online Service Website initiates anonymous form filling request to Online Service Server with browser's user agent name (use as value for request parameter “source”);
- Step 3. Online Service Server initiates an anonymous form filling request to invoke the “iAM Smart” API with the “businessID” of this request, required “eMEFields”, etc. API data encryption is required;

“iAM Smart” API (POST: Request Anonymous Form Filling)

- Step 4. “iAM Smart” System verifies and confirms receipt of the anonymous Form Filling request to Online Service Server by returning POST response with parameter “ticketID”;
- Step 5-7. Online Service Server prepares the request parameters such as “clientID”, “redirectURI”, “source” (set as browser’s user agent value), “scope”, “ticketID”, etc. and constructs the GET request to invoke the “iAM Smart” API using browser redirection;

“iAM Smart” API (GET: Request QR page)

- Step 8-10. “iAM Smart” System returns the broker page (if Online Service has set “brokerPage” to “true”) and after the broker page fails to find the “iAM Smart” Mobile App, it will request QR page to be displayed in browser. If Online Service does not request for broker page (i.e., no broker page will be returned), the browser will directly display QR Code page;
- Step 11. QR Code page of “iAM Smart” System polls “iAM Smart” System for the QR Code status;
- Step 12-14. “iAM Smart” user uses “iAM Smart” Mobile App to scan the QR Code, reviews the form filling authorisation request (e.g., continue or reject the request) and authorises those selected “eMEFields” to Online Services (e.g., authorise only his HKIC number and English name in account information and residential address in “e-ME profile” (unchecked other “eMEFields”));

Step 15-16. “iAM Smart” System verifies the validity of QR code and other necessary information, and return the authorisation result to “iAM Smart” Mobile App;

Step 17-19. QR Code page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e., response for the polling in Step 11) which includes authCode and businessID or any error code (e.g., “iAM Smart” user rejects the request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

Step 20-21. When Online Service Server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

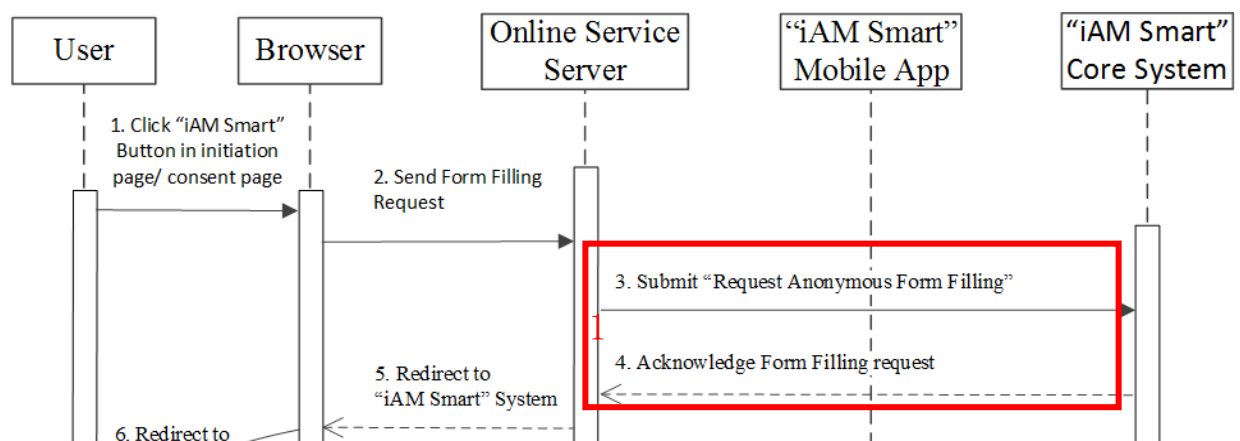
“iAM Smart” API (POST: Request accessToken & Tokenised ID)

Step 22-23. Online Service Server invokes the “iAM Smart” API with the accessToken and openID to obtain the result of anonymous form filling request. API data encryption is required;

“iAM Smart” API (POST: Obtain Anonymous Form Filling Result)

Step 24. Online Service Server processes and shows the result in the corresponding Online Service Website.

3.6.1.1 Implementing (1) POST: Request Anonymous Form Filling



Pre-conditions

- Online Service Website and “iAM Smart” Mobile App are in different devices in this scenario.
- Online Service generates a “businessID” for this request and uses this identifier to match the authCode returned from “iAM Smart” System.
- API data encryption is required.

Post-conditions

- API data decryption is required.

Error conditions

- Please refer to Section 錯誤! 找不到參照來源。 .

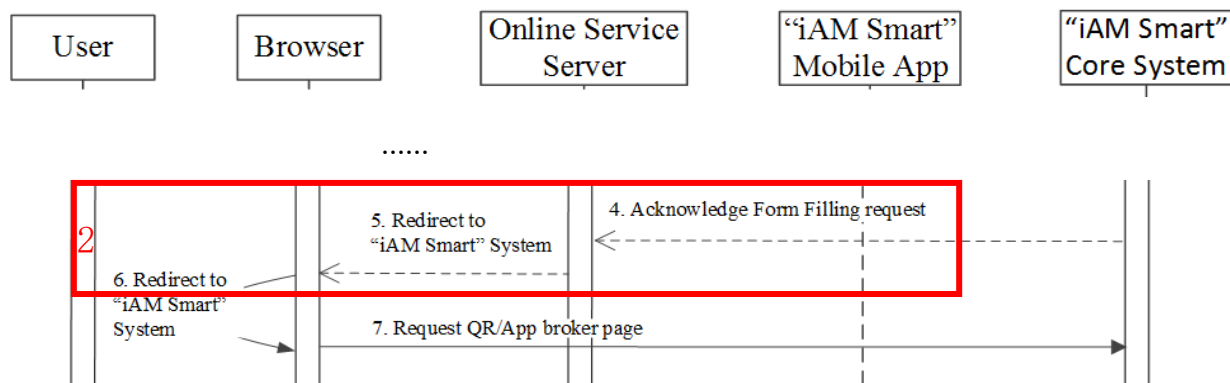
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 錯誤! 找不到參照來源。 , except for the following parameters:
 - No request parameter “accessToken”, “openID”, “source”, “redirectURI” and “state”.
 - No response parameter “authByQR”.
 - Response parameter “ticketID”: It is returned from “iAM Smart” System for “iAM Smart” APIs for anonymous request. It will be used for invoking subsequent “iAM Smart” APIs “Request QR Page” and “Open “iAM Smart” Mobile App for Getting Context” depending on the scenario. It is a 36-byte (or less) UUID number (ASCII character set).

3.6.1.2 Implementing (2) GET: Request QR Page



Pre-conditions

- Please refer to Section 3.4.1.1 except:
 - Online Service has the “ticketID” provided by “iAM Smart” System for the anonymous request in step 4.
 - Authorisation scope submitted should be “eidapi_formFilling”.

Post-conditions

- Please refer to Section 3.4.1.1.

Error conditions

- Please refer to Section 3.4.1.1.

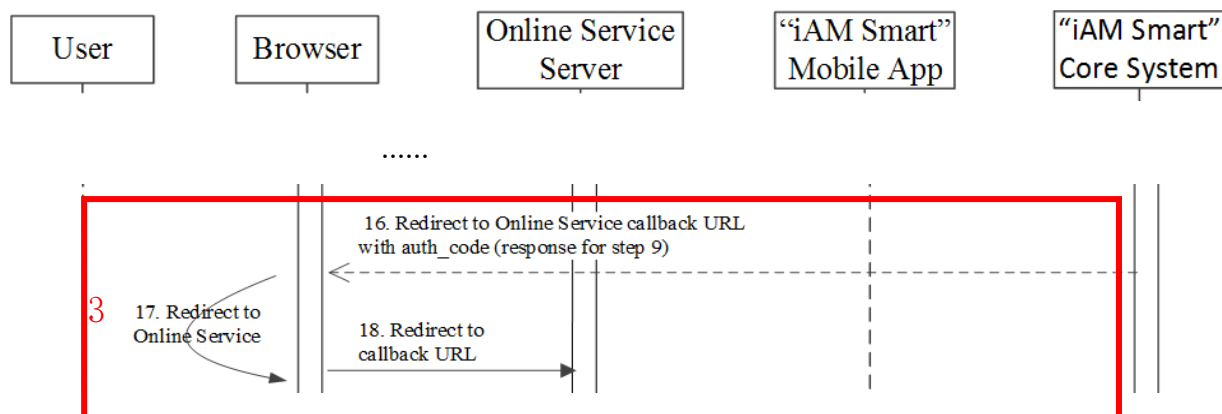
Request Parameters

- Please refer to Section 3.4.1.1.

Notes

- Please refer to Section 3.4.1.1 except the following parameter:
 - Request parameter “ticketID”: Value provided by “iAM Smart” System for the anonymous request.
 - Request parameter “scope”: It should be “eidapi_formFilling”.

3.6.1.3 Implementing (3) GET: Callback with authCode to Online Service Server



Pre-conditions

- Please refer to Section 3.4.1.2.

Post-conditions

- Please refer to Section 3.4.1.2 except:

- Online Service Server should match the callback result with corresponding Online Service client terminal using the “businessID”.

Error conditions

- This Online Service callback API is a HTTP GET request. If parameter “error_code” is returned, it means the request is failed.

Error Code	Error Description	Suggested Action
error_code - D60000	User cancelled form filling request	Inform user the Form Filling request is cancelled
error_code - D60001	User rejected form filling request	Inform user the Form Filling request is rejected
error_code - D60002	Failed to request form filling	Inform user the Form Filling request is failed and retry later

The error_code D60003 (Form Filling request timeout) does not appear in this scenario. For the QR code timeout or request confirmation timeout in the “iAM Smart” Mobile App, message will be prompted in the corresponding user interface.

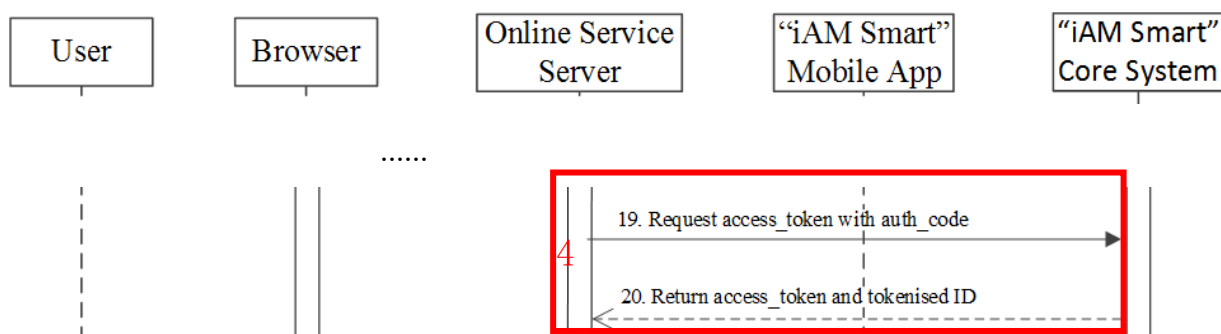
Callback Parameters

- Please refer to Section 3.4.1.2.

Notes

- Please refer to Section 3.4.1.2 except the following parameter:
 - Callback parameter “businessID”: Value returned by “iAM Smart” System should be the same one as Online Service submitted in the anonymous request.

3.6.1.4 Implementing (4) POST: Request accessToken & Tokenised ID



Pre-conditions

- Please refer to Section 3.4.1.3.

Post-conditions

- Please refer to Section 3.4.1.3. except:
 - The accessToken can only be used once before expiry.

Error conditions

- Please refer to Section 3.4.1.3.

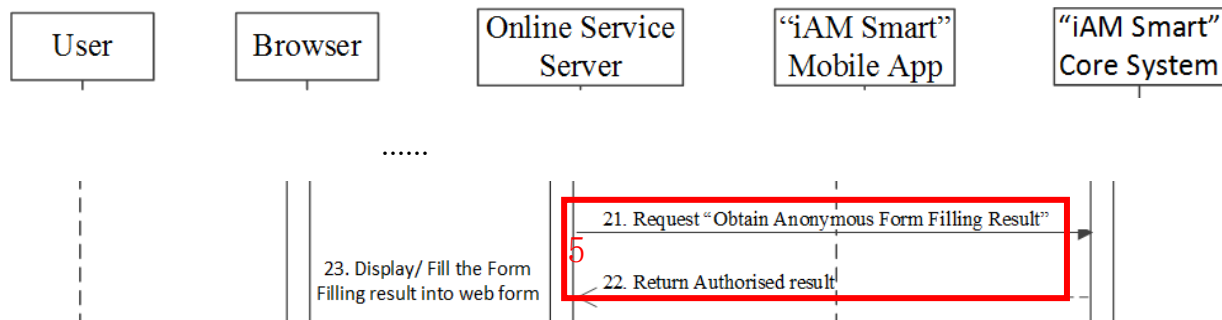
Request and Response Parameters

- Please refer to Section 3.4.1.3.

Notes

- Please refer to Section 3.4.1.3.

3.6.1.5 Implementing (5) POST: Obtain Anonymous Form Filling Result



Pre-conditions

- Online Service must possess a valid accessToken of the "iAM Smart" user with authorisation scope "form filling authorisation".

Post-conditions

- API data decryption is required.
- Online Service Server should check the requested data fields has authorised for the Anonymous Form Filling.
- Online Service should discard the accessToken as it can only be used once.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D60002	Failed to request Form Filling	Inform user the Form Filling request is failed and retry later
code - D60003	Form Filling request timeout	Inform user the Form Filling request is timeout, and provide way for user to retry

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- “e-ME” items not authorised by “iAM Smart” user will not return in result (i.e., JSON name/value of that item will not exist in result).

3.6.2 Scenario 2: Anonymous Form Filling (Online Service Website in Same Device)

The sequence diagram below shows how an anonymous user authorises an Online Service to use his/her “eMEFields” for form filling when Online

Service website and the “iAM Smart” Mobile App are running in the same device.

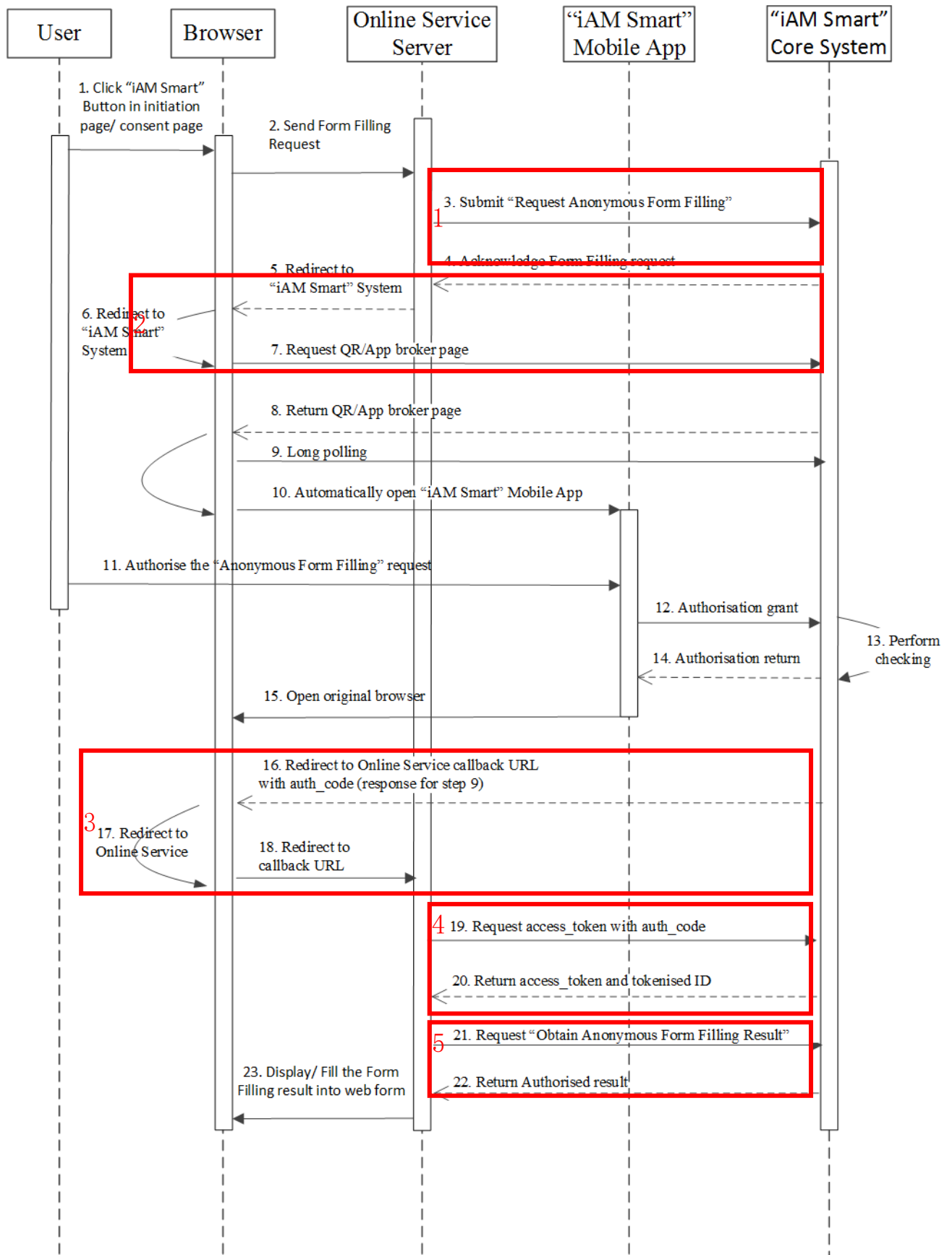


Figure-11 Anonymous Form Filling (Online Service Website in Same Device)

- Step 1. User reads the consent page (if “iAM Smart” profile fields requested) and clicks the button in Online Service Website;
- Step 2. Online Service Website initiates anonymous form filling request to Online Service Server with browser's user agent name (use as value for request parameter “source”);
- Step 3. Online Service Server initiates an anonymous Form Filling request to invoke the “iAM Smart” API with the “businessID” of this request, required data fields, etc. API data encryption is required;
“iAM Smart” API (POST: Request Anonymous Form Filling)
- Step 4. “iAM Smart” System verifies and confirms receipt of the anonymous Form Filling request to Online Service Server by returning POST response with parameter “ticketID”;
- Step 5-7. Online Service Server prepares the request parameters such as “clientID”, “redirectURI”, “source” (set as browser’s user agent value), “scope”, “brokerPage” (set as “true”), “ticketID”, etc. and constructs the GET request to invoke the “iAM Smart” API using browser redirection;
“iAM Smart” API (GET: Request QR page)
- Step 8. “iAM Smart” System returns the broker page to the Online Service Website;
- Step 9. Broker page of “iAM Smart” System polls “iAM Smart” System for further action;
- Step 10-12. Broker page invokes the “iAM Smart” Mobile App in the same device automatically and sends it the relevant parameters. “iAM Smart” user logs in “iAM Smart” Mobile App, reviews the Form Filling authorisation request (e.g., continue or reject the request) and authorises those selected “eMEFields” to Online Service (e.g., authorise only his HKIC number and English name in “iAM Smart” account information and residential address in “e-ME profile” (unchecked other “eMEFields”));
- Step 13-15. “iAM Smart” System verifies the validity of necessary information, and return the authorisation result to “iAM Smart” Mobile App.

“iAM Smart” Mobile App invokes the original Online Service browser (i.e., using the browser's user agent value submitted);

- Step 16-18. Broker page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e., response for the polling in Step 9) which includes authCode and businessID or any error code (e.g., “iAM Smart” user rejects the request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

- Step 19-20. When Online Service Server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

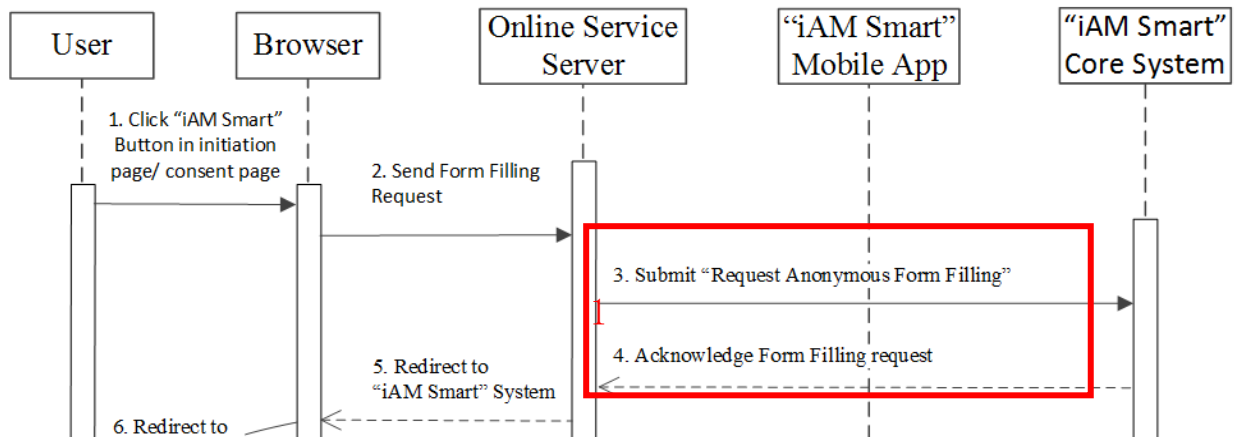
“iAM Smart” API (POST: Request accessToken & Tokenised ID)

- Step 21-22. Online Service Server invokes the “iAM Smart” API with the accessToken and openID to obtain the result of anonymous form filling request. API data encryption is required;

“iAM Smart” API (POST: Obtain Anonymous Form Filling Result)

- Step 23. Online Service Server processes and shows the result in the corresponding Online Service Website.

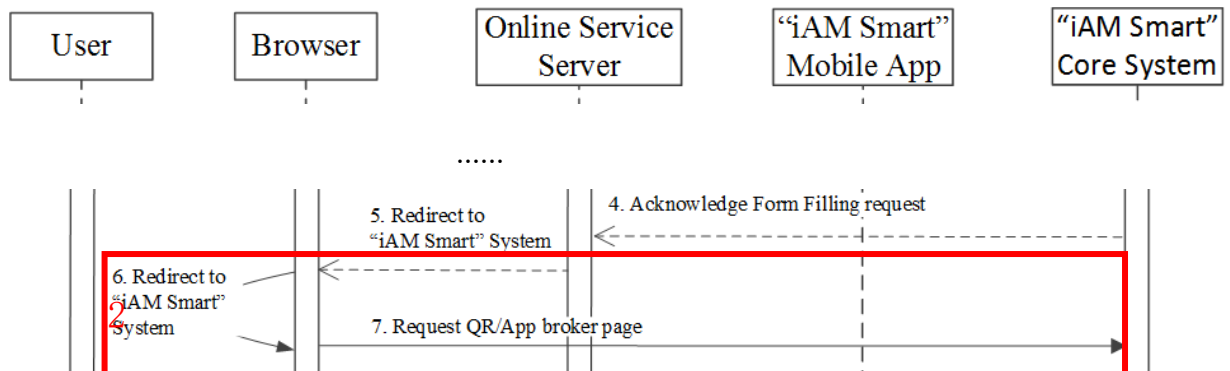
3.6.2.1 Implementing (1) POST: Request Anonymous Form Filling



Please refer to Section 3.6.1.1 except:

- Online Service Website and "iAM Smart" Mobile App are in the same device in this scenario.

3.6.2.2 Implementing (2) GET: Request QR Page



Pre-conditions

- Please refer to Section 3.4.2.1 except:
 - Online Service has the "ticketID" provided by "iAM Smart" System for the anonymous request in step 4.
 - Authorisation scope submitted should be "eidapi_formFilling".

Post-conditions

- Please refer to Section 3.4.2.1.

Error conditions

- Please refer to Section 3.4.2.1.

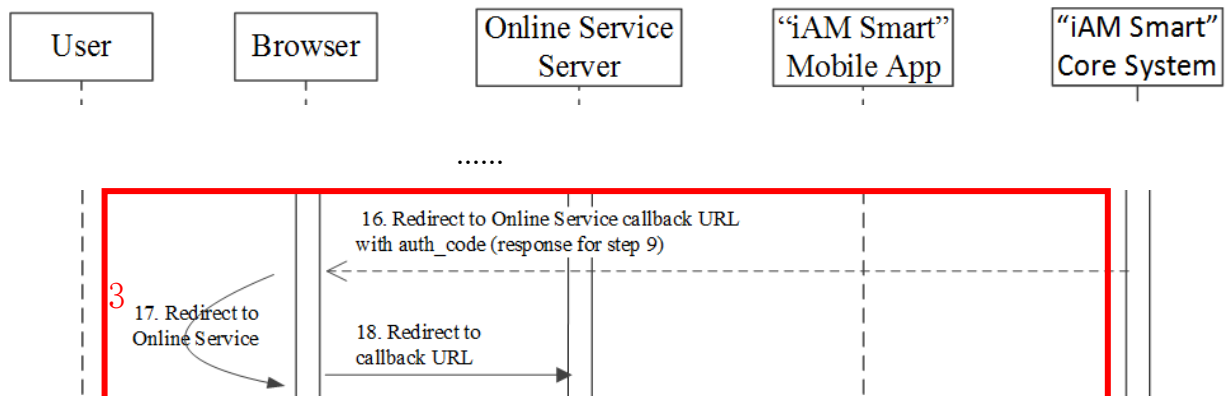
Request Parameters

- Please refer to Section 3.4.2.1.

Notes

- Please refer to Section 3.4.2.1 except the following parameter:
 - Request parameter “ticketID”: provided by “iAM Smart” System for the anonymous request.
 - Request parameter “scope”: It should be “eidapi_formFilling”.

3.6.2.3 Implementing (3) GET: Callback with authCode to Online Service Server



Pre-conditions

- Please refer to Section 3.4.2.2.

Post-conditions

- Please refer to Section 3.4.2.2 except:
 - Online Service Server should match the callback result with corresponding Online Service Website using the “businessID”.

Error conditions

- Please refer to Section 3.4.2.2

Callback Parameters

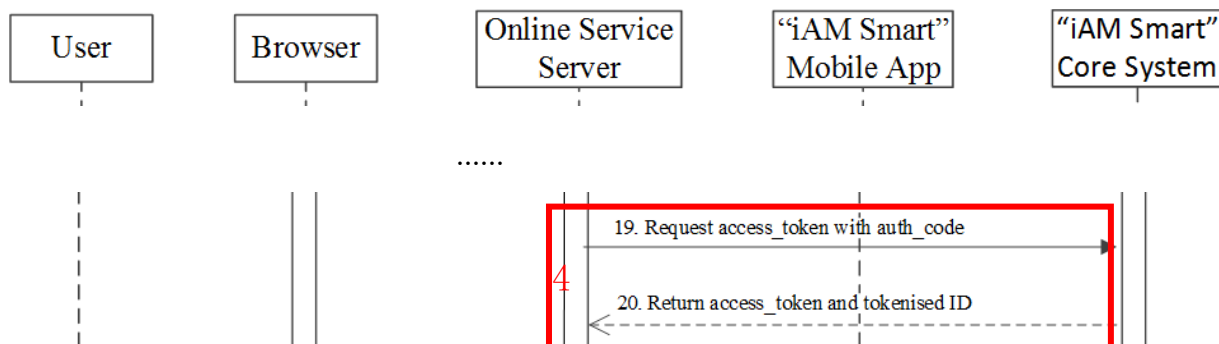
- Please refer to Section 3.4.2.2.

Notes

- Please refer to Section 3.4.2.2 except the following parameter:

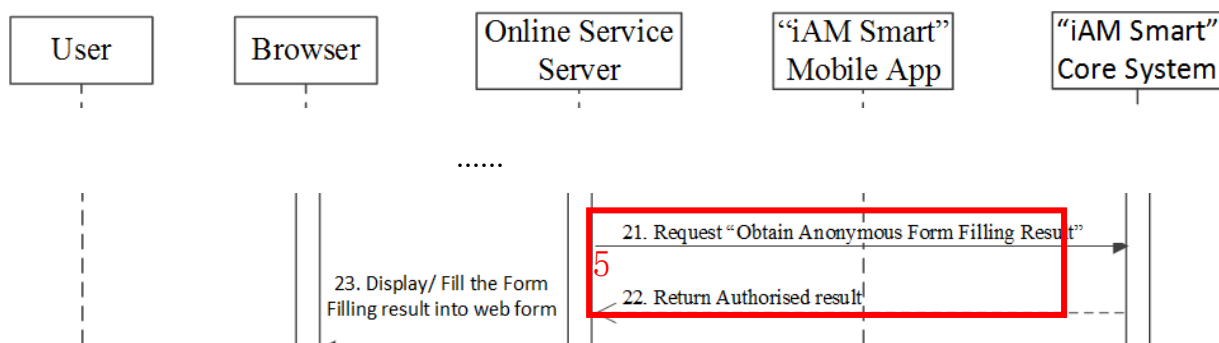
- Callback parameter “businessID”: Value returned by “iAM Smart” System should be the same one as Online Service submitted in the anonymous request.

3.6.2.4 Implementing (4) POST: Request accessToken & Tokenised ID



Please refer to Section 3.8.1.5.

3.6.2.5 Implementing (5) POST: Obtain Anonymous Form Filling Result



Please refer to Section 3.8.1.6.

3.6.3 Scenario 3: Anonymous Form Filling (Online Service App in Different Device)

The sequence diagram below shows how an anonymous user authorises an Online Service to use his/her “eMEFields” for form filling when Online

Service App and the “iAM Smart” Mobile App are running in different devices.

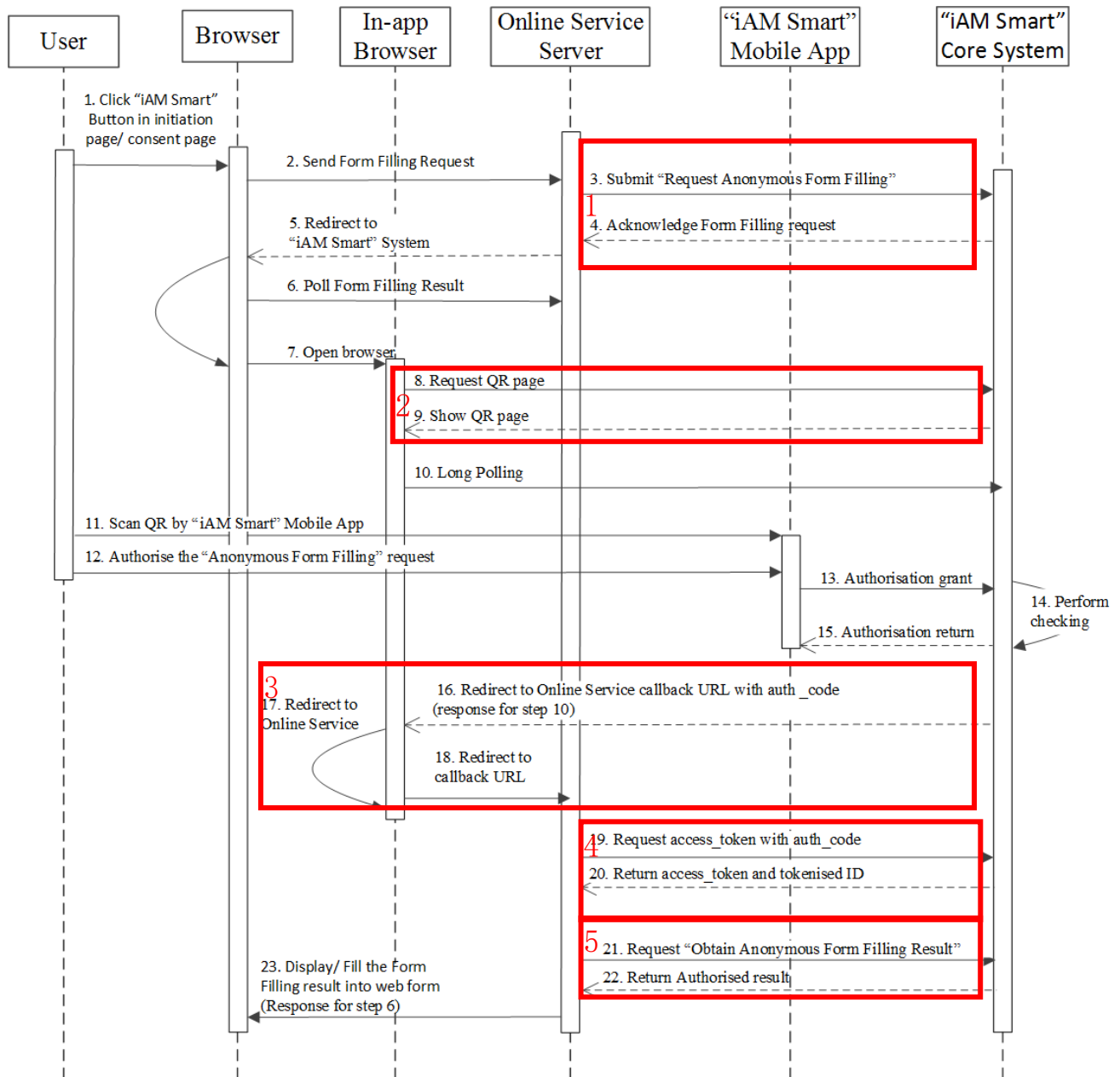


Figure-12 Anonymous Form Filling (Online Service App in Different Device)

- Step 1. User reads the consent page (if “iAM Smart” profile fields requested) and clicks the button in Online Service App;
- Step 2. Online Service App determines there is no “iAM Smart” Mobile App in the device using program code, initiates anonymous form filling request to Online Service Server with the in-app browser’s user agent value (use as value for request parameter “source”);

- Step 3. Online Service Server initiates an anonymous Form Filling request to invoke the “iAM Smart” API with the “businessID” of this request, required data fields, etc. API data encryption is required;
“iAM Smart” API (POST: Request Anonymous Form Filling)
- Step 4. “iAM Smart” System verifies and confirms receipt of the anonymous Form Filling request to Online Service Server by returning POST response with parameter “ticketID”;
- Step 5. After receiving the response, Online Service Server prepares necessary parameters such as “clientID”, “redirectURI”, “scope”, “source” (set as in-app browser’s user agent value), “brokerPage” (set as “false”), “ticketID”, etc. and constructs the GET request to invoke the “iAM Smart” API by Online Service App;
- Step 6. Online Service App polls Online Service Server for the form filling result from “iAM Smart” System;
- Step 7. Online Service App invokes in-app browser (Safari for iOS, Chrome for Android) to submit the GET request;
- Step 8-9. Browser opens the GET request to invoke the “iAM Smart” API and QR Code page will be shown;
“iAM Smart” API (GET: Request QR page)
- Step 10. QR Code page of “iAM Smart” System polls “iAM Smart” System for the QR Code status;
- Step 11-13. “iAM Smart” user logs in “iAM Smart” Mobile App to scan the QR Code, reviews the Form Filling authorisation request (e.g., continue or reject the request) and authorises those selected form filling items to Online Services (e.g., authorise only his HKIC number and residential address but no other items);
- Step 14-15. “iAM Smart” System verifies the validity of QR code and other necessary information, and return the authorisation result to “iAM Smart” Mobile App;
- Step 16-18. QR Code page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e., response for the polling in Step 10) which includes authCode and businessID or any error code (e.g., “iAM Smart” user rejects the request), assembles and

invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

Step 19-20. When Online Service Server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

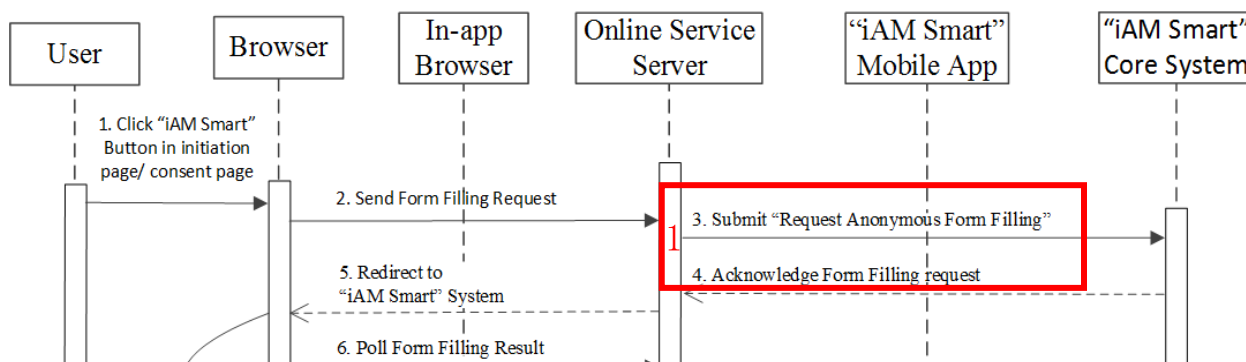
“iAM Smart” API (POST: Request accessToken & Tokenised ID)

Step 21-22. Online Service Server invokes the “iAM Smart” API with the accessToken and openID to obtain the result of anonymous form filling request. API data encryption is required;

“iAM Smart” API (POST: Obtain Anonymous Form Filling Result)

Step 23. Online Service Server processes and shows the result in the corresponding Online Service App (i.e., response for the polling in Step 6).

3.6.3.1 Implementing (1) POST: Request Anonymous Form Filling



Pre-conditions

- Please refer to Section 3.6.1.1 except:
 - Online Service should determine the “iAM Smart” Mobile App is not in the same device of Online Service App using program code.

Post-conditions

- Please refer to Section 3.6.1.1.

Error conditions

- Please refer to Section 3.6.1.1.

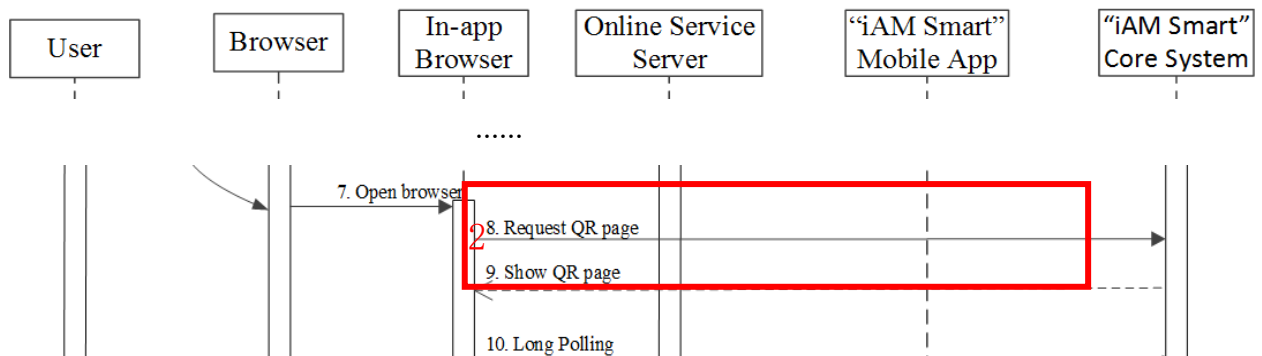
Request and Response Parameters

- Please refer to Section 3.6.1.1.

Notes

- Please refer to Section 3.6.1.1, except for the following parameter:
 - Request parameter “source”: Value should be the in-app browser's user agent value.

3.6.3.2 Implementing (2) GET: Request QR Page



Pre-conditions

- Please refer to Section 3.4.3.1 except:
 - Online Service has the “ticketID” provided by “iAM Smart” System for the anonymous request in step 4.
 - Authorisation scope submitted should be “eidapi_formFilling”.

Post-conditions

- Please refer to Section 3.4.3.1.

Error conditions

- Please refer to Section 3.4.3.1.

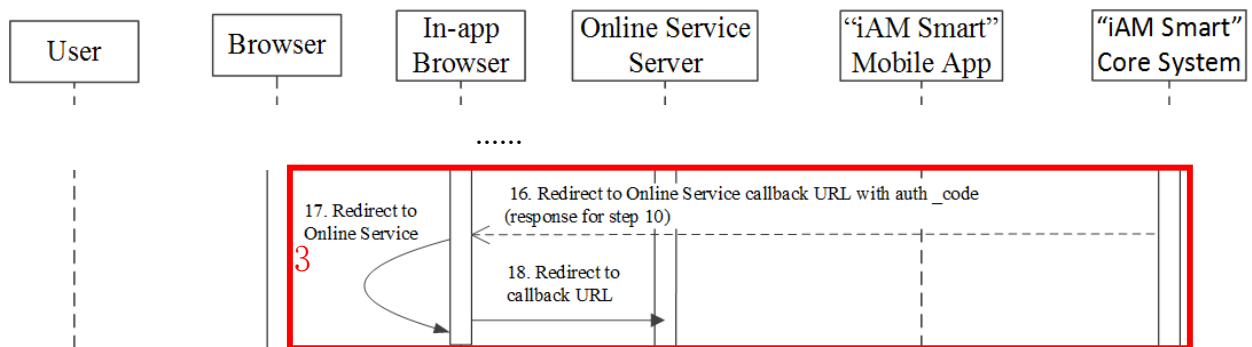
Request Parameters

- Please refer to “iAM Smart” API Specification.

Notes

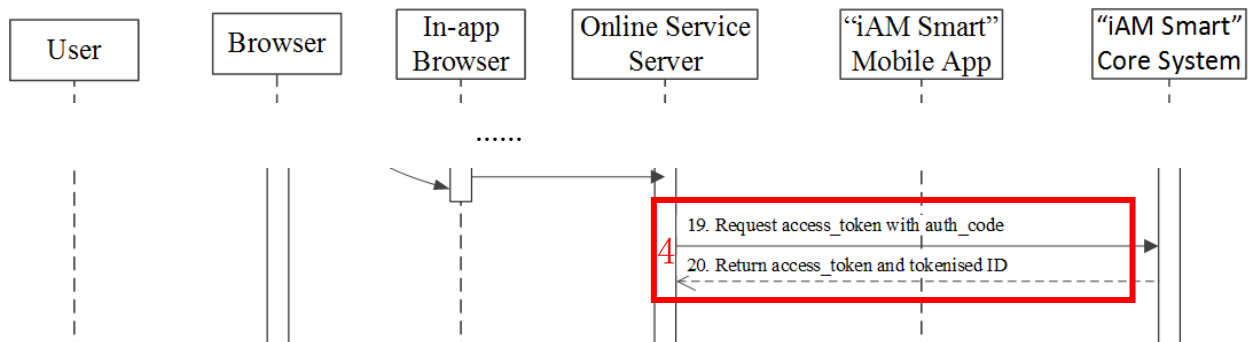
- Please refer to Section 3.4.3.1 except the following parameter:
 - Request parameter “ticketID”: Value provided by “iAM Smart” System for the anonymous request.
 - Request parameter “scope”: It should be “eidapi_formFilling”.

3.6.3.3 Implementing (3) GET: Callback with authCode to Online Service Server



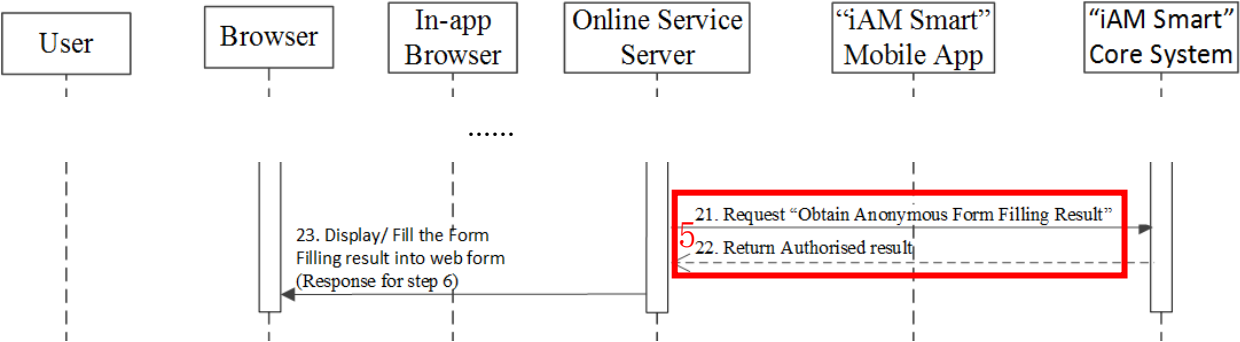
Please refer to Section 3.8.1.4.

3.6.3.4 Implementing (4) POST: Request accessToken & Tokenised ID



Please refer to Section 3.8.1.5.

3.6.3.5 Implementing (5) POST: Obtain Anonymous Form Filling Result



Please refer to Section 3.6.1.5.

3.6.4 Scenario 4: Anonymous Form Filling (Online Service App in Same Device)

The sequence diagram below shows how an anonymous user authorises an Online Service to use his/her “eMEFields” for form filling when Online Service App and the “iAM Smart” Mobile App are running in the same device.

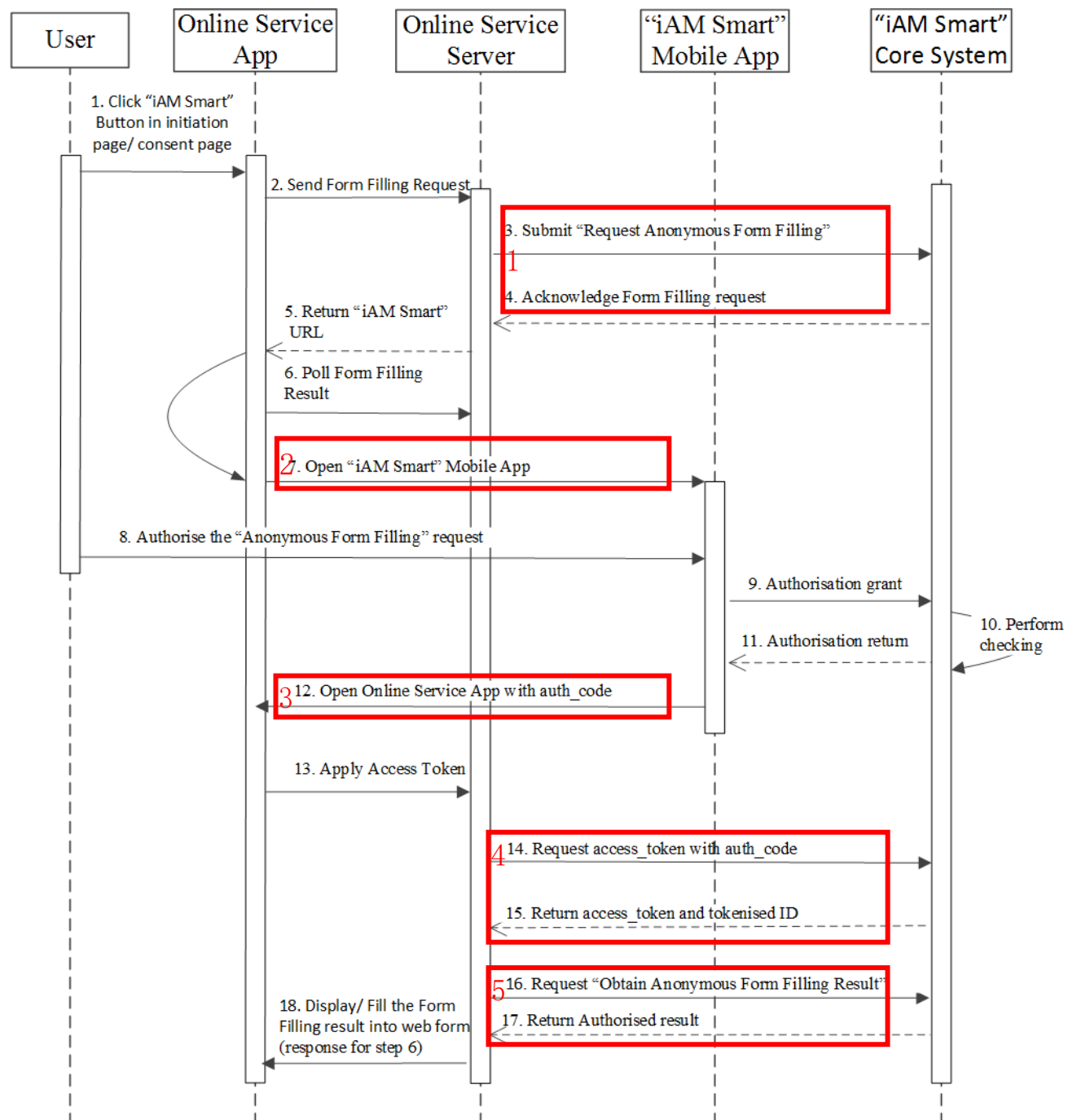


Figure-13 Anonymous Form Filling (Online Service App in Same Device)

- Step 1. User reads the consent page (if “iAM Smart” profile fields requested) and clicks the button in Online Service App;
- Step 2. Online Service App determines “iAM Smart” Mobile App is installed in the device using program code, initiates anonymous form filling request to Online Service Server with “App_Scheme” or “App_Link” (use as value for request parameter “source”);
- Step 3. Online Service Server initiates an anonymous Form Filling request to invoke the “iAM Smart” API with the “businessID” of this request, required data fields, etc. API data encryption is required;
“iAM Smart” API (POST: Request Anonymous Form Filling)
- Step 4. “iAM Smart” System verifies and confirms receipt of the anonymous Form Filling request to Online Service Server by returning POST response with parameter “ticketID”;
- Step 5. After receiving the response, Online Service Server prepares necessary parameters such as “clientID”, “redirectURI” (set as Online Service App URL Scheme or Universal Link/App Link depending on “source” parameter), “scope”, “source” (set as “App_Scheme” or “App_Link”), “ticketID”, etc. and constructs the URL Scheme to invoke “iAM Smart” Mobile App by Online Service App;
- Step 6. Online Service App polls Online Service Server for the form filling result from “iAM Smart” System;
- Step 7. Online Service App invokes “iAM Smart” Mobile App using URL Scheme with request parameters (set <Context> as “anon_form-filling” in URL Scheme);
“iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Getting Context)
- Step 8-9. “iAM Smart” user logs in “iAM Smart” Mobile App, reviews the Form Filling authorisation request (e.g., continue or reject the request) and authorises those selected form filling items to Online Service (e.g., authorise only his HKIC number and residential address but no other items);
- Step 10-11. “iAM Smart” System verifies the validity of necessary information, and return the authorisation result to “iAM Smart” Mobile App.

Step 12. “iAM Smart” Mobile App invokes Online Service App using URL Scheme or Universal Link/App Link with required parameters such as “authCode”, “businessID”;

Online Service Callback API (Callback with authCode to Online Service App)

Step 13. Online Service App sends authCode, businessID, etc. to Online Service Server;

Step 14-15. When Online Service Server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

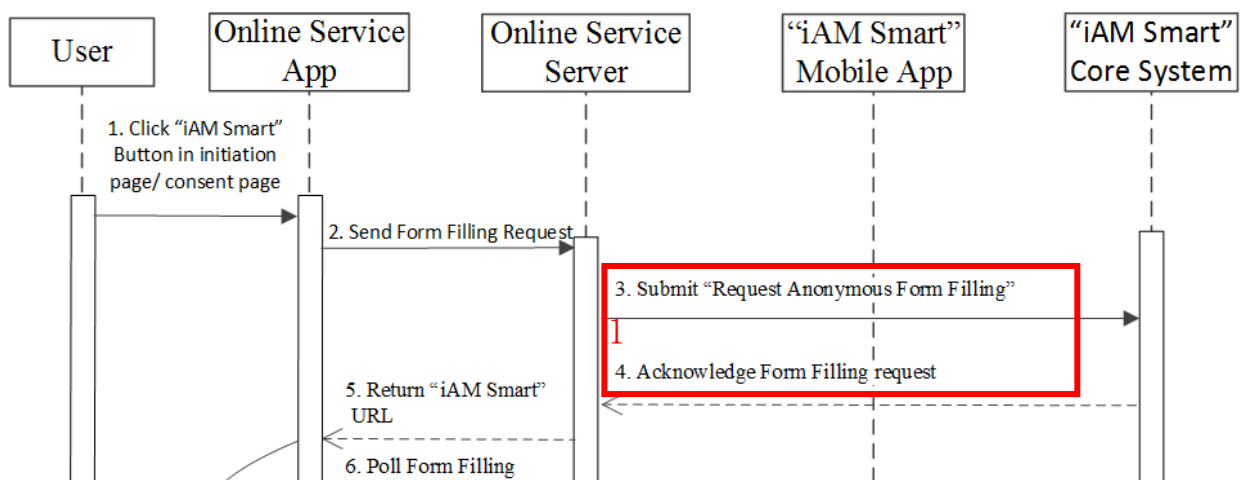
“iAM Smart” API (POST: Request accessToken & Tokenised ID)

Step 16-17. Online Service Server invokes the “iAM Smart” API with the accessToken and openID to obtain the result of anonymous form filling request. API data encryption is required;

“iAM Smart” API (POST: Obtain Anonymous Form Filling Result)

Step 18. Online Service Server processes and shows the result in the corresponding Online Service App (i.e., response for the polling in Step 6).

3.6.4.1 Implementing (1) POST: Request Anonymous Form Filling



Pre-conditions

- Please refer to Section 3.6.1.1 except:
 - Online Service should determine the “iAM Smart” Mobile App is on the same device of Online Service App using program code.

Post-conditions

- Please refer to Section 3.6.1.1.

Error conditions

- Please refer to Section 3.6.1.1.

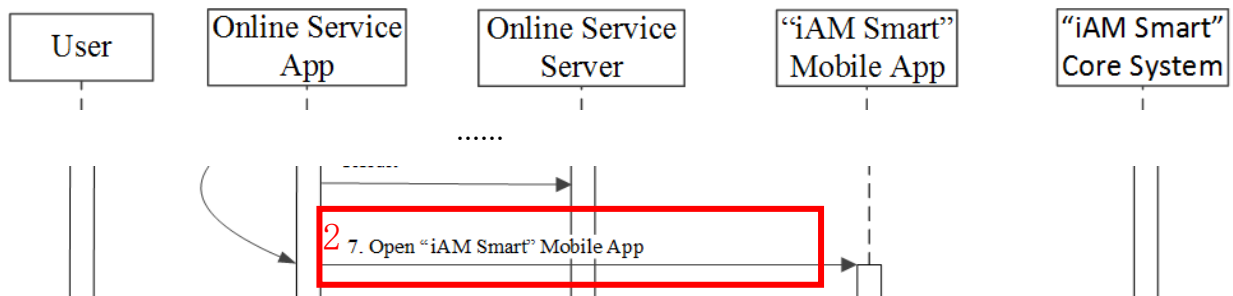
Request and Response Parameters

- Please refer to Section 3.6.1.1.

Notes

- Please refer to Section 3.6.1.1.

3.6.4.2 Implementing (2) URL Scheme: Open “iAM Smart” Mobile App for Getting Context



Pre-conditions

- Online Service App and “iAM Smart” Mobile App are in the same user device.
- Online Service has the “ticketID” provided by “iAM Smart” System for the anonymous request in step 4.
- Online Service should determine if it would generate the optional “state” request parameter to prevent CSRF attack. The “state” will be returned to Online Service for checking through the Online Service callback API for receiving the authorisation code from “iAM Smart” System in Step 12.

Post-conditions

- “iAM Smart” Mobile App will be launched.
- Online Service App should keep synchronising with Online Service server for the result from “iAM Smart” System.

Error conditions

- Nil

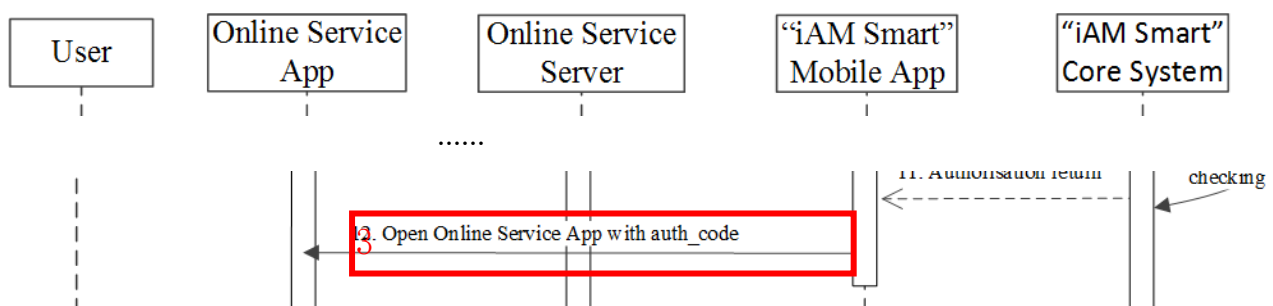
Request Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Request parameter “ticketID”: Value provided by “iAM Smart” System for the anonymous request.
- Request parameter “source”: Value should be “App_Scheme” (for Online Service App URL scheme), or “App_Link” (for the Universal Link/App Link).
- Request parameter “redirectURI”: This is Online Service App URL scheme or Universal Link/App Link depending on the “source” parameter, which is used for receiving the authorisation code. The value should be URL encoded. For details, please refer to Section 6.4.1 of “iAM Smart” API Specification. The value must be the same as provided during Online Service registration.
- Request parameter “state”: The value should be URL encoded.
- Set <Context> as “anon_form-filling” in URL Scheme.

3.6.4.3 Implementing (3) Callback with authCode to Online Service App



Pre-conditions

- Please refer to Section 3.4.2.2.

Post-conditions

- Please refer to Section 3.4.2.2 except:
 - Online Service Server should match the callback result with corresponding Online Service App using the “businessID”.

Error conditions

- This Online Service callback API is a URL Scheme, or Universal Link/App Link. If parameter “error_code” is returned, it means the authentication request is failed.

Error Code	Error Description	Suggested Action
error_code - D60000	User cancelled form filling request	Inform user the Form Filling request is cancelled
error_code - D60001	User rejected form filling request	Inform user the Form Filling request is rejected
error_code - D60002	Failed to request form filling	Inform user the Form Filling request is failed and retry later
error_code - D60003	Form Filling request timeout	Inform user the Form Filling request is timeout, and provide way for user to retry

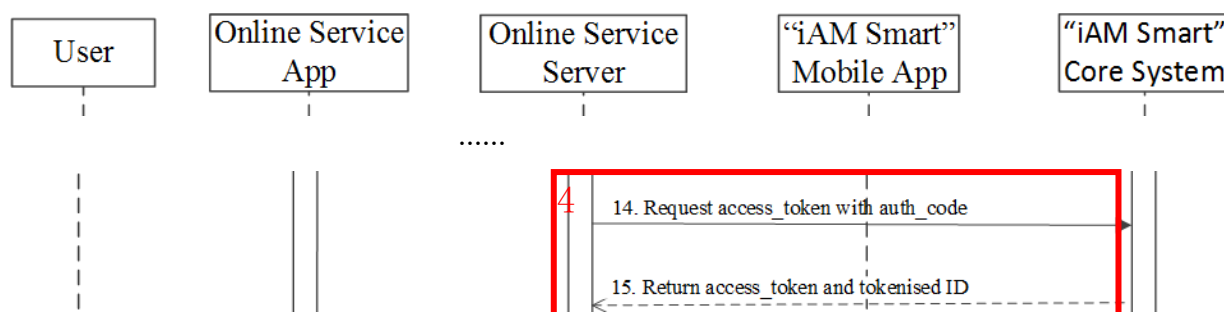
Callback Parameters

- Please refer to “iAM Smart” API Specification.

Notes

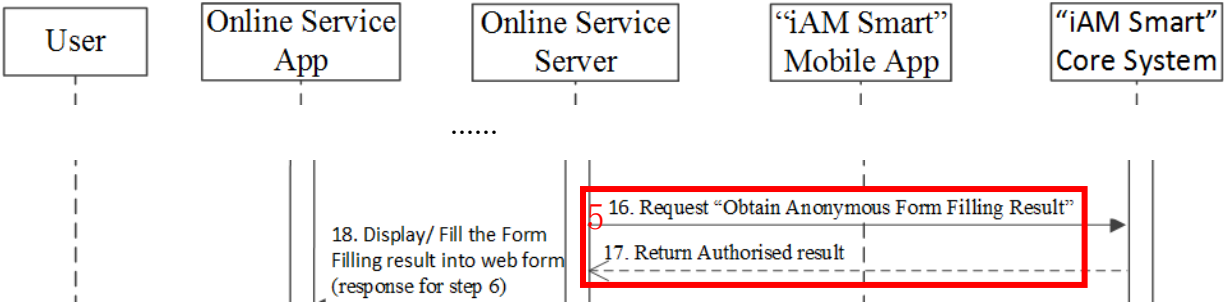
- Please refer to Section 3.4.2.2 except the following parameter:
 - Callback parameter “businessID”: Value returned by “iAM Smart” System should be the same one as Online Service submitted in the anonymous request.

3.6.4.4 Implementing (4) POST: Request accessToken & Tokenised ID



Please refer to Section 3.8.1.5.

3.6.4.5 Implementing (5) POST: Obtain Anonymous Form Filling Result



Please refer to Section 3.8.1.6.

3.7 WORKFLOWS FOR DIGITAL SIGNING WITH SERVICE LOGIN

The following process includes non-anonymous hash digital signing and non-anonymous pdf digital signing.

3.7.1 Scenario 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.

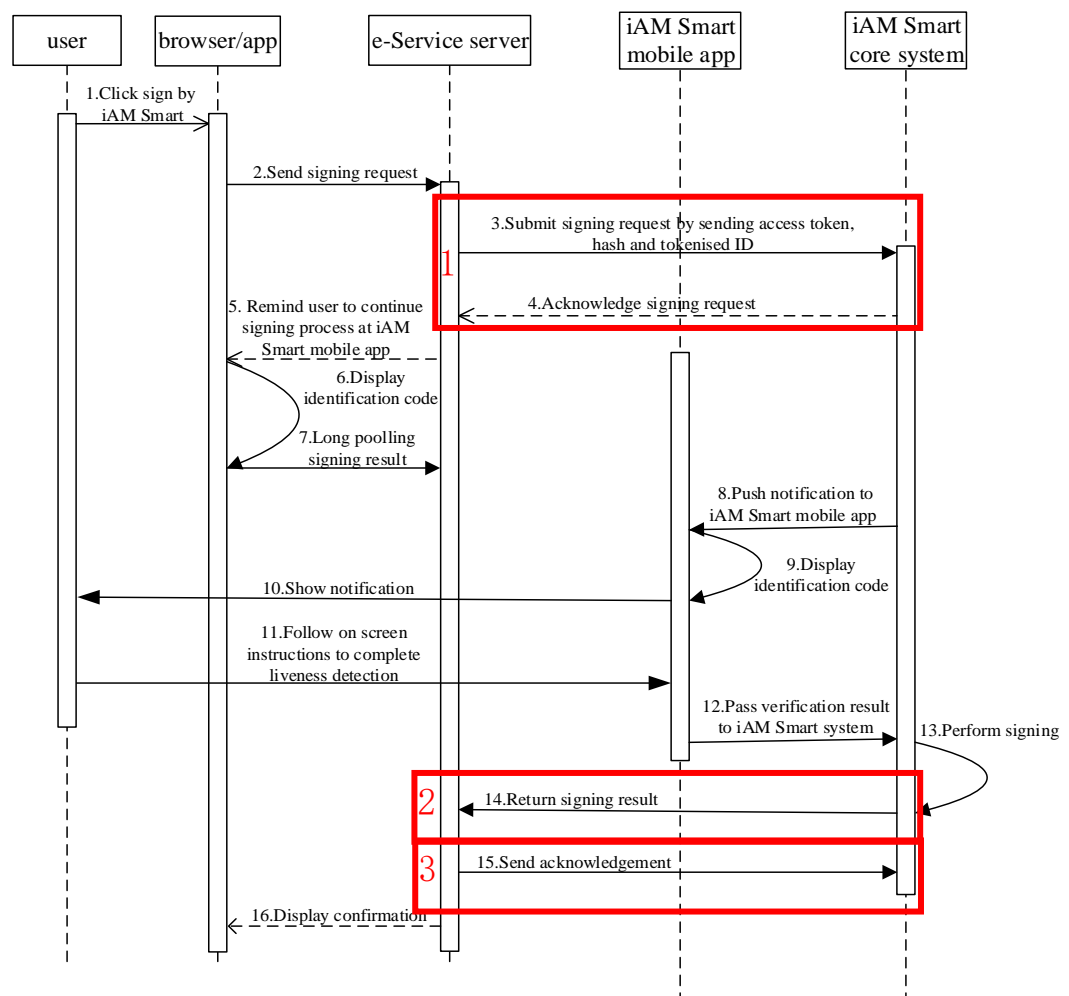


Figure-14 Digital Signing (Online Service Website/App in Different Device)

- Step 1. User clicks the “Sign by iAM Smart” button in Online Service Website/App;
- Step 2-3. Online Service initiate digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, accessToken,

Tokenised ID, Document Hash / PDF digest, HKICHash, signature algorithm (only for "*Request Digital Signing*"), etc. The request parameter "source" will be the browser's user agent value (for Online Service Website) or "App_Scheme"/"App_Link" (for Online Service App) in this scenario. API encryption is required;

"iAM Smart" API (POST: Request Digital Signing)

"iAM Smart" API (POST: Request PDF Digital Signing)

- Step 4. "iAM Smart" System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the digital signing request to Online Service server by returning POST response with parameter "authByQR" (set to "true") in this scenario;
- Step 5-6. Online Service Server uses the Document Hash / PDF digest and hash of Tokenised ID to calculate a 4-digit identification code and instructs Online Service Website/App to display the instructions with the 4-digit identification code to inform "iAM Smart" user to process the digital signing authorisation request in "iAM Smart" Mobile App;
- Step 7. Online Service Website/App should keep synchronising with Online Service Server for the digital signing processing result (e.g., polling);
- Step 8. "iAM Smart" System pushes a notification message to the "iAM Smart" Mobile App based on the Tokenised ID;
- Step 9. "iAM Smart" user logs in "iAM Smart" Mobile App, reviews the digital signing authorisation request (e.g., continue or reject the request) and verifies the 4-digit identification code with Online Service Website/App;
- Step 10-13. "iAM Smart" user follows the instructions in "iAM Smart" Mobile App to complete digital signing operation;
- Step 14. "iAM Smart" System invokes Online Service callback API to return the result with "businessID" of the digital signing request. API data decryption is required;

Online Service Callback API (POST: Callback to Receive Digital Signing Result)

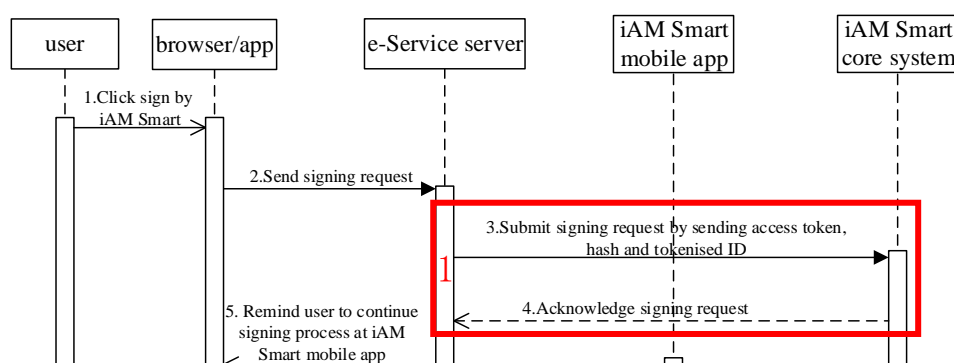
Online Service Callback API (POST: Callback to Receive PDF Digital Signing Result)

- Step 11. Online Service Server completes the document digital signing process, verify the result and acknowledges “iAM Smart” System the digital signing result using “iAM Smart” API with the same “businessID” of the digital signing request. API data encryption is required;

“iAM Smart” API (POST: Online Service Acknowledges Digital Signing Result)

- Step 12. Online Service Server matches the result using the “businessID” and shows the digital signing result in corresponding Online Service Website/App.

3.7.1.1 Implementing (1) POST: Request Digital Signing



Pre-conditions

- Online Service must possess a valid accessToken of the “iAM Smart” user with authorisation scope “Signing authorisation”.
- Online Service Website/App and “iAM Smart” Mobile App are in different devices in this scenario.
- The “iAM Smart” user who signs the document must have a sign version “iAM Smart”.
 - Online Service generates a “businessID” for this Digital Signing request and uses this identifier to match the callback result returned from “iAM Smart” System.
- Online Service generates the Document Hash as “hashCode” parameter from the original documents to be signed using a hash algorithm that can suit the

specific business need. It is recommended to use a hash algorithm that is at least SHA-256 or equivalent.

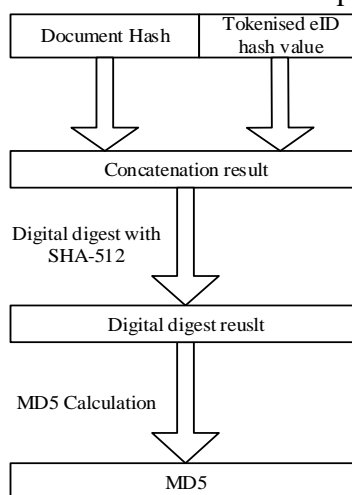
- API data encryption is required.

Post-conditions

- Online Service server should check the response parameter “authByQR” and “ticketID” and determine the next action. For details, please refer to Section 3.2.1. “ticketID” will not be provided by “iAM Smart” System in this scenario.
- Online Service has to calculate a 4-digit identification code using the Document Hash and Tokenised ID hash value. Online Service Website/App should show the 4-digit identification code and instruction to inform “iAM Smart” user to process the digital signing request in “iAM Smart” Mobile App when “authByQR” is “true”.

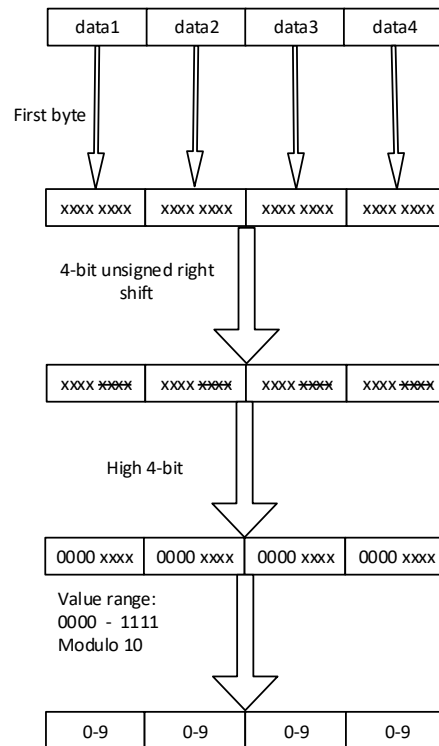
4-digit identification code generation algorithm is as follows:

- 1) Concatenate the Document Hash and the Tokenised ID hash value (calculated with SHA-512), and perform the digital digest for the concatenation result with SHA-512.
- 2) Perform the MD5 calculation to obtain a unique MD5 hash value.



- 3) Divide the 128-bit MD5 hash value into four groups of 4 bytes binary value and then extract the first byte from each group as a data sample.
- 4) Perform 4-bit unsigned right shift to each byte of the result obtained above for the hexadecimal data.
- 5) Perform the modulo operation on each byte (modulo 10) to obtain a number of 0-9.

- 6) Assemble the 4 numbers obtained above as a 4-digit identification code.



Pseudo code for generating 4-digit identification code

```
private static final char hexDigits[] = {'0', '1', '2', '3', '4',
'5', '6', '7', '8', '9'};

public static String getSignCode (String hashCode, String openID)
{
    // Assume Document Hash is BASE64 encoded and perform decode
    byte[] hashCodeBytes = Base64Util.base64Decode(hashCode);
    // Calculate SHA-512 hashcode of Tokenised ID
    byte[] openIDBytes = DigestUtil.digestToByte(openID,
Alg.SHA512);
    char code[] = new char [4];
    byte[] source = new byte[hashCodeBytes.length +
openIDBytes.length];
    System.arraycopy(hashCodeBytes, 0, source, 0,
hashCodeBytes.length);
    System.arraycopy(openIDBytes, 0, source, hashCodeBytes.length,
openIDBytes.length);
    // Calculate SHA-512 hashcode from concatenated Document Hash
and
    // SHA-512 hashed Tokenised ID
    byte[] inData = DigestUtil.digestToByte(source, Alg.SHA512);
    // Calculate MD5 hashcode
    byte[] digestMD5Data = DigestUtil.digestToByte(inData, Alg.MD5);
    // Convert MD5 value 4-digit identification code
    for (int i = 0; i < 4; i++) {
        code[i] = hexDigits[(digestMD5Data[i * 4] >>> 4 & 0xf) %
10];
    }
    return new String(code);
}
```

- Online Service Website/App should keep synchronising with Online Service Server for the digital signing result returned from “iAM Smart” System.
- API data decryption is required.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D70002	Failed to request digital signing	Inform user the “iAM Smart” digital signing request is failed and retry later
code - D70004	User not allowed to sign	Inform user that “iAM Smart” digital signing is not allowed for default “iAM Smart” and suggest user to upgrade to “iAM Smart” with digital signing function
code - D70005	Inconsistent HKIC number	Hash of HKIC number provided is not matched with “iAM Smart” user. Check hash of HKIC number or inform user to authorise the digital signing using the correct “iAM Smart” account

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

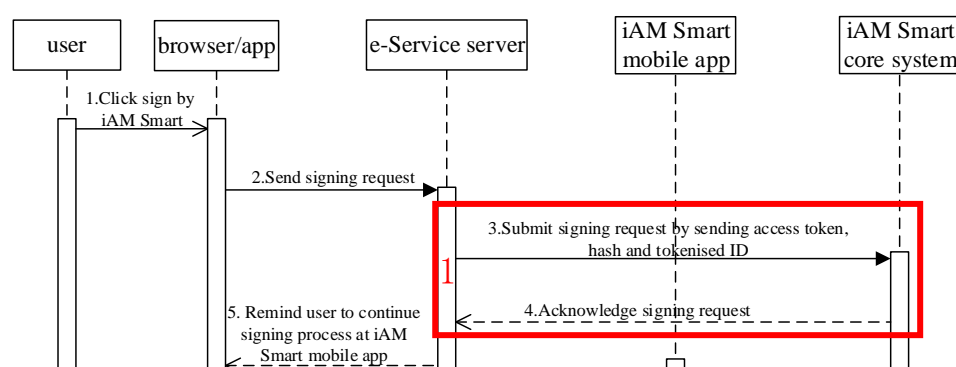
Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Request parameter “source”: Value should be matched with Online Service client terminal (e.g., “App_Scheme”/“App_Link” or browser’s user agent value).
- Request parameter “openID”: It is the Tokenised ID of the “iAM Smart” user. Online Service should ensure the accessToken and Tokenised ID are in pair and belongs to the target “iAM Smart” user for the request. The target “iAM Smart” user must have a sign version “iAM Smart”.
- Request Parameter - “redirectURI”: Value should equal to URI of “Callback to Receive Digital Signing Result” Online Service callback API. It must be in the same value as provided during Online Service registration.
- Request parameter “hashCode”: It is the Document Hash to be signed. SHA-256 or equivalent is recommended.
- Request parameter “sigAlgo”: It is the signature algorithm to be used by “iAM Smart”. The default value is "SHA256withRSA". While using "NONEwithRSA", hashCode provided must be hashed with SHA-256.
- Request parameter “HKICHash”: “iAM Smart” System will verify the HKIC hash of the “iAM Smart” user with the “HKICHash” provided by Online

Service, and proceed the digital signing only when they matched. Online Service should convert the HKIC number into hash value using SHA-256 before sending to “iAM Smart” System. Only the identifier of HKIC number will be hashed, no check digit is needed.

- Request parameters “department”, “serviceName”, “documentName”: They are the information of the document to be signed and will be shown in “iAM Smart” Mobile App for “iAM Smart” user's reference with the 4-digit identification code.
- Response parameter “authByQR”: The value returned by “iAM Smart” System will be “true” in this scenario.
- “iAM Smart” System will not return the response parameter “ticketID” in this scenario.

3.7.1.2 Implementing (1) POST: Request PDF Digital Signing



Pre-conditions

- Please refer to Section 3.7.1.1 except:
 - Online Service generates the digest as the parameter “docDigest” from the pdf document to be signed using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter.

Post-conditions

- Please refer to Section 3.7.1.1 except:
 - Computing 4-digit identification code with “docDigest” instead of “hashCode”.

Error conditions

- Please refer to Section 3.7.1.1.

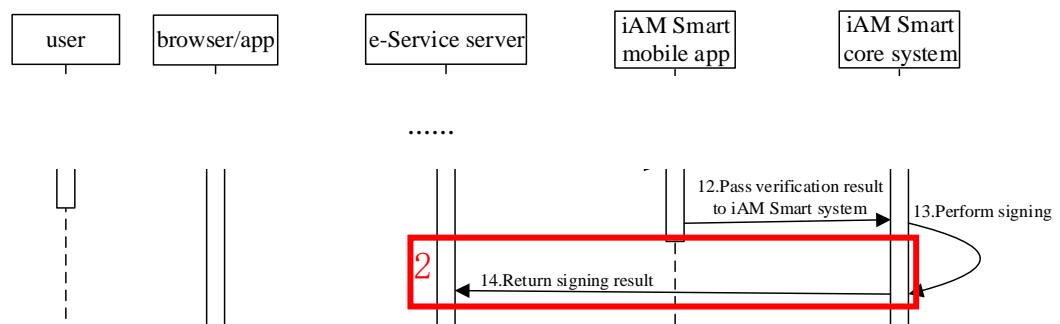
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.7.1.1, except “docDigest” request parameter is applied instead of “hashCode”.

3.7.1.3 Implementing (2) POST: Callback to Receive Digital Signing Result



Pre-conditions

- “iAM Smart” user can accept and complete the digital signing request.
- “iAM Smart” user can also reject the digital signing request.

Post-conditions

- API data decryption is required.
- Online Service Server should match the callback result with corresponding Online Service Website/App using the “businessID”.
- Online Service server completes the document digital signing process and acknowledges “iAM Smart” System the digital signing result.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D70000	User cancelled signing request	Inform user the “iAM Smart” digital signing request is cancelled
code - D70001	User rejected signing request	Inform user the “iAM Smart” digital signing request is rejected

code - D70002	Failed to request signing	Inform user the “iAM Smart” digital signing request is failed and retry later
code - D70003	Signing request timeout	Inform user the “iAM Smart” digital signing request is timeout and provide way for user to retry

Callback Parameters

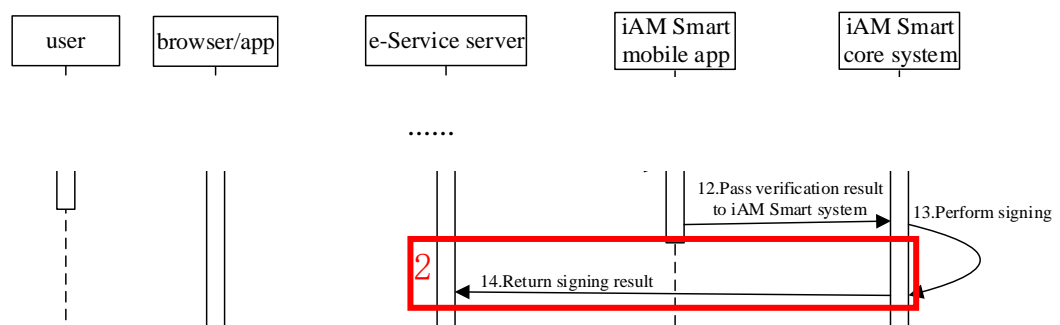
- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Callback parameter “hashCode”: It is the Document Hash provided by Online Service and Online Service may use for checking purpose.
- Callback parameter “timestamp”: It is the timestamp returned by “iAM Smart” System when the digital signing operation is successful.
- Callback parameter “signature”: It is the RSA signature of the Document Hash submitted by Online Service for digital signing. The value is BASE64 encoded.
- Callback parameter “cert”: It is the “iAM Smart” e-Cert of the “iAM Smart” user. It is in X.509 format and the value is BASE64 encoded.

3.7.1.4 Implementing (2) POST: Callback to Receive PDF Digital Signing

Result



Pre-conditions

- Please refer to Section 3.6.1.5.

Post-conditions

- Please refer to Section 3.6.1.5.

Error conditions

- Please refer to Section 3.6.1.5.

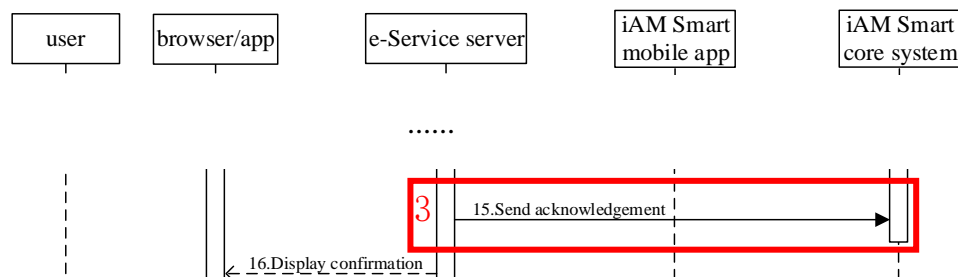
Callback Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Callback parameter “docDigest”: It is the pdf document digest submitted by Online Service and Online Service may use for checking purpose.
- Callback parameter “pdfSignature”: It is the 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” will provide this to Online Service only when the digital signing is successful.

3.7.1.5 Implementing (3) POST: Online Service Acknowledges Digital Signing Result



Pre-conditions

- Online Service receives the “iAM Smart” digital signing result and completes its document digital signing process (i.e., verify the result).
- API data encryption is required.

Post-conditions

- Online Service may record the digital signing acknowledgement result.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D70006	Failed to process signing acknowledgement	“iAM Smart” digital signing acknowledgement processing failed, retry later

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Request parameter “businessID”: It is the same “businessID” submitted in the digital signing request.
- Request parameter “signingResult”: It is the status of Online Service document digital signing process after receiving the “iAM Smart” digital signing result. Its value must be equal to those specified in this API.

3.7.2 Scenario 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.

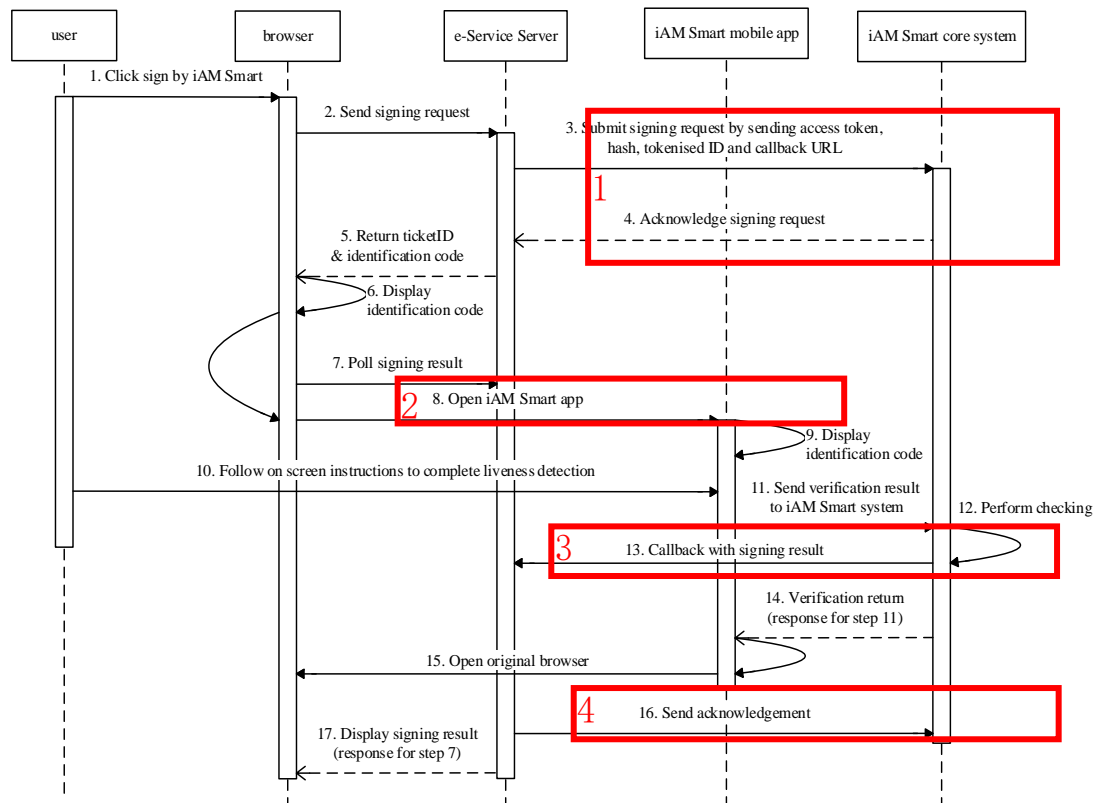


Figure-15 Digital Signing (Online Service Website in Same Device)

- Step 1. User clicks the “Sign by iAM Smart” button in Online Service Website;
- Step 2-3. Online Service Website initiates digital signing request to Online Service server with browser's user agent name (for use as value for request parameter “source”). Online Service initiates digital signing request to invoke “iAM Smart” API with the “businessID” of this request, accessToken, Tokenised ID, Document Hash / PDF digest, HKICHash, signature algorithm (only for “Request Digital Signing”), etc. The request parameter “source” will be browser's user agent value in this scenario. API data encryption is required; “iAM Smart” API (POST: Request Digital Signing)

“iAM Smart” API (POST: Request PDF Digital Signing)

- Step 4. “iAM Smart” System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the digital signing request to Online Service server by returning POST response with parameters “authByQR” (set to “false”) and “ticketID”;
- Step 5-6. Online Service Server uses the Document Hash / PDF digest and hash of Tokenised ID to calculate a 4-digit identification code. Online Service prepares and sends necessary parameters such as “authByQR”, “ticketID”, 4-digit identification code, etc. to Online Service Website and instructs Online Service Website to display the instruction with the 4-digit identification Code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;
- Step 7. Online Service Website should keep synchronising with Online Service Server for digital signing processing result (e.g., polling);
- Step 8. Online Service Website invokes “iAM Smart” Mobile App using URL Scheme with “ticketID” (set <Context> as “hash-sign” or “pdf-sign” in URL Scheme). Other parameters of this API are not used in this scenario.

“iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Getting Context)

- Step 9. “iAM Smart” user logs in “iAM Smart” Mobile App, reviews the digital signing authorisation request (e.g., continue or reject the request) and verify the 4-digit identification code with Online Service Website;
- Step 10-12. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 13-14. “iAM Smart” System invokes Online Service callback API to return the result with “businessID” of the digital signing request to Online Service server. API data decryption is required;

Online Service Callback API (POST: Callback to Receive Digital Signing Result)

Online Service Callback API (POST: Callback to Receive PDF Digital Signing Result)

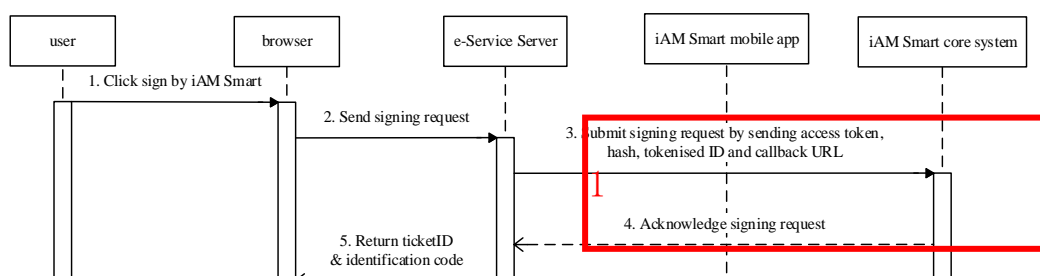
Step 15. “iAM Smart” System instructs “iAM Smart” Mobile App to invoke the original Online Service browser (i.e., using the browser's user agent name submitted in digital signing request in Step 3);

Step 16. Online Service Server completes the document digital signing process, verify the result and acknowledges “iAM Smart” System the digital signing result using “iAM Smart” API with the same “businessID” of the digital signing request. API data encryption is required;]

“iAM Smart” API (POST: Online Service Acknowledges Digital Signing Result)

Step 17. Online Service Server matches the result using the “businessID” and shows the digital signing result in corresponding Online Service Website.

3.7.2.1 Implementing (1) POST: Request Digital Signing



Pre-conditions

- Please refer to Section 3.7.1.1 except:
 - Online Service Website and “iAM Smart” Mobile App are in the same device in this scenario.

Post-conditions

- Please refer to Section 3.7.1.1 except:
 - Response “authByQR” is “false”, “ticketID” will be provided by “iAM Smart” System in this scenario.

Error conditions

- Please refer to Section 3.7.1.1.

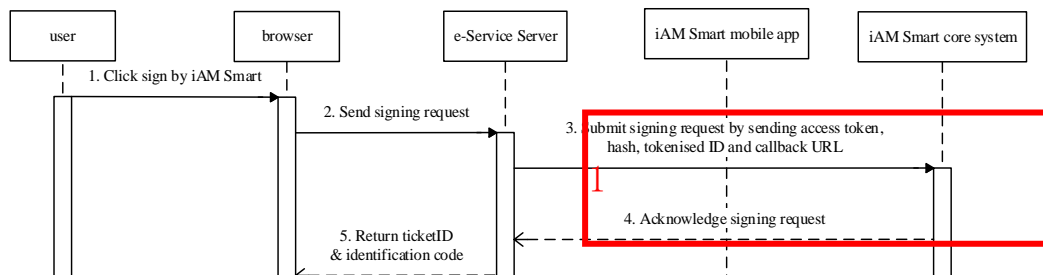
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.7.1.1, except for the following parameters:
 - Request parameter “source”: Value should be the browser's user agent value.
 - Response parameter “authByQR”: The value is “false” in this scenario.
 - Response parameter “ticketID”: It will be provided by “iAM Smart” System in this scenario. It is a 36-byte (or less) UUID number (ASCII character set).

3.7.2.2 Implementing (1) POST: Request PDF Digital Signing



Pre-conditions

- Please refer to Section 3.7.1.2 except:
 - Online Service generates the digest as the parameter “docDigest” from the pdf document to be signed using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter.

Post-conditions

- Please refer to Section 3.7.1.2 except:
 - Computing 4-digit identification code with “docDigest” instead of “hashCode”.

Error conditions

- Please refer to Section 3.7.1.2.

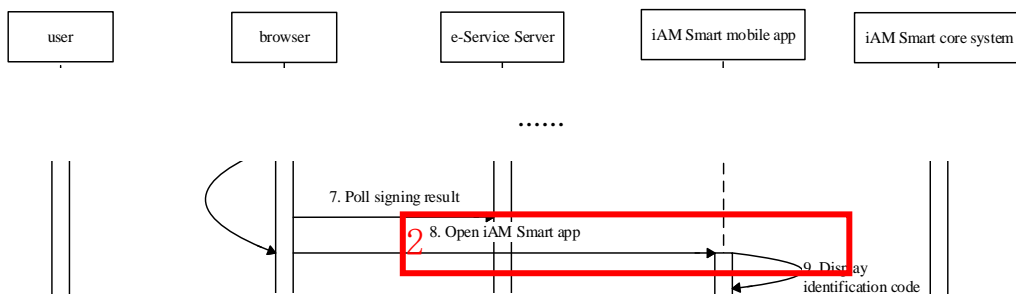
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.7.2.1, except “docDigest” request parameter is applied instead of “hashCode”.

3.7.2.3 Implementing (2) URL Scheme: Open “iAM Smart” Mobile App for Getting Context



Pre-conditions

- Online Service Website and “iAM Smart” Mobile App are in the same device.
- Online Service has the “ticketID” provided by “iAM Smart” System for the digital signing request;
- Other parameters of this API are not used in this scenario.

Post-conditions

- “iAM Smart” Mobile App will be launched.
- Online Service Website should keep synchronising with Online Service server for the callback response of the digital signing request from “iAM Smart” System.

Error conditions

- Nil

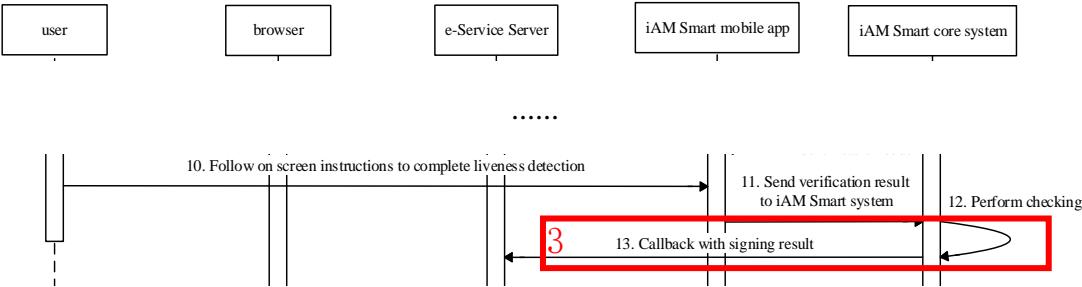
Request Parameters

- Please refer to “iAM Smart” API Specification.

Notes

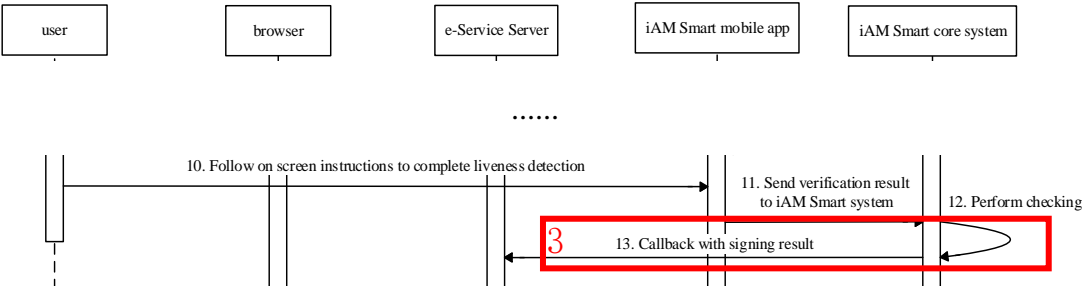
- Set <Context> as “hash-sign” or “pdf-sign” in URL Scheme.

3.7.2.4 Implementing (3) POST: Callback to Receive Digital Signing Result



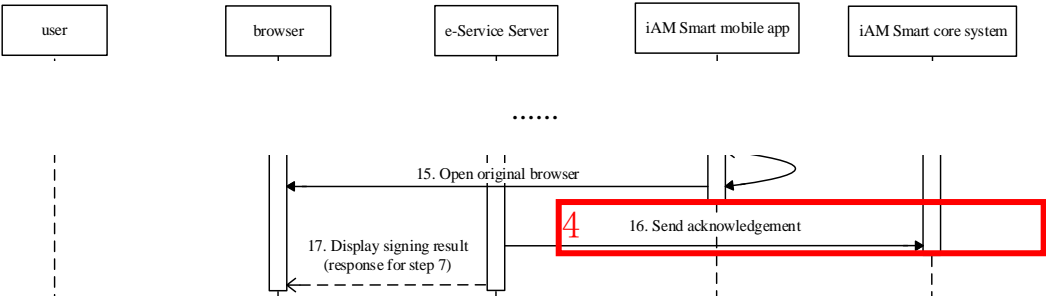
Please refer to Section 3.7.1.3.

3.7.2.5 Implementing (3) POST: Callback to Receive PDF Digital Signing Result



Please refer to Section 3.7.1.4.

3.7.2.6 Implementing (4) POST: Online Service Acknowledges Digital Signing Result



Please refer to Section 3.7.1.5.

3.7.3 Scenario 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.

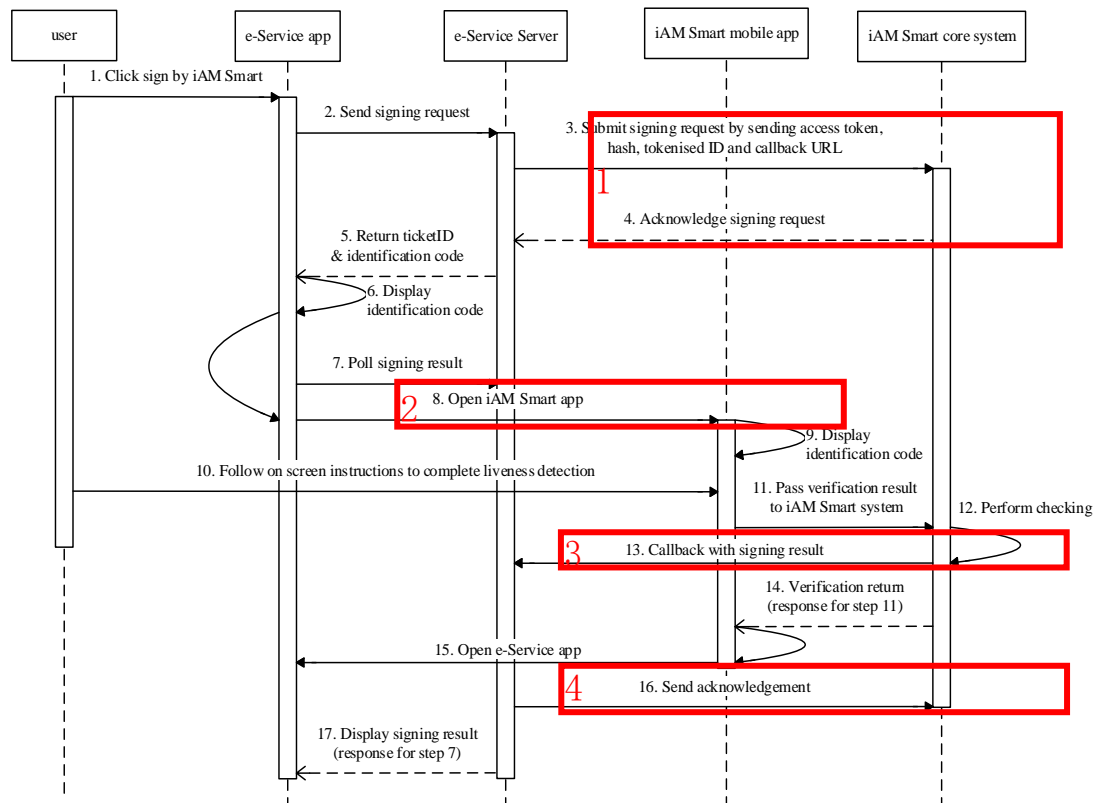


Figure-16 Digital Signing (Online Service App in Same Device)

- Step 1. User clicks the “Sign by iAM Smart” button in Online Service App;
- Step 2-3. Online Service App initiates digital signing request to Online Service server with “App_Scheme” or “App_Link” (use as the value of request parameter “source”). Online Service initiates digital signing request to invoke “iAM Smart” API with the “businessID” of this request, accessToken, Tokenised ID, Document Hash / PDF digest, HKICHash, signature algorithm (only for “Request Digital Signing”), etc. The request parameter “source” will be “App_Scheme” or “App_Link” in this scenario. API data encryption is required;

“iAM Smart” API (POST: Request Digital Signing)

“iAM Smart” API (POST: Request PDF Digital Signing)

- Step 4. “iAM Smart” System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the digital signing request to Online Service server by returning POST response with parameters “authByQR” (set to “false”) and “ticketID”;
- Step 5-6. Online Service server uses the Document Hash / PDF digest and hash of Tokenised ID to calculate a 4-digit identification code. Online Service prepares and sends necessary parameters such as “authByQR”, “ticketID”, 4-digit identification code, etc. to Online Service App. Online Service App displays the instruction with the 4-digit identification Code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;
- Step 7. Online Service App should keep synchronising with Online Service Server for digital signing processing result (e.g., polling);
- Step 8. Online Service App invokes “iAM Smart” Mobile App using URL Scheme with “ticketID” (set <Context> as “hash-sign” or “pdf-sign” in URL Scheme). Other parameters of this API are not used in this scenario;

“iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Getting Context)

- Step 9. “iAM Smart” user logs in “iAM Smart” Mobile App, reviews the digital signing authorisation request (e.g., continue or reject the request) and verifies the 4-digit identification code with Online Service App;
- Step 10-12. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 13. “iAM Smart” System invokes Online Service callback API to return the result with “businessID” of the digital signing request to Online Service Server. API data decryption is required;

Online Service Callback API (POST: Callback to Receive Digital Signing Result)

Online Service Callback API (POST: Callback to Receive PDF Digital Signing Result)

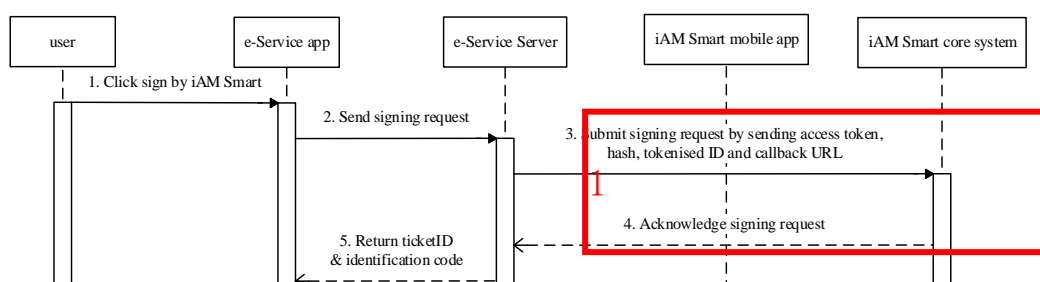
Step 14-15. “iAM Smart” System instructs “iAM Smart” Mobile App to invoke the Online Service App using URL Scheme, or Universal Link/App Link (“iAM Smart” System queries the information registered at Online Service registration depending on the “source” submitted in Step 3);

Step 16. Online Service Server completes the document digital signing process, verify the result and acknowledges “iAM Smart” System the digital signing result using “iAM Smart” API with the same “businessID” of the digital signing request. API data encryption is required;

“iAM Smart” API (POST: Online Service Acknowledges Digital Signing Result)

Step 17. Online Service Server matches the result using the “businessID” and shows the digital signing result in corresponding Online Service App.

3.7.3.1 Implementing (1) POST: Request Digital Signing



Pre-conditions

- Please refer to Section 3.7.2.1 except:
 - Online Service App and “iAM Smart” Mobile App are in the same device in this scenario.

Post-conditions

- Please refer to Section 3.7.2.1.

Error conditions

- Please refer to Section 3.7.2.1.

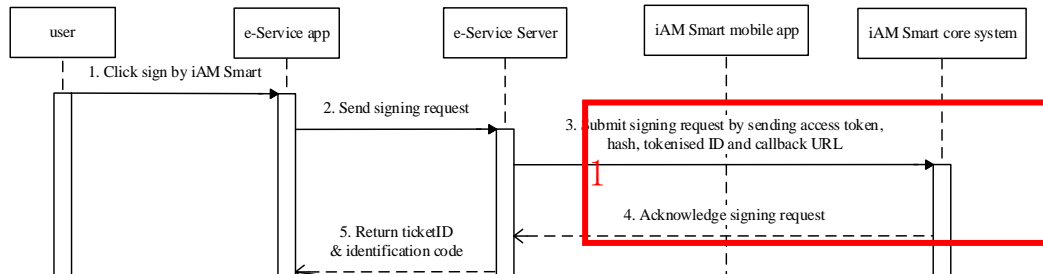
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.7.2.1, except for the following parameter:
 - Request parameter “source”: Value should be “App_Scheme” (for the Online Service App URL scheme), or “App_Link” (for the Universal Link/App Link).

3.7.3.2 Implementing (1) POST: Request PDF Digital Signing



Pre-conditions

- Please refer to Section 3.7.2.2 except:
 - Online Service generates the digest as the parameter “docDigest” from the pdf document to be signed using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter.

Post-conditions

- Please refer to Section 3.7.2.2 except:
 - Computing 4-digit identification code with “docDigest” instead of “hashCode”.

Error conditions

- Please refer to Section 3.7.2.2.

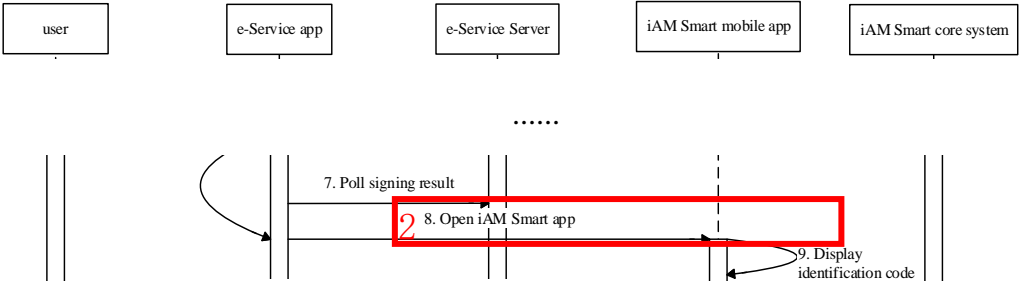
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

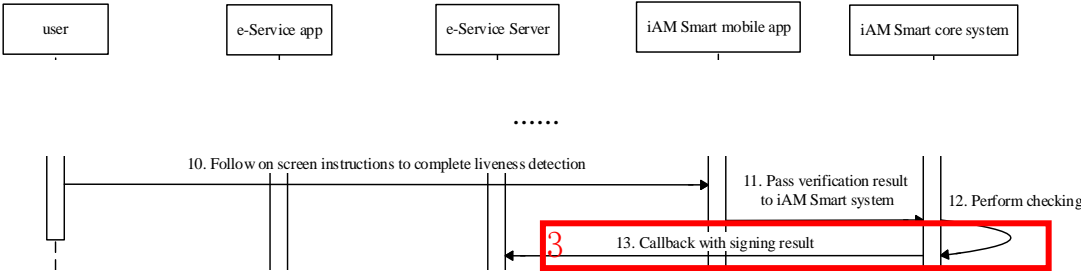
- Please refer to Section 3.7.2.2, except “docDigest” request parameter is applied instead of “hashCode”.

3.7.3.3 Implementing (2) URL Scheme: Open “iAM Smart” Mobile App for Getting Context



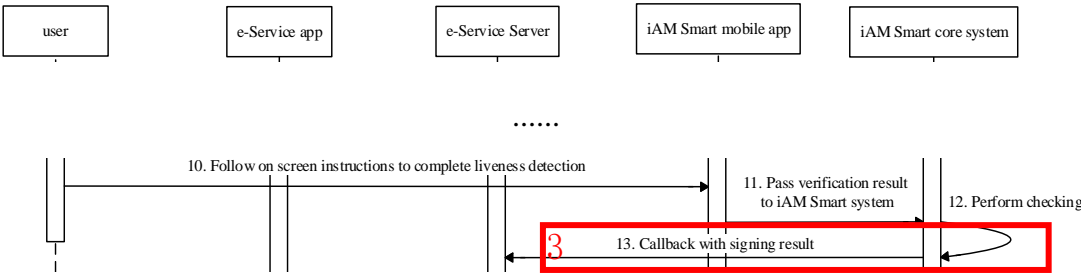
Please refer to Section 3.7.2.3.

3.7.3.4 Implementing (3) POST: Callback to Receive Digital Signing Result



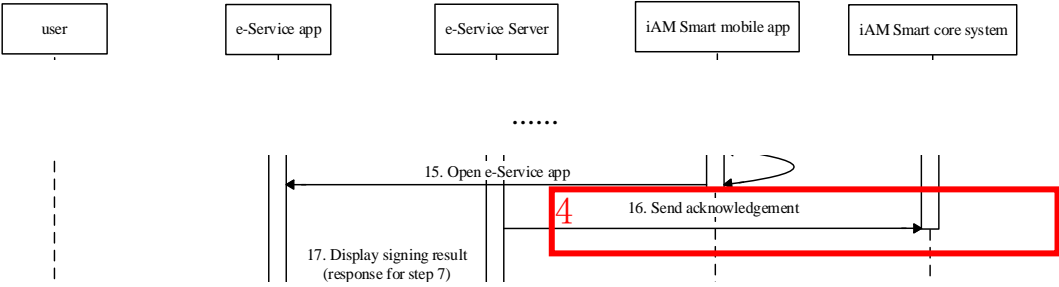
Please refer to Section 3.7.1.3.

3.7.3.5 Implementing (3) POST: Callback to Receive PDF Digital Signing Result



Please refer to Section 3.7.1.4.

3.7.3.6 Implementing (4) POST: Online Service Acknowledges Digital Signing Result



Please refer to Section 3.7.1.5.

3.8 WORKFLOWS FOR DIGITAL SIGNING WITHOUT SERVICE LOGIN (AKA ANONYMOUS DIGITAL SIGNING)

The following process includes anonymous hash digital signing and anonymous pdf digital signing.

3.8.1 Scenario 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.

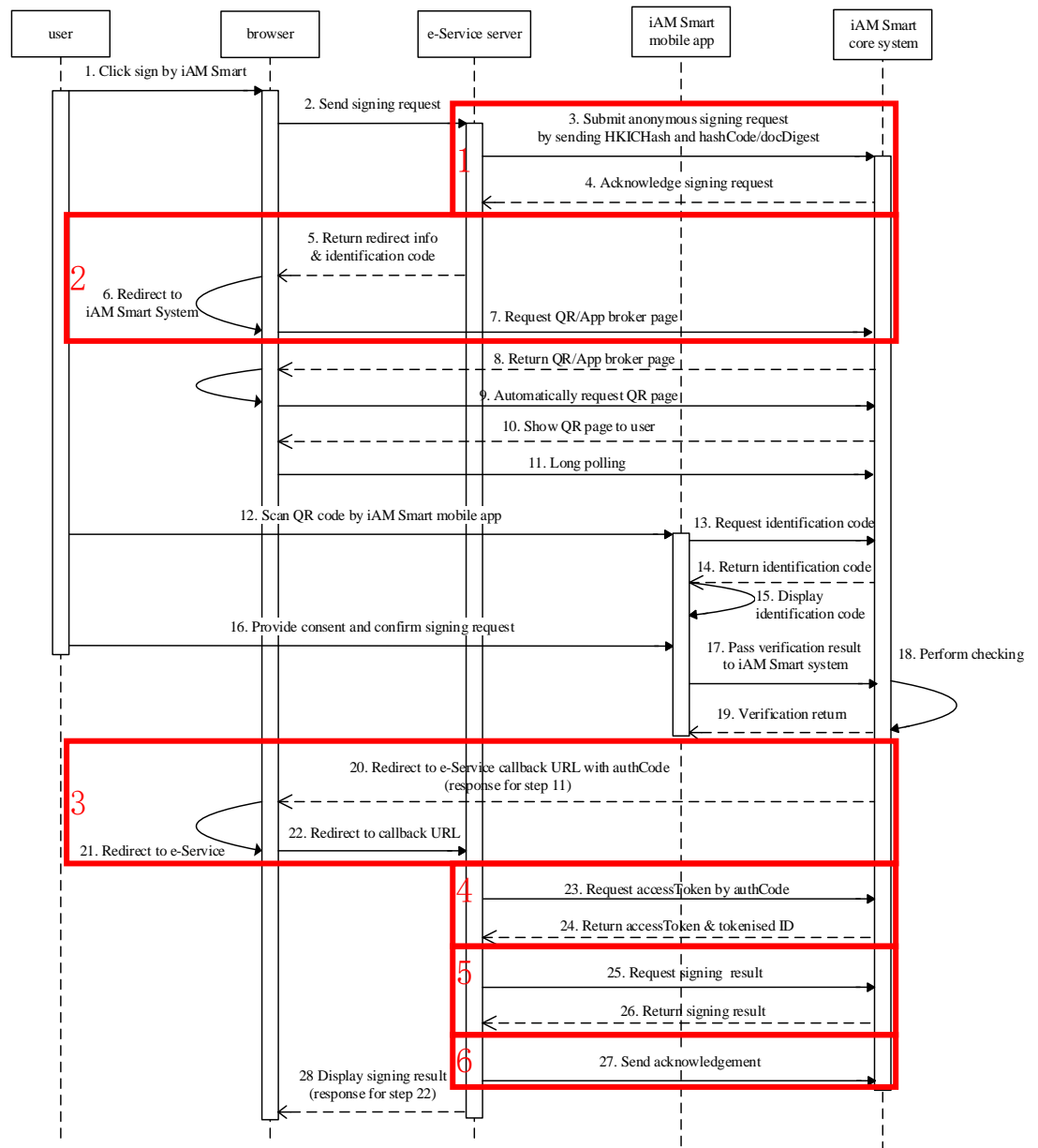


Figure-17 Anonymous Digital Signing (Online Service Website in Different Device)

- Step 1. After entering HKIC number, the user clicks the “Anonymous hash digital signing” (or “Anonymous pdf digital signing”) button in Online Service Website;

- Step 2. Online Service Website initiates anonymous digital signing request to Online Service Server with browser's user agent name (use as value for request parameter “source”);

- Step 3. Online Service Server initiates an anonymous digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, Document Hash / PDF digest, HKICHash, signature algorithm (only for "*Request Anonymous Digital Signing*"), etc. API encryption is required;

“iAM Smart” API (POST: Request Anonymous Digital Signing)

“iAM Smart” API (POST: Request Anonymous PDF Digital Signing)

- Step 4. “iAM Smart” System verifies and confirms receipt of the anonymous digital signing request to Online Service Server by returning POST response with parameter “ticketID”;

- Step 5-7. Online Service Server uses the Document Hash / PDF digest, HKICHash and hash of clientID to calculate a 4-digit identification code and instructs Online Service Website to display the instructions with the 4-digit identification code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;

Online Service Server prepares the request parameters such as “clientID”, “redirectURI”, “source” (set as browser’s user agent value), “scope”, “ticketID”, etc. and constructs the GET request to invoke the “iAM Smart” API using browser redirection;

“iAM Smart” API (GET: Request QR page)

- Step 8-10. “iAM Smart” System returns the broker page (if Online Service has set “brokerPage” to “true”) and after the broker page fails to find the “iAM Smart” Mobile App, it will request QR page to be displayed in browser. If Online Service does not request for

broker page (i.e., no broker page will be returned), the browser will directly display QR Code page;

- Step 11. QR Code page of “iAM Smart” System polls “iAM Smart” System for the QR Code status;
- Step 12. “iAM Smart” user logs in “iAM Smart” Mobile App to scan the QR Code;
- Step 13-15. “iAM Smart” Mobile App requests and displays 4-digit identification code from “iAM Smart” System. “iAM Smart” user reviews the digital signing authorisation request (e.g., continue or reject the request) and verifies the 4-digit identification code with Online Service Website;
- Step 16-17. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 18-19. “iAM Smart” System verifies the validity of QR code and other necessary information, and return the authorisation result to “iAM Smart” Mobile App;
- Step 20-22. QR Code page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e., response for the polling in Step 11) which includes authCode and businessID or any error code (e.g., “iAM Smart” user rejects the request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

- Step 23-24. When Online Service Server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

“iAM Smart” API (POST: Request accessToken & Tokenised ID)

- Step 25-26. Online Service Server invokes the “iAM Smart” API with the accessToken and openID to obtain the result of anonymous digital signing. API data encryption is required;

“iAM Smart” API (POST: Obtain Anonymous Digital Signing Result)

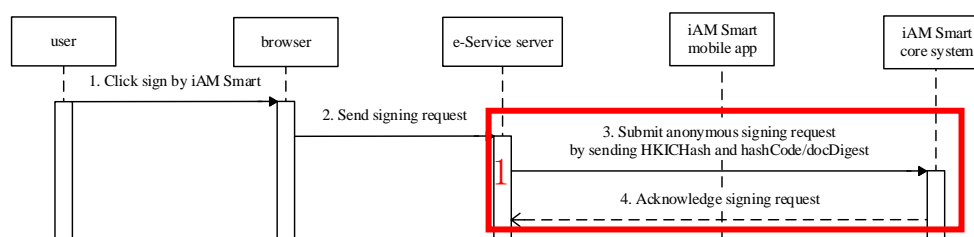
“iAM Smart” API (POST: Obtain Anonymous PDF Digital Signing Result)

- Step 27. Online Service Server completes the document digital signing process, verify the result and acknowledges “iAM Smart” System the digital signing result using “iAM Smart” API with the same “businessID” of the digital signing request. API data encryption is required;

“iAM Smart” API (POST: Online Service Acknowledges Digital Signing Result)

- Step 28. Online Service Server matches the result using the “businessID” and shows the digital signing result in corresponding Online Service Website.

3.8.1.1 Implementing (1) POST: Request Anonymous Digital Signing



Pre-conditions

- Please refer to Section 3.7.1.1 except:
 - No request parameter “accessToken” exists.
 - Online Service generates a “businessID” for this request and uses this identifier to match the authCode returned from “iAM Smart” System.
 - Online Service should convert the HKIC number (no check digit is needed) into hash value using SHA256.

Post-conditions

- Please refer to Section 3.7.1.1 except:
 - No response parameter “authByQR” exists.
 - Online Service calculates a 4-digit identification code using:
Document Hash + HKICHash + clientID hash value (calculated with SHA-512)

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D70002	Failed to request signing	Inform user the “iAM Smart” digital signing request is failed and retry later

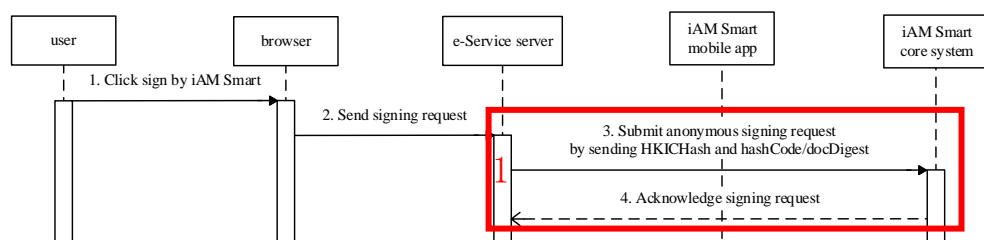
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.7.1.1, except for the following parameters:
 - No request parameter “accessToken”, “openID”, “source”, “redirectURI” and “state”.
 - No response parameter “authByQR”.
 - Response parameter “ticketID”: It is returned from “iAM Smart” System for “iAM Smart” APIs for anonymous request. It will be used for invoking subsequent “iAM Smart” APIs “Request QR Page” and “Open “iAM Smart” Mobile App for Getting Context” depending on the scenario. It is a 36-byte (or less) UUID number (ASCII character set).

3.8.1.2 Implementing (1) POST: Request Anonymous Anonymous PDF Digital Signing



Pre-conditions

- Please refer to Section 3.7.3.2.

Post-conditions

- Please refer to Section 3.7.3.2 except:
 - Online Service calculates a 4-digit identification code using:

docDigest of PDF document + HKICHash + clientID hash value
(calculated with SHA-512)

Error conditions

- Please refer to Section 3.7.3.2.

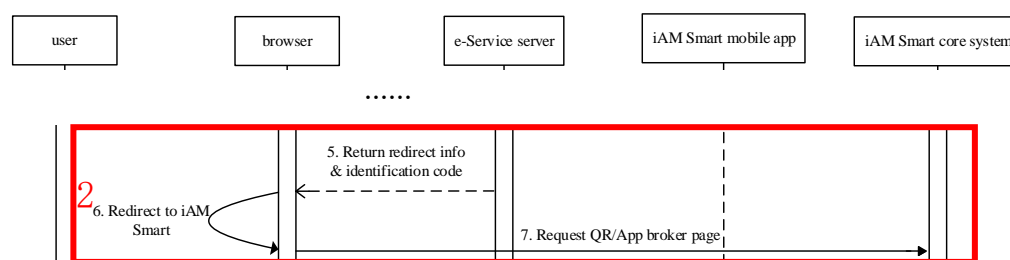
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.7.3.2.

3.8.1.3 Implementing (2) GET: Request QR Page



Pre-conditions

- Please refer to Section 3.4.1.1 except:
 - Online Service has the “ticketID” provided by “iAM Smart” System for the anonymous request in step 4.

Post-conditions

- Please refer to Section 3.4.1.1.

Error conditions

- Please refer to Section 3.4.1.1.

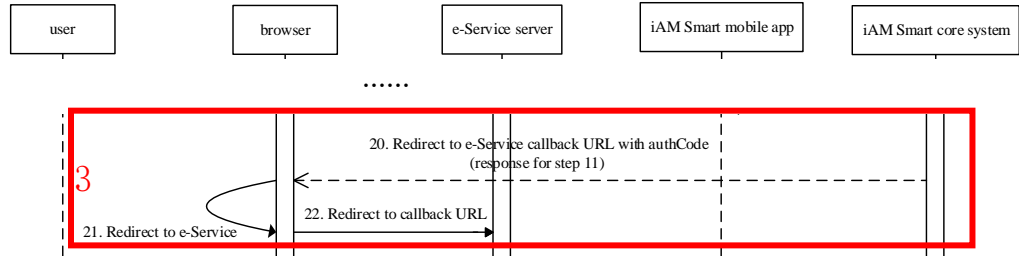
Request Parameters

- Please refer to Section 3.4.1.1.

Notes

- Please refer to Section 3.4.1.1 except the following parameter:
 - Request parameter “ticketID”: Value provided by “iAM Smart” System for the anonymous request.

3.8.1.4 Implementing (3) GET: Callback with authCode to Online Service Server



Pre-conditions

- Please refer to Section 3.4.1.2.

Post-conditions

- Please refer to Section 3.4.1.2 except:
 - Online Service Server should match the callback result with corresponding Online Service client terminal using the “businessID”.

Error conditions

- This Online Service callback API is a HTTP GET request. If parameter “error_code” is returned, it means the request is failed.

Error Code	Error Description	Suggested Action
error_code - D70000	User cancelled signing request	Inform user the digital signing request is cancelled
error_code - D70001	User rejected signing request	Inform user the digital signing request is rejected
error_code - D70002	Failed to request signing	Inform user the digital signing request is failed and retry later
error_code - D70004	User not allowed to sign	Inform user that “iAM Smart” digital signing is not allowed for default “iAM Smart” and suggest user to upgrade to “iAM Smart” with digital signing function

Error Code	Error Description	Suggested Action
error_code - D70005	Inconsistent HKIC number	Hash of HKIC number provided is not matched with “iAM Smart” user. Check hash of HKIC number or inform user to authorise the digital signing using the correct “iAM Smart” account

The error_code D70003 (signing request timeout) does not appear in this scenario. For the QR code timeout or request confirmation timeout in the “iAM Smart” Mobile App, message will be prompted in the corresponding user interface.

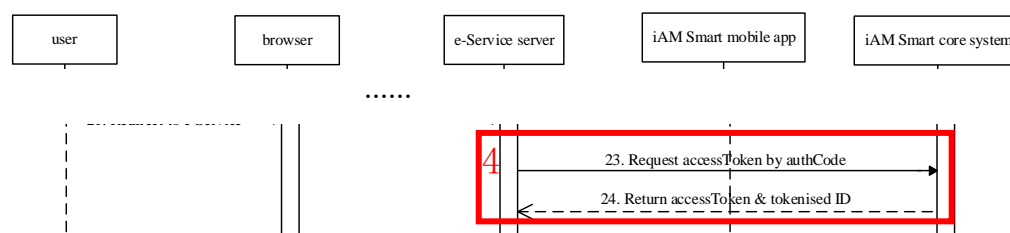
Callback Parameters

- Please refer to Section 3.4.1.2.

Notes

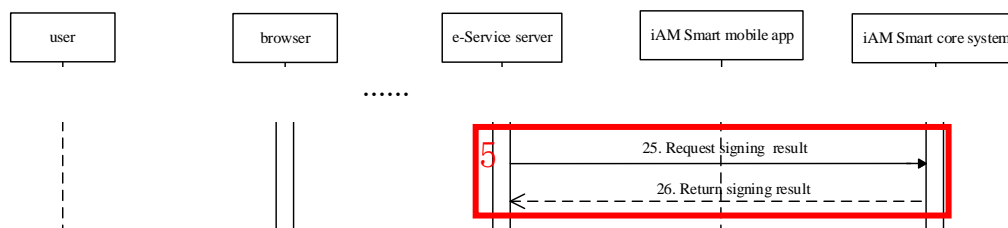
- Please refer to Section 3.4.1.2 except the following parameter:
 - Callback parameter “businessID”: Value returned by “iAM Smart” System should be the same one as Online Service submitted in the anonymous request.

3.8.1.5 Implementing (4) POST: Request accessToken & Tokenised ID



Please refer to Section 3.4.1.3

3.8.1.6 Implementing (5) POST: Obtain Anonymous Digital Signing Result



Pre-conditions

- Online Service must possess a valid accessToken of the “iAM Smart” user with authorisation scope “Signing authorisation”.
- API data encryption is required.

Post-conditions

- API data decryption is required.
- Online Service Server completes the document digital signing process and acknowledges “iAM Smart” System the digital signing result using “iAM Smart” API with the same “businessID” of the digital signing request.
- Online Service should discard the accessToken as it can only be used once.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D70002	Failed to request signing	Inform user the “iAM Smart” digital signing request is failed and retry later
code - D70003	Signing request timeout	Inform user the “iAM Smart” digital signing request is timeout and provide way for user to retry
code - D70004	User not allowed to sign	Inform user that “iAM Smart” digital signing is not allowed for default “iAM Smart” and suggest user to upgrade to “iAM Smart” with digital signing function

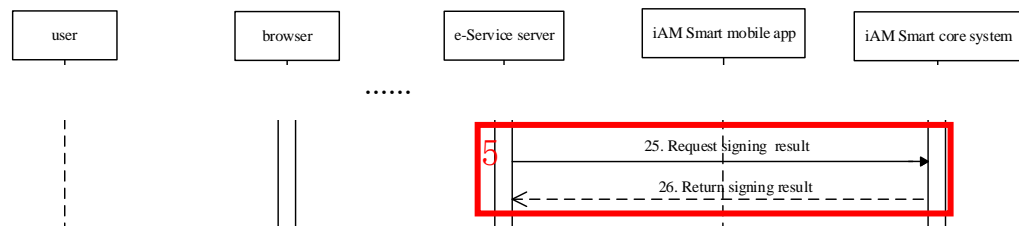
.Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.7.1.3 except:
 - Response parameters “businessID” and “state” do not exist.

3.8.1.7 Implementing (5) POST: Obtain Anonymous PDF Digital Signing Result



Pre-conditions

- Please refer to Section 3.6.4.5

Post-conditions

- Please refer to Section 3.6.4.5.

Error conditions

- Please refer to Section 3.6.4.5.

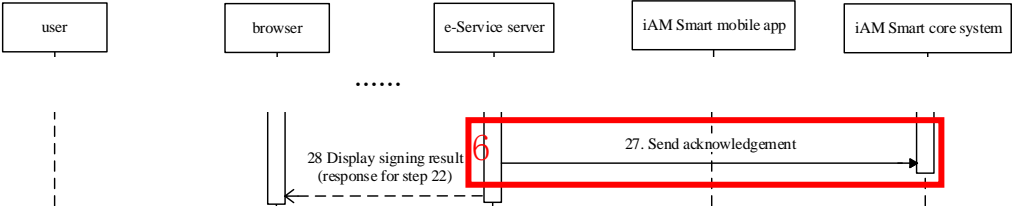
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.7.1.4 except:
 - Response parameter “businessID” and “state” do not exist.

3.8.1.8 Implementing (6) POST: Online Service Acknowledges Digital Signing Result



Please refer to Section 3.7.1.5.

3.8.2 Scenario 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.

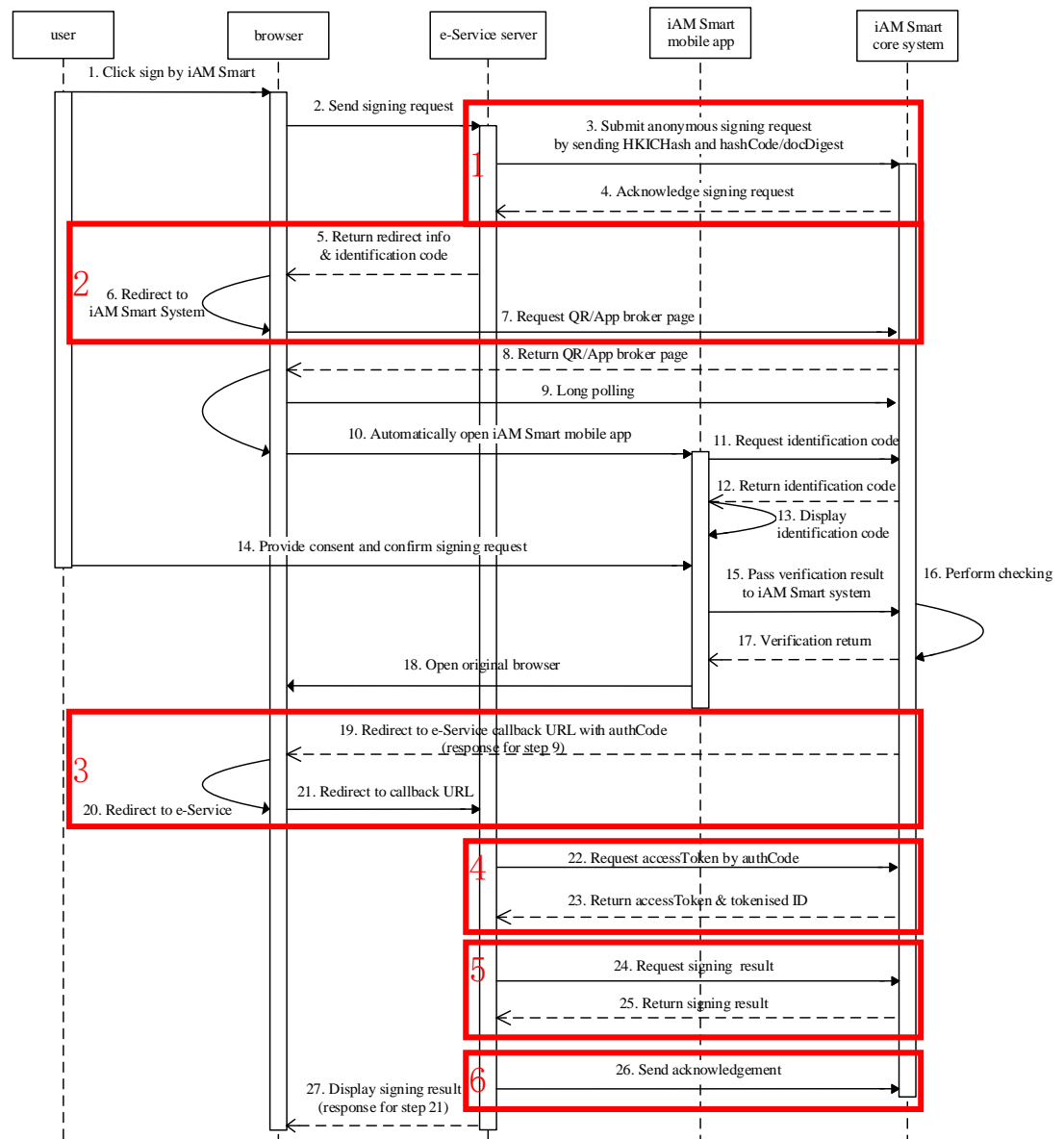


Figure-18 Anonymous Digital Signing (Online Service Website in Same Device)

- Step 1. After entering HKIC number, the user clicks the “Anonymous hash digital signing” (or “Anonymous pdf digital signing”) button in Online Service Website;
- Step 2. Online Service Website initiates anonymous digital signing request to Online Service Server with browser's user agent name (use as value for request parameter “source”);
- Step 3. Online Service Server initiates an anonymous digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, Document Hash / PDF digest, HKICHash, signature algorithm (only for “Request Anonymous Digital Signing”), etc. API data encryption is required;
- “iAM Smart” API (POST: Request Anonymous Digital Signing)*
- “iAM Smart” API (POST: Request Anonymous PDF Digital Signing)*
- Step 4. “iAM Smart” System verifies and confirms receipt of the anonymous digital signing request to Online Service Server by returning POST response with parameter “ticketID”;
- Step 5-7. Online Service Server uses the Document Hash / PDF digest, HKICHash and hash of clientID to calculate a 4-digit identification code and instructs Online Service Website to display the instructions with the 4-digit identification code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;
- Online Service Server prepares the request parameters such as “clientID”, “redirectURI”, “source” (set as browser’s user agent value), “scope”, “brokerPage” (set as “true”), “ticketID”, etc. and constructs the GET request to invoke the “iAM Smart” API using browser redirection;
- “iAM Smart” API (GET: Request QR page)*
- Step 8. “iAM Smart” System returns the broker page to the Online Service Website;
- Step 9. Broker page of “iAM Smart” System polls “iAM Smart” System for further action;

- Step 10. Broker page invokes the “iAM Smart” Mobile App in the same device automatically and sends it the relevant parameters.
- Step 11-13. “iAM Smart” user logs in “iAM Smart” Mobile App, “iAM Smart” Mobile App requests and displays 4-digit identification code from “iAM Smart” System. “iAM Smart” user reviews the digital signing authorisation request (e.g., continue or reject the request) and verifies the 4-digit identification code with Online Service Website;
- Step 14-15. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 16-18. “iAM Smart” System verifies the validity of necessary information, and return the authorisation result to “iAM Smart” Mobile App. “iAM Smart” Mobile App invokes the original Online Service browser (i.e., using the browser's user agent value submitted);
- Step 19-21. Broker page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e., response for the polling in Step 9) which includes authCode and businessID or any error code (e.g., “iAM Smart” user rejects the request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

- Step 22-23. When Online Service Server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

“iAM Smart” API (POST: Request accessToken & Tokenised ID)

- Step 24-25. Online Service Server invokes the “iAM Smart” API with the accessToken and openID to obtain the result of anonymous digital signing. API data encryption is required;

“iAM Smart” API (POST: Obtain Anonymous Digital Signing Result)

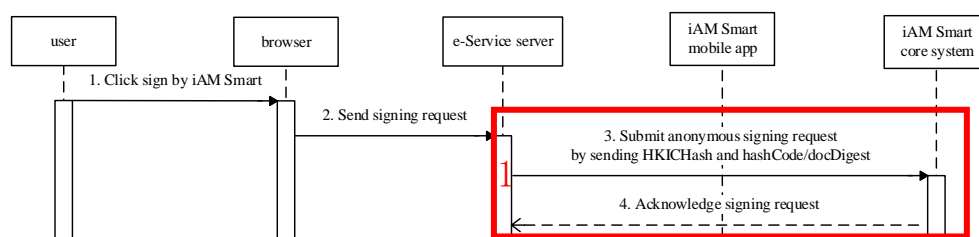
“iAM Smart” API (POST: Obtain Anonymous PDF Digital Signing Result)

- Step 26. Online Service Server completes the document digital signing process, verify the result and acknowledges “iAM Smart” System the digital signing result using “iAM Smart” API with the same “businessID” of the digital signing request. API data encryption is required;

“iAM Smart” API (POST: Online Service Acknowledges Digital Signing Result)

- Step 27. Online Service Server matches the result using the “businessID” and shows the digital signing result in corresponding Online Service Website.

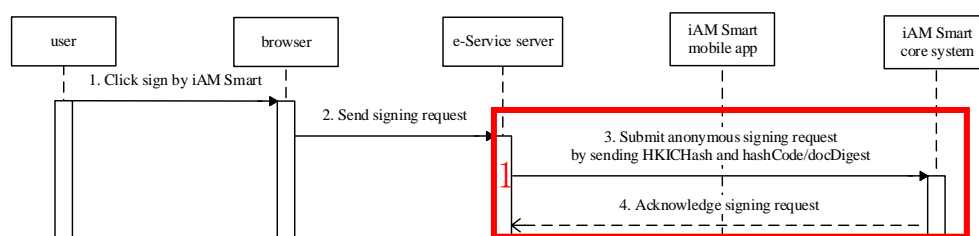
3.8.2.1 Implementing (1) POST: Request Anonymous Digital Signing



Please refer to Section 3.8.1.1 except:

- Online Service Website and “iAM Smart” Mobile App are in the same device in this scenario.

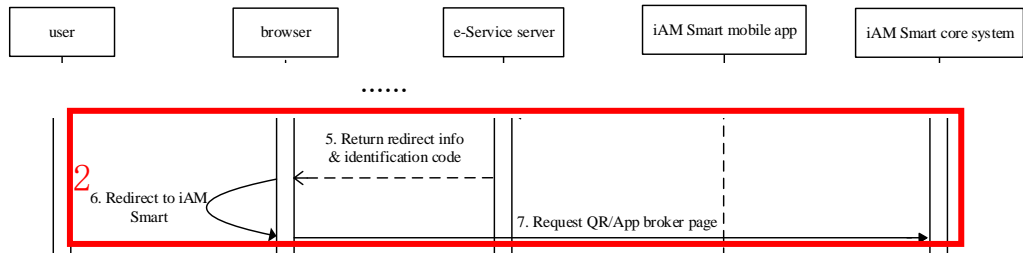
3.8.2.2 Implementing (1) POST: Request Anonymous PDF Digital Signing



Please refer to Section 3.8.1.2 except:

- Online Service Website and “iAM Smart” Mobile App are in the same device in this scenario.

3.8.2.3 Implementing (2) GET: Request QR Page



Pre-conditions

- Please refer to Section 3.4.2.1 except:
 - Online Service has the “ticketID” provided by “iAM Smart” System for the anonymous request in step 4.

Post-conditions

- Please refer to Section 3.4.2.1.

Error conditions

- Please refer to Section 3.4.2.1.

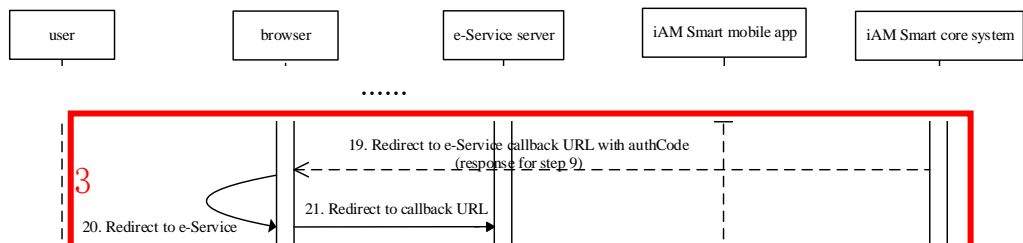
Request Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.4.2.1 except the following parameter:
 - Request parameter “ticketID”: Value provided by “iAM Smart” System for the anonymous request.

3.8.2.4 Implementing (3) GET: Callback with authCode to Online Service Server



Pre-conditions

- Please refer to Section 3.6.2.3.

Post-conditions

- Please refer to Section 3.6.2.3.

Error conditions

- Please refer to Section 3.8.1.4.

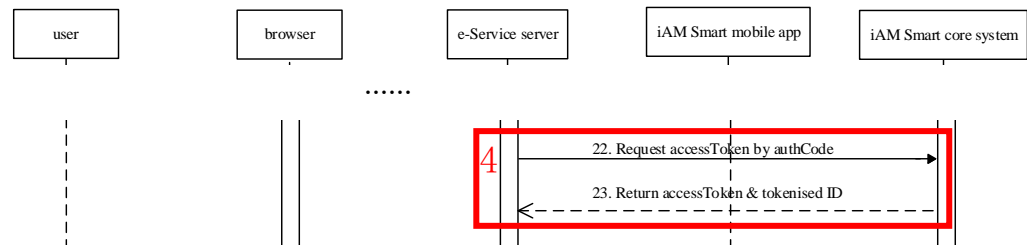
Callback Parameters

- Please refer to Section 3.6.2.3.

Notes

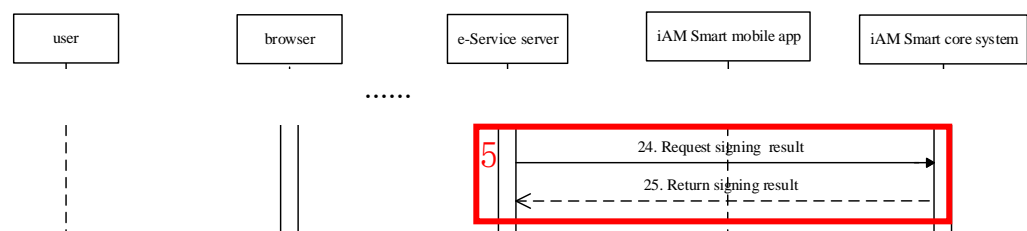
- Please refer to Section 3.6.2.3.

3.8.2.5 Implementing (4) POST: Request accessToken & Tokenised ID



Please refer to Section 3.6.1.4.

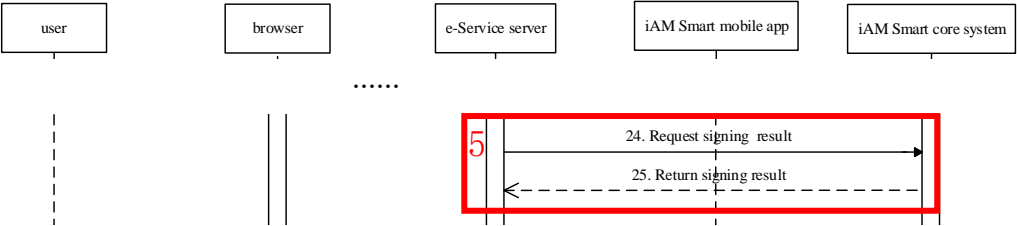
3.8.2.6 Implementing (5) POST: Obtain Anonymous Digital Signing Result



Please refer to Section 3.8.1.6.

3.8.2.7 Implementing (5) POST: Obtain Anonymous PDF Digital Signing

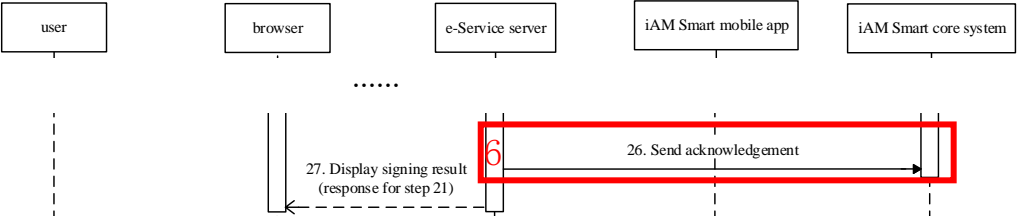
Result



Please refer to Section 3.8.1.7.

3.8.2.8 Implementing (6) POST: Online Service Acknowledges Digital Signing

Result



Please refer to Section 3.7.1.5.

3.8.3 Scenario 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.

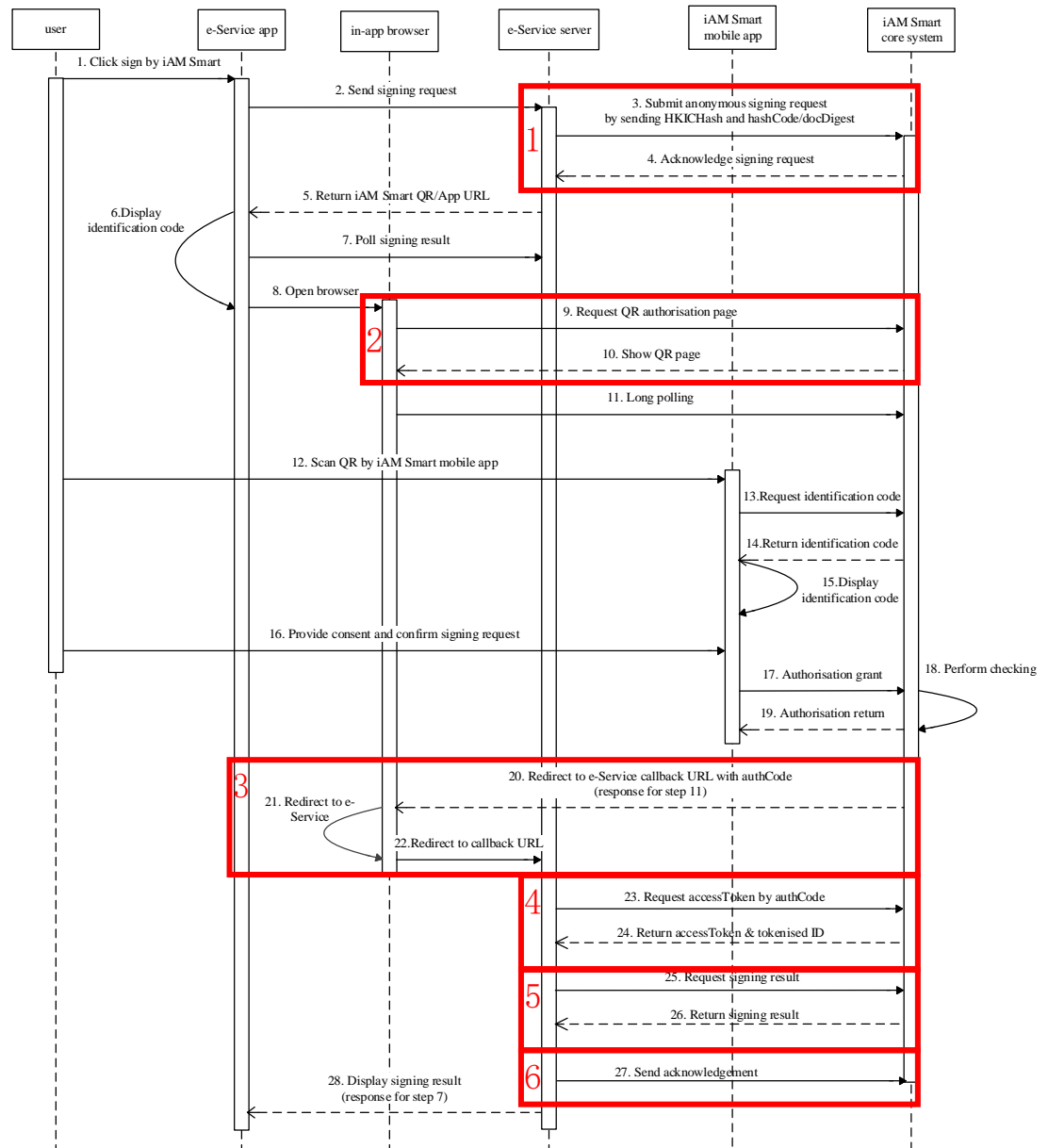


Figure-19 Anonymous Digital Signing (Online Service App in Different Device)

- Step 1. After entering HKIC number, the user clicks the “Anonymous hash digital signing” (or “Anonymous pdf digital signing”) button in Online Service App;
- Step 2. Online Service App determines there is no “iAM Smart” Mobile App in the device using program code, initiates anonymous digital signing request to Online Service Server with the in-app browser’s user agent value (use as value for request parameter “source”);
- Step 3. Online Service Server initiates an anonymous digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, Document Hash / PDF digest, HKICHash, signature algorithm (only for "*Request Anonymous Digital Signing*"), etc. API data encryption is required;
- “iAM Smart” API (POST: Request Anonymous Digital Signing)*
- “iAM Smart” API (POST: Request Anonymous PDF Digital Signing)*
- Step 4. “iAM Smart” System verifies and confirms receipt of the anonymous digital signing request to Online Service Server by returning POST response with parameter “ticketID”;
- Step 5-6. Online Service Server uses the Document Hash / PDF digest, HKICHash and hash of clientID to calculate a 4-digit identification code and instructs Online Service App to display the instructions with the 4-digit identification code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;
- Online Service Server prepares necessary parameters such as “clientID”, “redirectURI”, “scope”, “source” (set as in-app browser’s user agent value), “brokerPage” (set as “false”), “ticketID”, etc. and constructs the GET request to invoke the “iAM Smart” API by Online Service App;
- Step 7. Online Service App polls Online Service Server for the digital signing result from “iAM Smart” System;
- Step 8. Online Service App invokes in-app browser (Safari for iOS, Chrome for Android) to submit the GET request;

Step 9-10. Browser opens the GET request to invoke the “iAM Smart” API and QR Code page will be shown;

“iAM Smart” API (GET: Request QR page)

Step 11. QR Code page of “iAM Smart” System polls “iAM Smart” System for the QR Code status;

Step 12. “iAM Smart” user logs in “iAM Smart” Mobile App to scan the QR Code;

Step 13-15. “iAM Smart” Mobile App requests and displays 4-digit identification code from “iAM Smart” System. “iAM Smart” user reviews the digital signing authorisation request (e.g., continue or reject the request) and verifies the 4-digit identification code with Online Service Website;

Step 16-17. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;

Step 18-19. “iAM Smart” System verifies the validity of QR code and other necessary information, and return the authorisation result to “iAM Smart” Mobile App;

Step 20-22. QR Code page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e., response for the polling in Step 11) which includes authCode and businessID or any error code (e.g., “iAM Smart” user rejects the request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

Step 23-24. When Online Service Server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;

“iAM Smart” API (POST: Request accessToken & Tokenised ID)

Step 25-26. Online Service Server invokes the “iAM Smart” API with the accessToken and openID to obtain the result of anonymous digital signing. API data encryption is required;

“iAM Smart” API (POST: Obtain Anonymous Digital Signing Result)

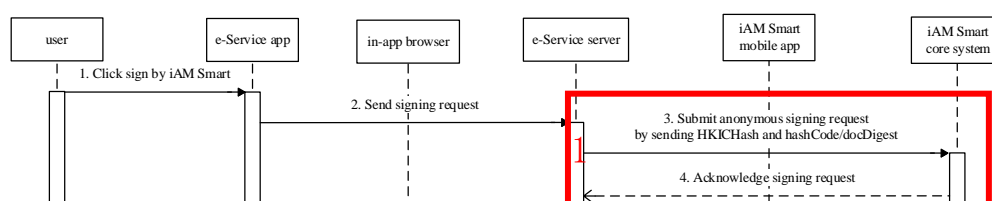
“iAM Smart” API (POST: Obtain Anonymous PDF Digital Signing Result)

Step 27. Online Service Server completes the document digital signing process, verify the result and acknowledges “iAM Smart” System the digital signing result using “iAM Smart” API with the same “businessID” of the digital signing request. API data encryption is required;

“iAM Smart” API (POST: Online Service Acknowledges Digital Signing Result)

Step 28. Online Service Server matches the result using the “businessID” and shows the digital signing result in corresponding Online Service App (i.e., response for the polling in Step 7).

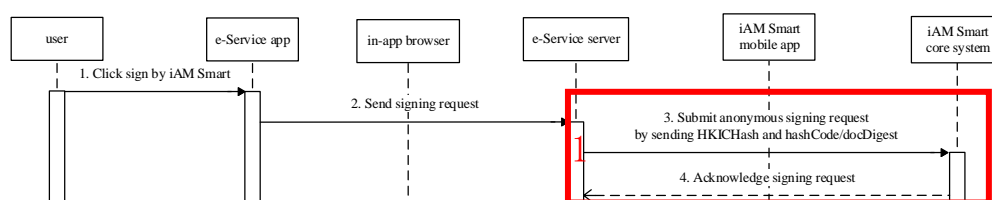
3.8.3.1 Implementing (1) POST: Request Anonymous Digital Signing



Please refer to Section 3.8.1.1 except:

- Online Service should determine the “iAM Smart” Mobile App is not in the same device of Online Service App using program code.

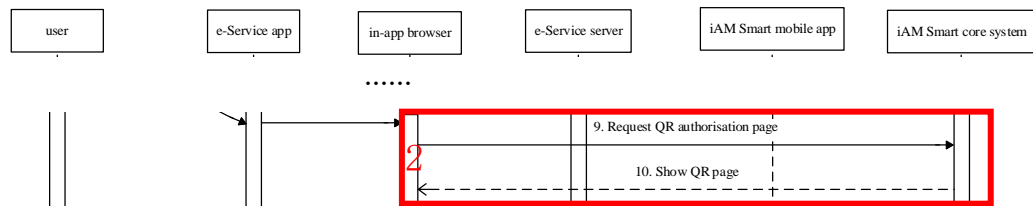
3.8.3.2 Implementing (1) POST: Request Anonymous PDF Digital Signing



Please refer to Section 3.8.1.2 except:

- Online Service should determine the “iAM Smart” Mobile App is not in the same device of Online Service App using program code.

3.8.3.3 Implementing (2) GET: Request QR Page



Pre-conditions

- Please refer to Section 3.4.3.1 except:
 - Online Service has the “ticketID” provided by “iAM Smart” System for the anonymous request in step 4.

Post-conditions

- Please refer to Section 3.4.3.1.

Error conditions

- Please refer to Section 3.4.3.1.

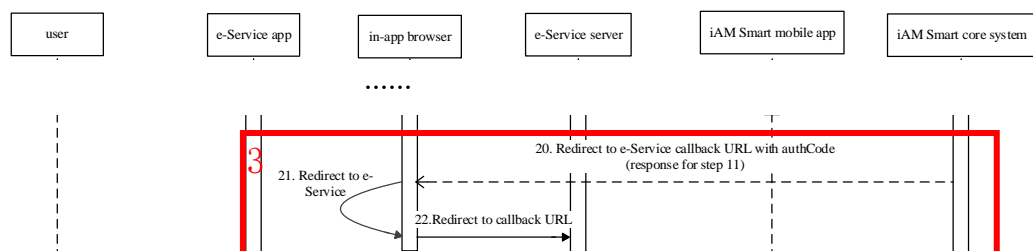
Request Parameters

- Please refer to “iAM Smart” API Specification.

Notes

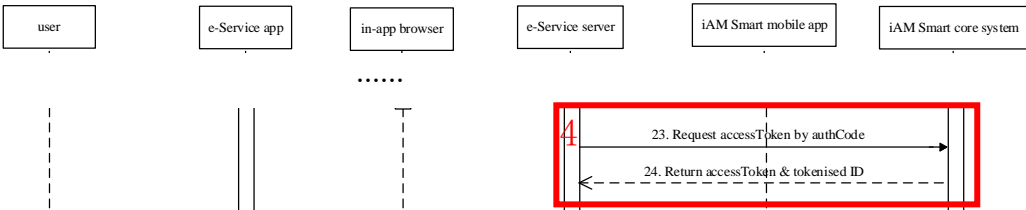
- Please refer to Section 3.4.3.1 except the following parameter:
 - Request parameter “ticketID”: Value provided by “iAM Smart” System for the anonymous request.

3.8.3.4 Implementing (3) GET: Callback with authCode to Online Service Server



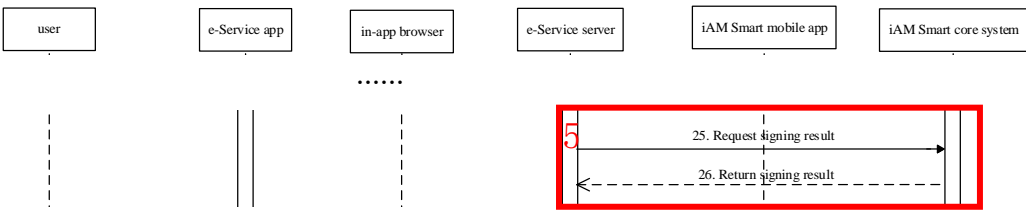
Please refer to Section 3.8.1.4.

3.8.3.5 Implementing (4) POST: Request accessToken & Tokenised ID



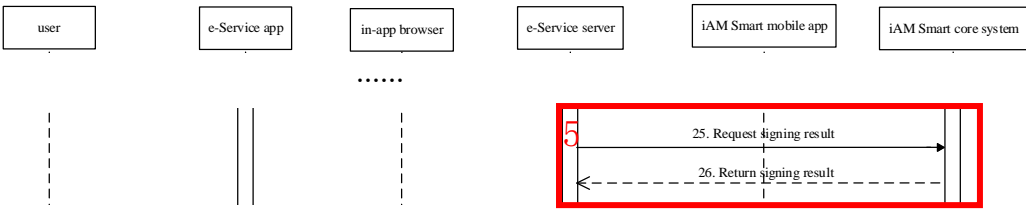
Please refer to Section 3.6.1.4.

3.8.3.6 Implementing (5) POST: Obtain Anonymous Digital Signing Result



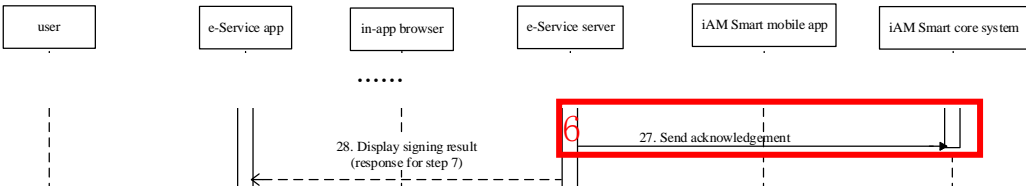
Please refer to Section 3.8.1.6.

3.8.3.7 Implementing (5) POST: Obtain Anonymous PDF Digital Signing Result



Please refer to Section 3.8.1.7.

3.8.3.8 Implementing (6) POST: Online Service Acknowledges Digital Signing Result



Please refer to Section 3.7.1.5.

3.8.4 Scenario 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.

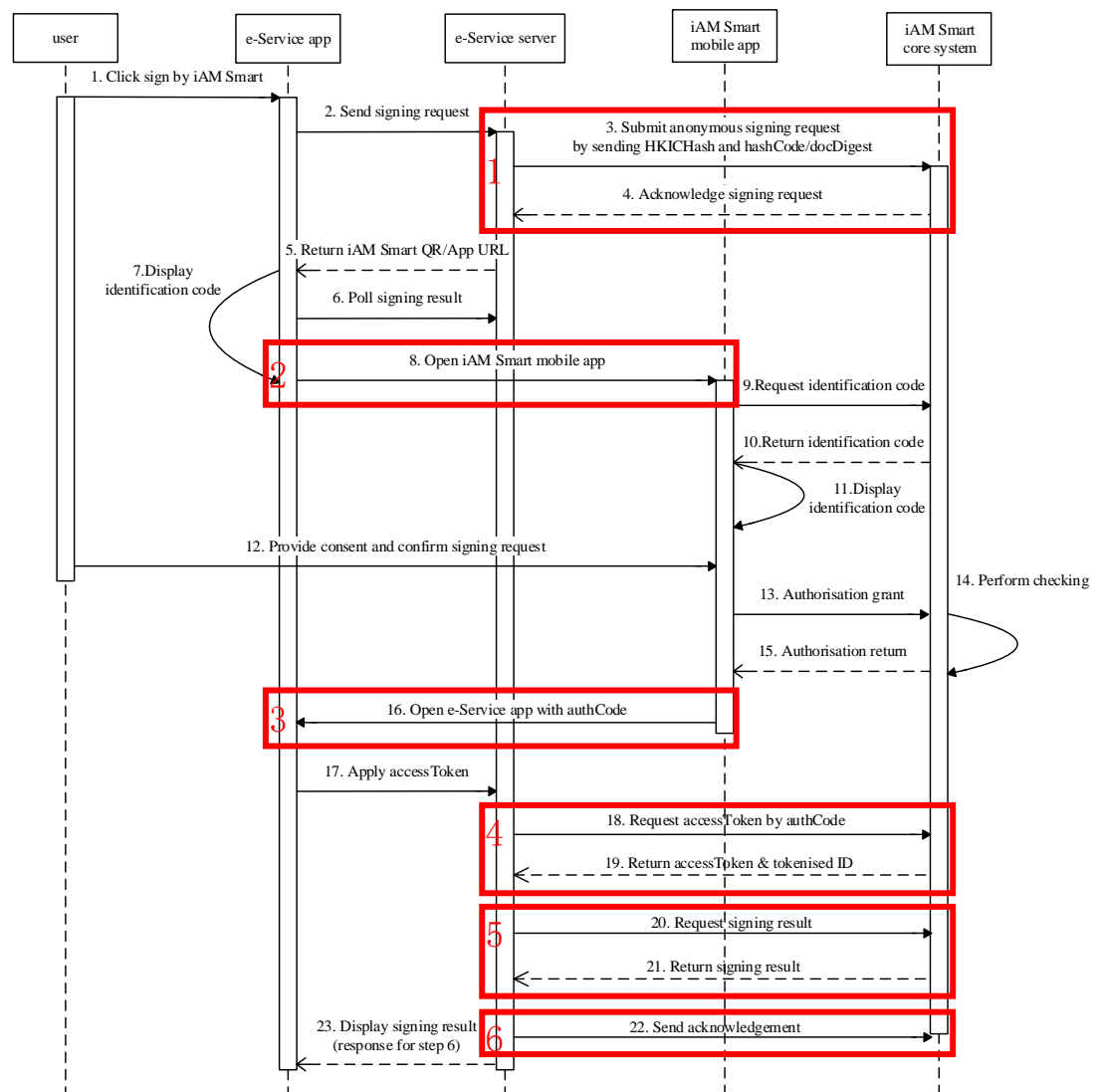


Figure-20 Anonymous Digital Signing (Online Service App in Same Device)

Step 1. After entering HKIC number, the user clicks the “Anonymous hash digital signing” (or “Anonymous pdf digital signing”) button in Online Service App;

- Step 2. Online Service App determines “iAM Smart” Mobile App is installed in the device using program code, initiates anonymous digital signing request to Online Service Server with “App_Scheme” or “App_Link” (use as value for request parameter “source”);
- Step 3. Online Service Server initiates an anonymous digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, Document Hash / PDF digest, HKICHash, signature algorithm (only for "Request Anonymous Digital Signing"), etc. API data encryption is required;
- “iAM Smart” API (POST: Request Anonymous Digital Signing)*
- “iAM Smart” API (POST: Request Anonymous PDF Digital Signing)*
- Step 4. “iAM Smart” System verifies and confirms receipt of the anonymous digital signing request to Online Service Server by returning POST response with parameter “ticketID”;
- Step 5&7. Online Service Server uses the Document Hash / PDF digest, HKICHash and hash of clientID to calculate a 4-digit identification code and instructs Online Service App to display the instructions with the 4-digit identification code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;
- Online Service Server prepares necessary parameters such as “clientID”, “redirectURI” (set as Online Service App URL Scheme or Universal Link/App Link depending on “source” parameter), “scope”, “source” (set as “App_Scheme” or “App_Link”), “ticketID”, etc. and constructs the URL Scheme to invoke “iAM Smart” Mobile App by Online Service App;
- Step 6. Online Service App polls Online Service Server for the digital signing result from “iAM Smart” System;
- Step 8. Online Service App invokes “iAM Smart” Mobile App using URL Scheme with request parameters (set <Context> as “anon_hash-sign” or “anon_pdf-sign” in URL Scheme);
- “iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Getting Context)*

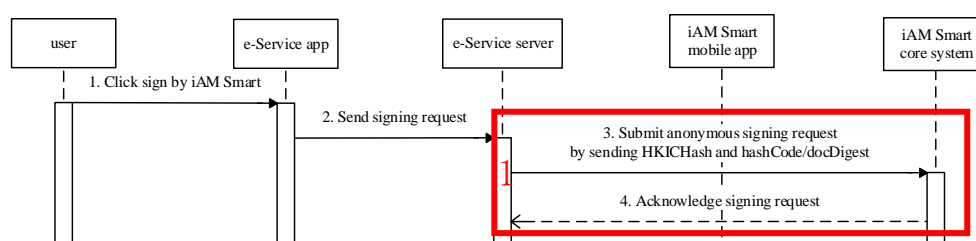
- Step 9-11. “iAM Smart” user logs in “iAM Smart” Mobile App, “iAM Smart” Mobile App requests and displays 4-digit identification code from “iAM Smart” System. “iAM Smart” user reviews the digital signing authorisation request (e.g., continue or reject the request) and verifies the 4-digit identification code with Online Service Website;
- Step 12-13. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 14-15. “iAM Smart” System verifies the validity of necessary information, and return the authorisation result to “iAM Smart” Mobile App.
- Step 16. “iAM Smart” Mobile App invokes Online Service App using URL Scheme or Universal Link/App Link with required parameters such as “authCode”, “businessID”;
- Online Service Callback API (Callback with authCode to Online Service App)*
- Step 17. Online Service App sends authCode, businessID, etc. to Online Service Server;
- Step 18-19. When Online Service Server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e., openID) of the “iAM Smart” user. API data encryption is required;
- “iAM Smart” API (POST: Request accessToken & Tokenised ID)*
- Step 20-21. Online Service Server invokes the “iAM Smart” API with the accessToken and openID to obtain the result of anonymous digital signing. API data encryption is required;
- “iAM Smart” API (POST: Obtain Anonymous Digital Signing Result)*
- “iAM Smart” API (POST: Obtain Anonymous PDF Digital Signing Result)*
- Step 22. Online Service Server completes the document digital signing process, verify the result and acknowledges “iAM Smart” System the digital signing result using “iAM Smart” API with the same

“businessID” of the digital signing request. API data encryption is required;

“iAM Smart” API (POST: Online Service Acknowledges Digital Signing Result)

Step 23. Online Service Server matches the result using the “businessID” and shows the digital signing result in corresponding Online Service App (i.e., response for the polling in Step 6).

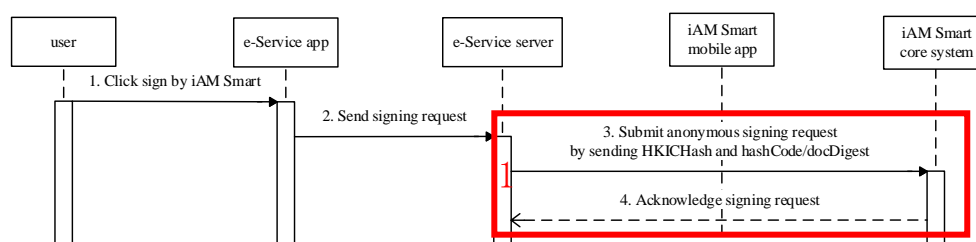
3.8.4.1 Implementing (1) POST: Request Anonymous Digital Signing



Please refer to Section 3.8.1.1 except:

- Online Service should determine the “iAM Smart” Mobile App is on the same device of Online Service App using program code.

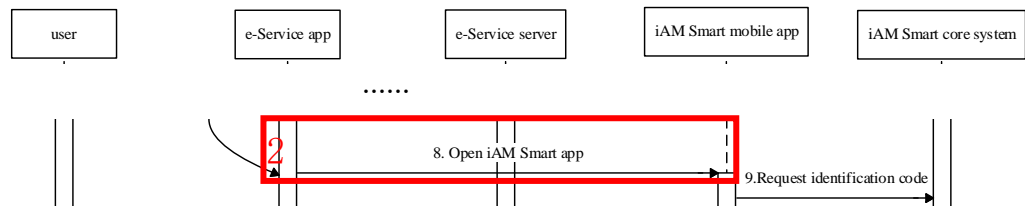
3.8.4.2 Implementing (1) POST: Request Anonymous PDF Digital Signing



Please refer to Section 3.8.1.2 except:

- Online Service should determine the “iAM Smart” Mobile App is on the same device of Online Service App using program code.

3.8.4.3 Implementing (2) URL Scheme: Open “iAM Smart” Mobile App for Getting Context



Pre-conditions

- Online Service Website and “iAM Smart” Mobile App are in the same device.
- Online Service has the “ticketID” provided by “iAM Smart” System for the digital signing request in step 4.
- Online Service should determine if it would generate the optional “state” request parameter to prevent CSRF attack. The “state” will be returned to Online Service for checking through the Online Service callback API for receiving the authorisation code from “iAM Smart” System in Step 16.

Post-conditions

- “iAM Smart” Mobile App will be launched.
- Online Service Website should keep synchronising with Online Service server for the callback response of the digital signing request from “iAM Smart” System.

Error conditions

- Nil

Request Parameters

- Please refer to “iAM Smart” API Specification.

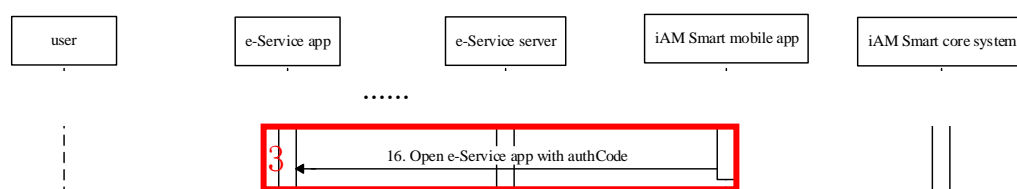
Notes

- Request parameter “ticketID”: Value provided by “iAM Smart” System for the anonymous request.
- Request parameter “source”: Value should be “App_Scheme” (for Online Service App URL scheme), or “App_Link” (for the Universal Link/App Link).
- Request parameter “redirectURI”: This is Online Service App URL scheme or Universal Link/App Link depending on the “source” parameter, which is used for receiving the authorisation code. The value should be URL encoded. For details, please refer to Section 6.4.1 of “iAM Smart” API

Specification. The value must be the same as provided during Online Service registration.

- Request parameter “state”: The value should be URL encoded.
- Set <Context> as “anon_hash-sign” or “anon_pdf-sign” in URL Scheme.

3.8.4.4 Implementing (3) Callback with authCode to Online Service App



Pre-conditions

- Please refer to Section 3.4.2.2.

Post-conditions

- Please refer to Section 3.4.2.2 except:
 - Online Service Server should match the callback result with corresponding Online Service App using the “businessID”.

Error conditions

- Please refer to Section 3.8.1.4 except:
 - This Online Service callback API is a URL Scheme, or Universal Link/App Link.

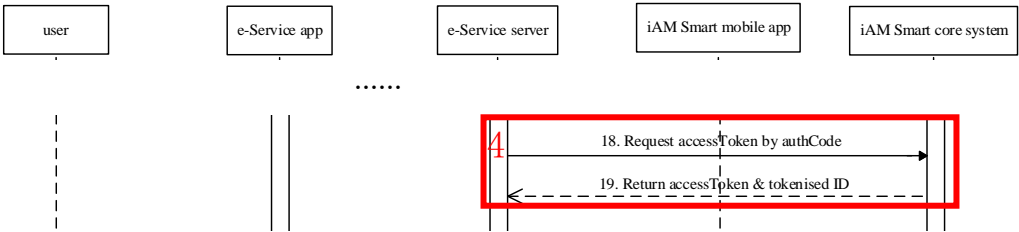
Callback Parameters

- Please refer to “iAM Smart” API Specification.

Notes

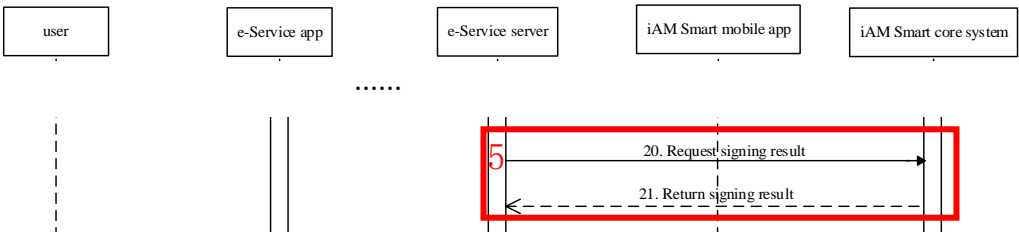
- Please refer to Section 3.4.2.2 except the following parameter:
 - Callback parameter “businessID”: Value returned by “iAM Smart” System should be the same one as Online Service submitted in the anonymous request.

3.8.4.5 Implementing (4) POST: Request accessToken & Tokenised ID



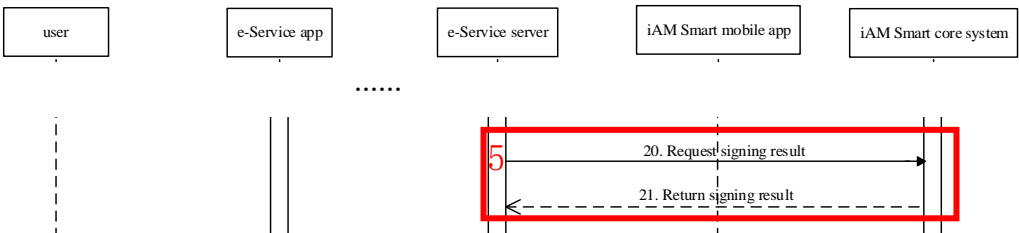
Please refer to Section 3.6.1.4.

3.8.4.6 Implementing (5) POST: Obtain Anonymous Digital Signing Result



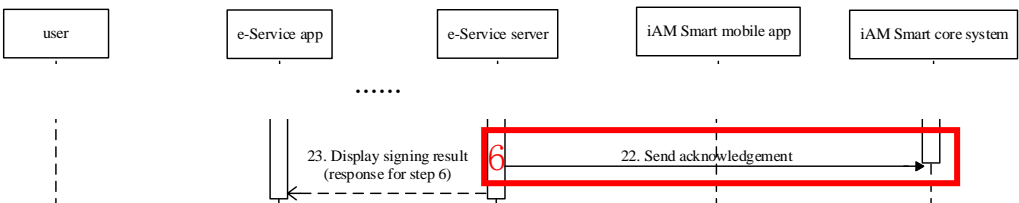
Please refer to Section 3.8.1.6.

3.8.4.7 Implementing (5) POST: Obtain Anonymous PDF Digital Signing Result



Please refer to Section 3.8.1.7.

3.8.4.8 Implementing (6) POST: Online Service Acknowledges Digital Signing Result



Please refer to Section 3.7.1.5.

3.9 WORKFLOWS FOR RE-AUTHENTICATION WITH SERVICE LOGIN

3.9.1 Scenario 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.

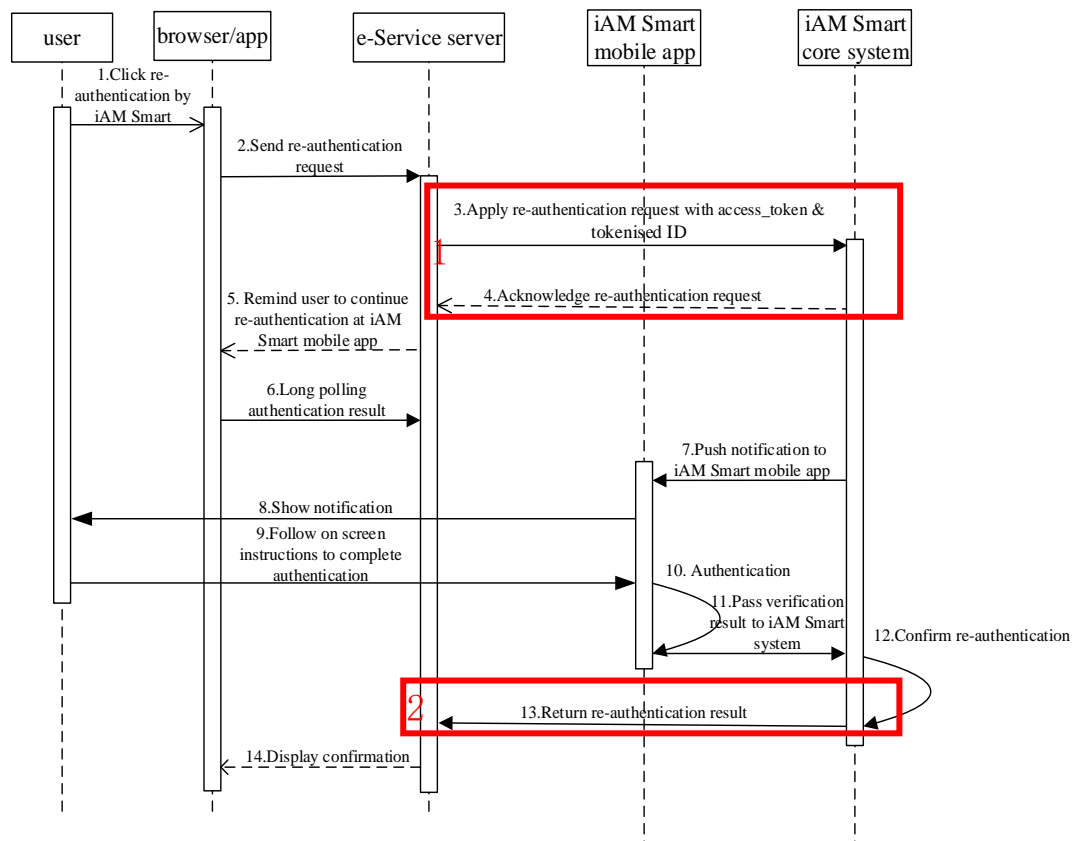
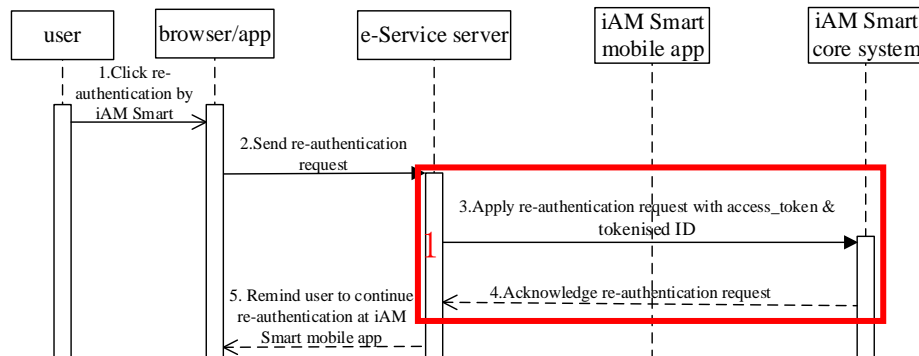


Figure-21 Re-authentication (Online Service Website/App in Different Device)

- Step 1. User clicks the “Re-authentication by iAM Smart” button in Online Service Website/App;
- Step 2. Online Service Website/App initiates Re-authentication request to Online Service Server with browser's user agent value (for Online Service Website) or “App_Scheme”/“App_Link” (for Online Service App) in this scenario (use as the value of request parameter “source”);

- Step 3. Online Service Server initiates Re-authentication request to invoke “iAM Smart” API with the “businessID” of this request, accessToken, Tokenised ID, etc;
- “iAM Smart” API (POST: Request Re-authentication)*
- Step 4. “iAM Smart” System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the Re-authentication request to Online Service server by returning POST response with parameter “authByQR” (sets to “true”) in this scenario;
- Step 5. Online Service Website/App shows instruction to inform “iAM Smart” user to process the Re-authentication authorisation request in “iAM Smart” Mobile App;
- Step 6. Online Service Website/App should keep synchronising with Online Service Server for request result (e.g., polling);
- Step 7. “iAM Smart” System pushes a notification message to “iAM Smart” Mobile App based on the Tokenised ID;
- Step 8-12. “iAM Smart” user logs in “iAM Smart” Mobile App, reviews the Re-authentication authorisation request (e.g., continue or reject the request) and authorises the request;
- Step 13. “iAM Smart” System invokes Online Service callback API to return the result with the “businessID” of the Re-authentication request. API decryption is required;
- Online Service Callback API (POST: Callback to Receive Re-authentication Result)*
- Step 14. Online Service Server processes and matches the result using the “businessID” and shows the result in corresponding Online Service Website/App.

3.9.1.1 Implementing (1) POST: Request Re-authentication



Pre-conditions

- Online Service must possess a valid accessToken of the “iAM Smart” user with authorisation scope “Re-authentication authorisation”.
- Online Service Website/App and “iAM Smart” Mobile App are in different devices in this scenario.
- Online Service generates a “businessID” for this Re-authentication request and use this identifier to match the callback result returned from “iAM Smart” System.
- API data encryption is required.

Post-conditions

- Online Service server should check the response parameter “authByQR” and “ticketID” and determine the next action. For details please refer to Section 3.2.1. “ticketID” will not be provided by “iAM Smart” System in this scenario.
- Online Service Website/App should show instructions to inform “iAM Smart” user to process the Re-authentication request in “iAM Smart” Mobile App when “authByQR” is “true”.
- Online Service server should keep synchronising with Online Service Website/App for the result returned from “iAM Smart” System.
- API data decryption is required.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
------------	-------------------	------------------

code - D80002	Failed to request re-authentication	Inform user the “iAM Smart” re-authentication request is failed and retry later
---------------	-------------------------------------	---

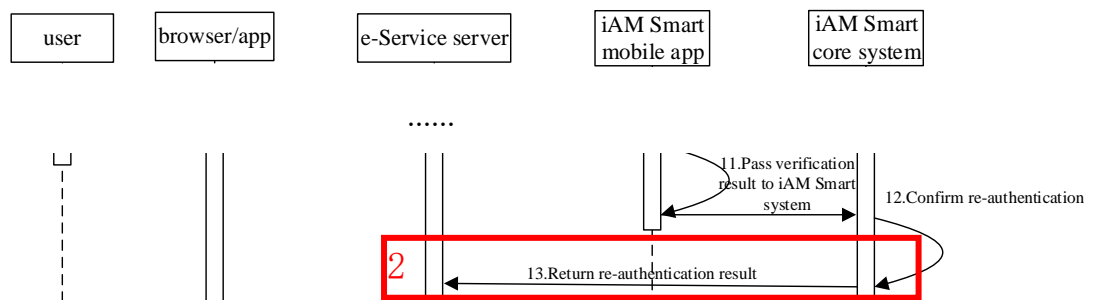
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Request parameter “source”: Value should be matched with Online Service client terminal (e.g., “App_Scheme”/“App_Link” or browser’s user agent value).
- Request Parameter - redirectURI: Value should equal to URI of “Callback to Receive Re-authentication Result” Online Service callback API. It has to be the same value as provided during Online Service registration.
- Response parameter “authByQR”: The value returned by “iAM Smart” System will be “true” in this scenario.
- “iAM Smart” System will not return the response parameter “ticketID” in this scenario.

3.9.1.2 Implementing (2) POST: Callback to Receive Re-authentication Result



Pre-conditions

- “iAM Smart” user can accept and complete the Re-authentication request.
- “iAM Smart” user can also reject the Re-authentication request.

Post-conditions

- API data decryption is required.
- Online Service server should match the callback result with corresponding Online Service Website/App using the “businessID”.

Error conditions

For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D80001	User rejected re-authentication request	Inform user the “iAM Smart” re-authentication request is rejected
code - D80002	Failed to request re-authentication	Inform user the “iAM Smart” re-authentication request is failed and retry later
code - D80003	Re-authentication request timeout	Inform user the “iAM Smart” re-authentication request is timeout and provide way for user to retry

Callback Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Callback parameter “isPassed”: It is the Re-authentication result. If it is the same “iAM Smart” user of the Tokenised ID, its value is “true”. Otherwise, its value is “false”.

3.9.2 Scenario 2: Re-authentication (Online Service Website in Same Device)

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

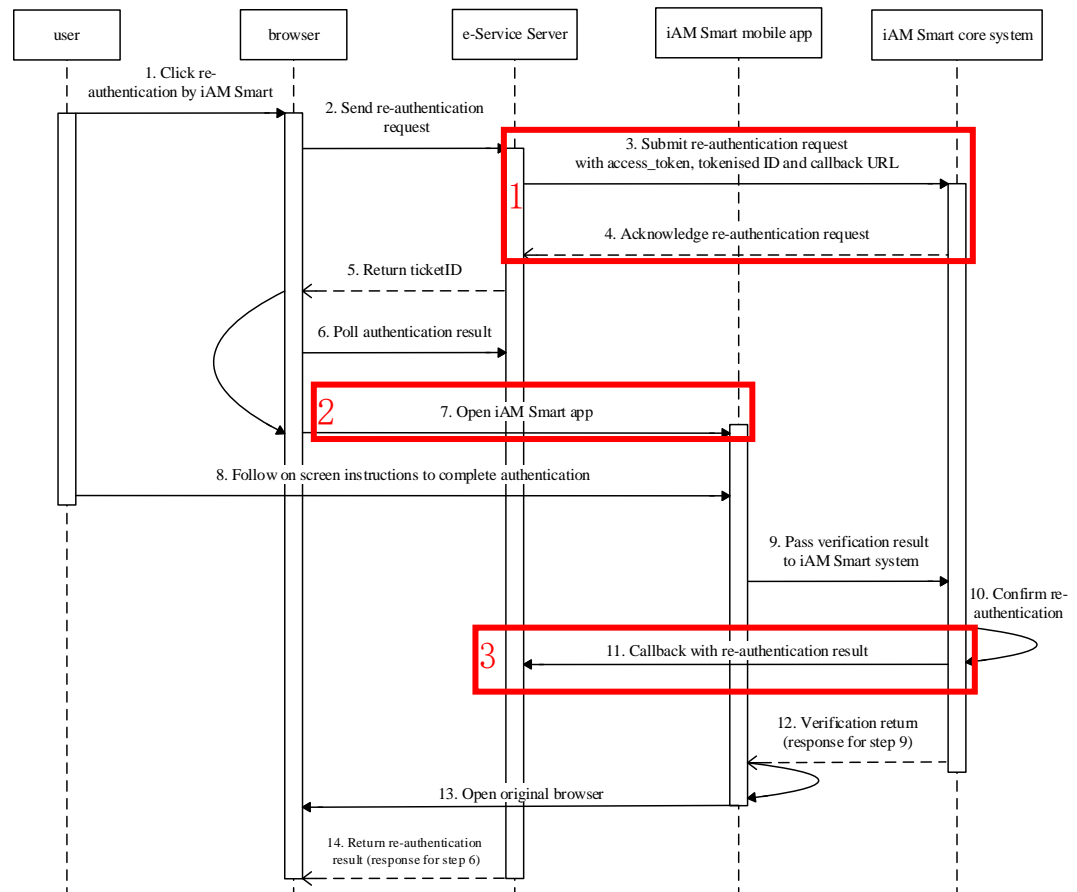


Figure-22 Re-authentication (Online Service Website in Same Device)

- Step 1. User clicks the “Re-authentication by iAM Smart” button in Online Service Website;
- Step 2. Online Service Website initiates Re-authentication request to Online Service Server with browser's user agent value (for use as value for request parameter “source”);
- Step 3. Online Service Server initiates Re-authentication request to invoke “iAM Smart” API with the “businessID” of this request, accessToken, Tokenised ID, etc. The request parameter “source” will be the browser's user agent value. API data encryption is required;

“iAM Smart” API (POST: Request Re-authentication)

- Step 4. “iAM Smart” System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the Re-authentication request to Online Service server by returning POST response with parameter “authByQR” (set to “false”) and “ticketID” in this scenario;
- Step 5-6. Online Service Server prepares and sends necessary parameters such as “authByQR”, “ticketID”, etc. to Online Service Website. Online Service Website should keep synchronising with the Online Service server for the request result (e.g., polling);
- Step 7. Online Service Website invokes “iAM Smart” Mobile App using URL Scheme with “ticketID” (set <Context> as “re-auth” in URL Scheme). Other parameters of this API are not used in this scenario;

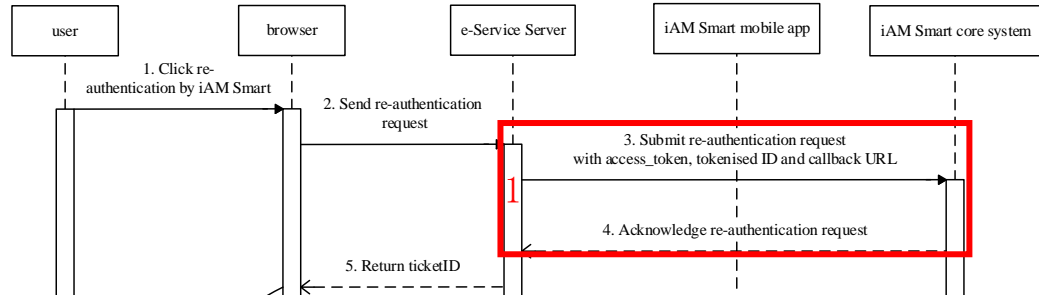
“iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Getting Context)

- Step 8-10. “iAM Smart” user logs in “iAM Smart” Mobile App, reviews the Re-authentication authorisation request (e.g., continue or reject the request) and authorises the request;
- Step 11. “iAM Smart” System invokes Online Service callback API to return the result with “businessID” of the Re-authentication request to Online Service server. API data decryption is required;

Online Service Callback API (POST: Callback to Receive Re-authentication Result)

- Step 12-13. “iAM Smart” System instructs “iAM Smart” Mobile App to invoke the original Online Service browser (using the browser's user agent name submitted in the Re-authentication request in Step 3);
- Step 14. Online Service Server processes and matches the result using the “businessID” and shows the result in the corresponding Online Service Website.

3.9.2.1 Implementing (1) POST: Request Re-authentication



Pre-conditions

- Please refer to Section 3.9.1.1 except:
 - Online Service Website and “iAM Smart” Mobile App are in the same device in this scenario.

Post-conditions

- Please refer to Section 3.9.1.1 except:
 - Response “authByQR” is “false”, “ticketID” will be provided by “iAM Smart” System in this scenario.

Error conditions

- Please refer to Section 3.9.1.1.

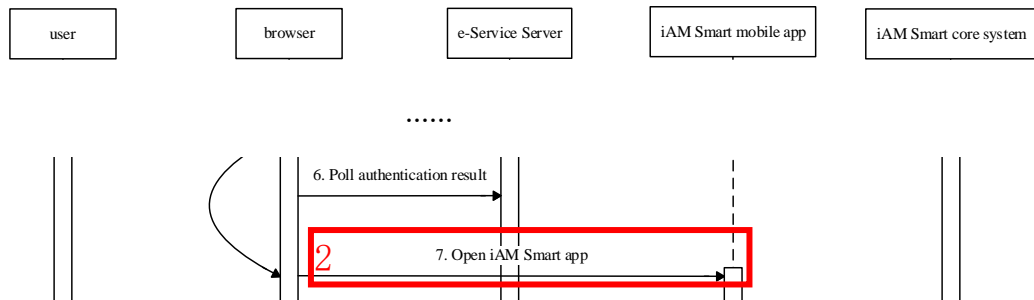
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.9.1.1, except for the following parameters:
 - Request parameter “source”: Value should be the browser’s user agent value.
 - Response parameter “authByQR”: The value is “false” in this scenario.
 - Response parameter “ticketID”: It will be provided by “iAM Smart” System in this scenario. It is a 36-byte (or less) UUID number (ASCII character set).

3.9.2.2 Implementing (2) URL Scheme: Open “iAM Smart” Mobile App for Getting Context



Pre-conditions

- Online Service Website and “iAM Smart” Mobile App are in the same device.
- Online Service has the “ticketID” provided by “iAM Smart” System for the Re-authentication request;
- Other parameters of this API are not used in this scenario.

Post-conditions

- “iAM Smart” Mobile App will be launched.
- Online Service Website should keep synchronising with Online Service server for the callback response of the Re-authentication request from “iAM Smart” System.

Error conditions

- Nil

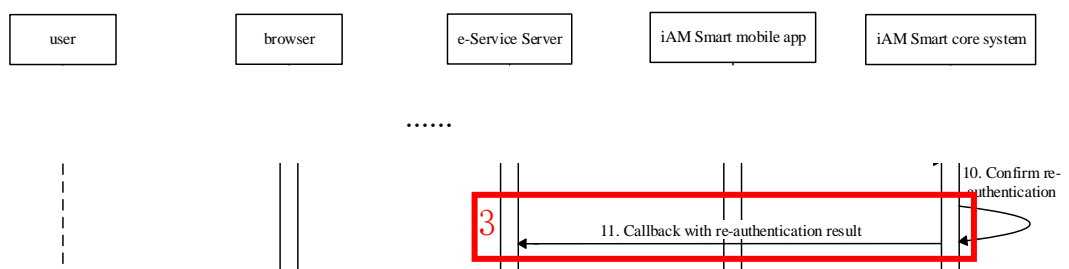
Request Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Set <Context> as “re-auth” in URL Scheme.

3.9.2.3 Implementing (3) POST: Callback to Receive Re-authentication Result



Please refer to Section 3.9.1.2.

3.9.3 Scenario 3: Re-authentication (Online Service App in Same Device)

The sequence diagram below shows how Online Service performs “iAM Smart” Re-authentication when Online Service mobile application and the “iAM Smart” Mobile App are running the same device.

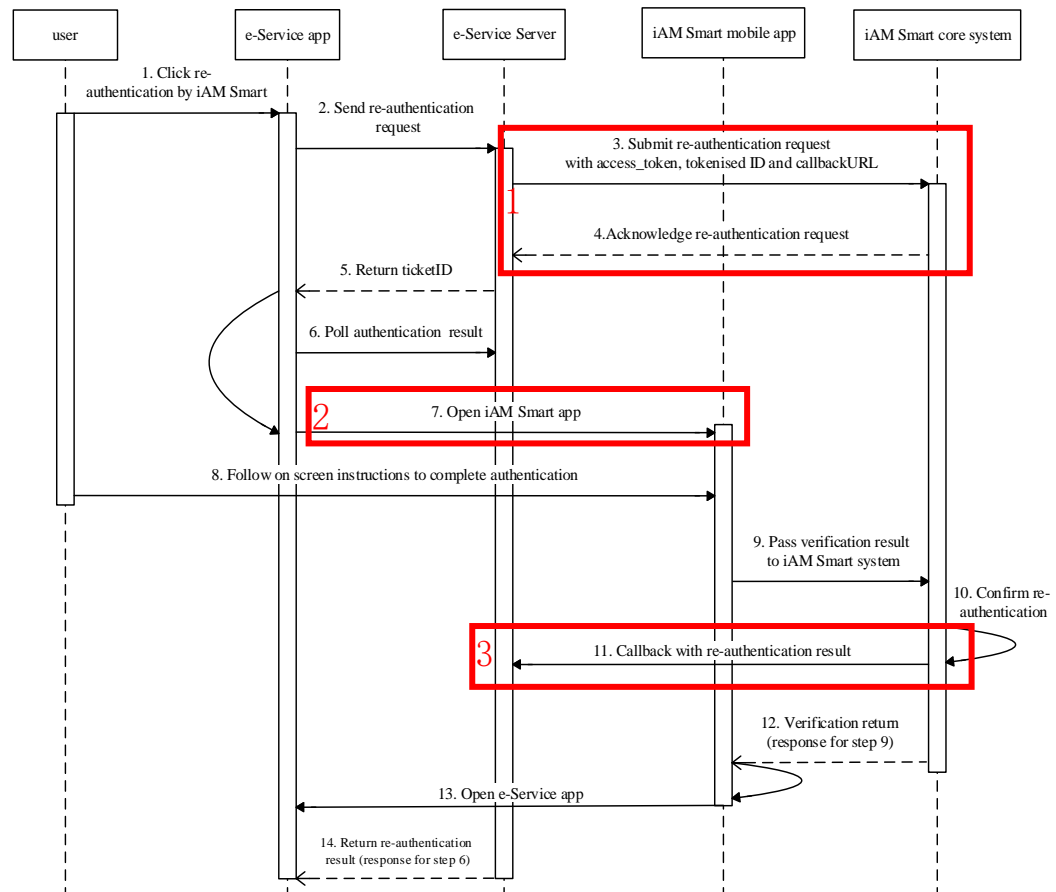


Figure-23 Re-authentication (Online Service App in Same Device)

- Step 1. User clicks the “Re-authentication by iAM Smart” button in Online Service App;
- Step 2. Online Service App initiates Re-authentication request to Online Service Server with “App_Scheme” or “App_Link” in this scenario (for use as value for request parameter “source”);
- Step 3. Online Service Server initiates Re-authentication request to invoke “iAM Smart” API with the “businessID” of this request, accessToken, Tokenised ID etc. The request parameter “source” will be “App_Scheme” or “App_Link”. API data encryption is required;

“iAM Smart” API (POST: Request Re-authentication)

- Step 4. “iAM Smart” System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the Re-authentication request to Online Service server by returning POST response with parameter “authByQR” (set to “false”) and “ticketID” in this scenario;
- Step 5. Online Service Server prepares and sends the necessary parameters such as “authByQR”, “ticketID”, etc. to Online Service App. Online Service App should keep synchronising with the Online Service server for the request result (e.g., polling);
- Step 7. Online Service App invokes “iAM Smart” Mobile App using URL Scheme with “ticketID” (set <Context> as “re-auth” in URL Scheme). Other parameters of this API are not used in this scenario;

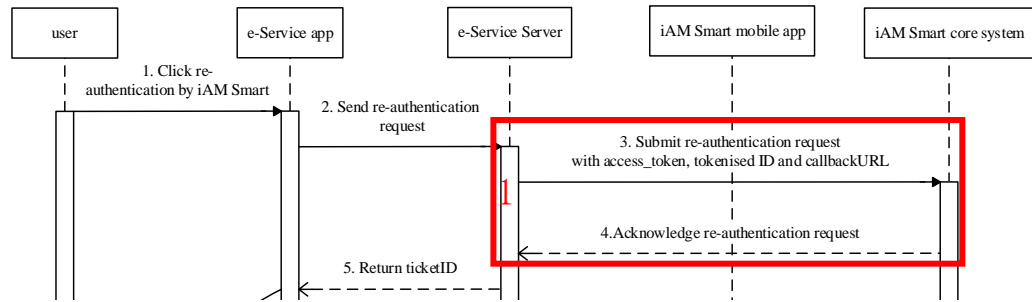
“iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Getting Context)

- Step 8-10. “iAM Smart” user logs in “iAM Smart” Mobile App, reviews the Re-authentication authorisation request (e.g., continue or reject the request) and authorises the request;
- Step 11. “iAM Smart” System invokes Online Service callback API to return the result with “businessID” of the Re-authentication request to Online Service server. API data decryption is required;

Online Service Callback API (POST: Callback to Receive Re-authentication Result)

- Step 12-13. “iAM Smart” System instructs “iAM Smart” Mobile App to invoke the Online Service App using URL Scheme, or Universal Link/App Link (“iAM Smart” System queries the information registered at Online Service registration depending on the “source” submitted in Step 3);
- Step 14. Online Service Server processes and matches the result using the “businessID” and shows the result in the corresponding Online Service App.

3.9.3.1 Implementing (1) POST: Request Re-authentication



Pre-conditions

- Please refer to Section 3.9.2.1 except:
 - Online Service App and “iAM Smart” Mobile App are in the same device in this scenario.

Post-conditions

- Please refer to Section 3.9.2.1.

Error conditions

- Please refer to Section 3.9.2.1.

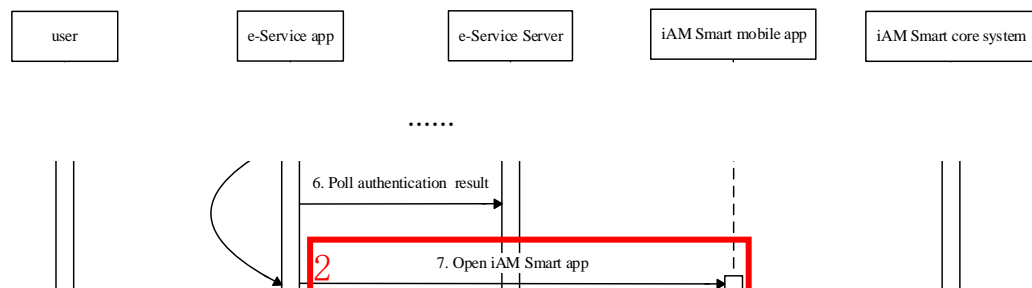
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

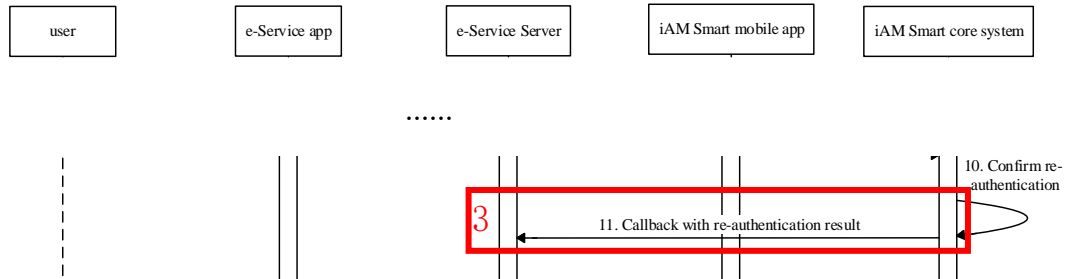
- Please refer to Section 3.9.2.1, except for the following parameter:
 - Request parameter “source”: Value should be “App_Scheme” (for the Online Service App URL scheme), or “App_Link” (for the Universal Link/App Link).

3.9.3.2 Implementing (2) URL Scheme: Open “iAM Smart” Mobile App for Getting Context



Please refer to Section 3.9.2.2.

3.9.3.3 Implementing (3) POST: Callback to Receive Re-authentication Result



Please refer to Section 3.9.1.2.

3.10 WORKFLOWS FOR DIRECT LOGIN & DIRECT ACCESS

3.10.1 Scenario 1: Direct Login with Online Service Website (aka Direct Login v2)

The sequence diagram below shows how users initiate Direct Login in that “iAM Smart” Mobile App and then automatically login Online Service website opened in a supported mobile browser.

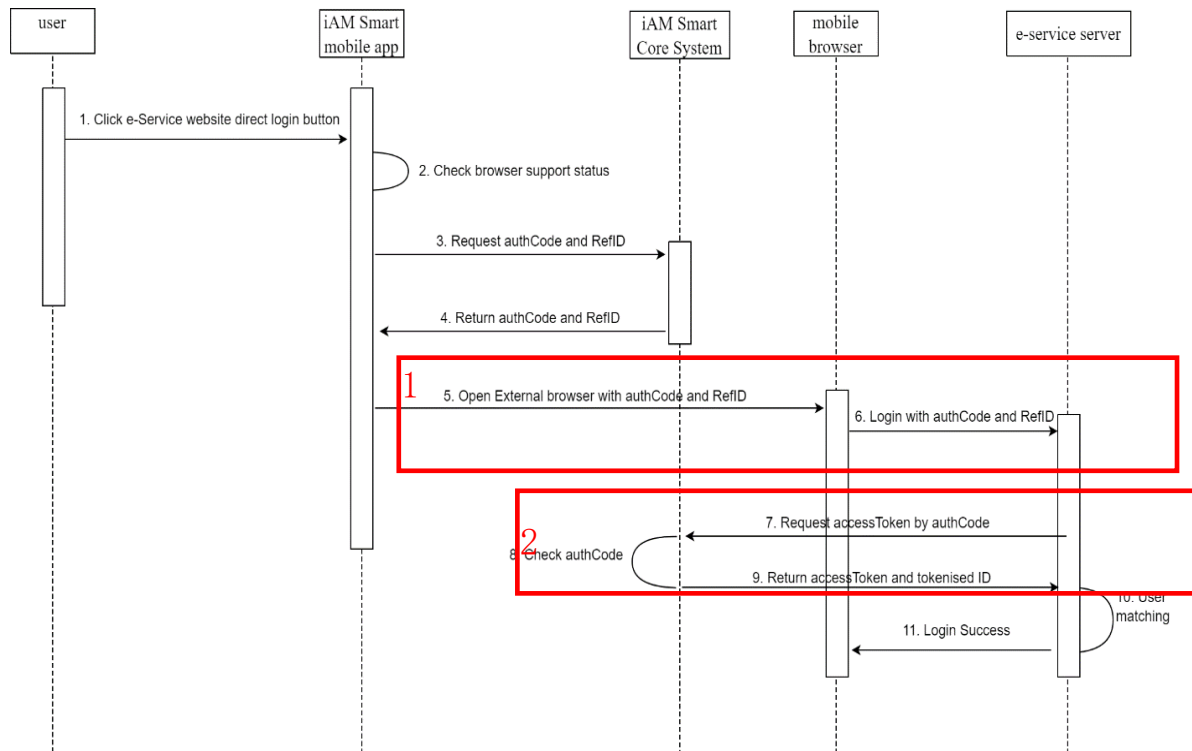


Figure-24 Direct login v2 from “iAM Smart” to Online Service

- Step 1. User opens “iAM Smart” Mobile App, browses Online Service catalogue or views the detailed information of Online Service, and clicks the button to invoke the Online Service;
- Step 2. “iAM Smart” Mobile App checks if the supported mobile browser and its browser version⁴ is installed. If there is no supported

⁴ “iAM Smart” will maintain the and review the supported browser list from time to time.

Currently the browser priority list for Android is Chrome (highest ranking), Firefox, Samsung

browser found, the default browser in “iAM Smart” mobile device will be opened and redirected to the fallback URL that has been registered in the “iAM Smart” Platform. Please refer to Section 3.10.3 for the fallback details;

Step 3-4. “iAM Smart” Mobile App request “iAM Smart” server for the authCode and RefID;

Step 5-6. “iAM Smart” Mobile App opens the external browser, and requests the direct login callback endpoint with the authCode and RefID according to the current UI display language. Online Service shall provide three direct login endpoints with three different languages (EN/TC/SC) and register in the “iAM Smart” Platform in advance. If there is any exception happened, “iAM Smart” Mobile App will open the fallback URL without the authCode and RefID instead of the direct login URL. In addition, the three fallback URLs of Online Service for three languages (EN/TC/SC) should have been registered so that the “iAM Smart” Mobile App can decide which URL is for fallback with respect to the current UI display language. Online Service should check if any login session exists and perform session management properly according to the business needs;

Online Service Callback API (GET: Callback with AuthCode to Online Service Server (Direct Login V2))

Step 7-9. When the Online Service Server gets the authCode, it should invoke the “iAM Smart” API to obtain the accessToken and the tokenised ID (i.e., openID) of the “iAM Smart” user for the selected Online Service. API data encryption is required;

“iAM Smart” API (POST: Request accessToken & tokenised ID)

Step 10. Online Service server then performs user matching using the tokenised ID with its user repository and determines the result of this login request.

Step 11. The Online Service Website shows the successful login page accordingly.

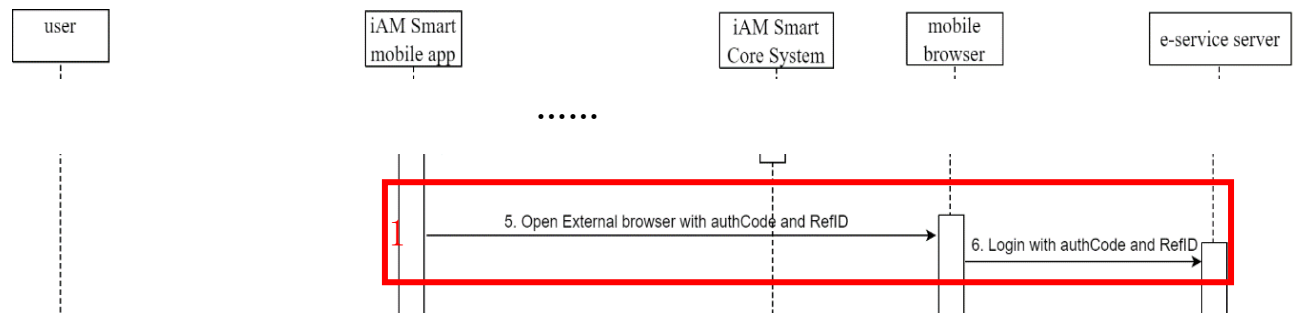
browser, Huawei browser, Xiaomi browser and Edge browser (lowest ranking). iOS in addition supports Safari.

Notes:

Online Services shall provide the following URLs to DPO for setup in the “iAM Smart” Platform:

No.	URL Type	URL Description	Described in
1.	Direct Login (English)	Receive authorisation code of web direct login with English language	Step 5-6
2.	Direct Login (Traditional Chinese)	Receive authorisation code of web direct login with Traditional Chinese language	Step 5-6
3.	Direct Login (Simplified Chinese)	Receive authorisation code of web direct login with Simplified Chinese language	Step 5-6
4.	Fallback / Direct Access (English)	Provide fallback for web direct login with English language when there is no supported browser, i.e., Online Service login page; or provide direct access to the Online Service in English without login requirement	Step 2 Step 5-6
5.	Fallback / Direct Access (Traditional Chinese)	Provide fallback for web direct login with Traditional Chinese language when there is no supported browser, i.e., Online Service login page; or provide direct access to the Online Service in Traditional Chinese without login requirement	Step 2 Step 5-6
6.	Fallback / Direct Access (Simplified Chinese)	Provide fallback for web direct login with Simplified Chinese language when there is no supported browser, i.e., Online Service login page; or provide direct access to the Online Service in Simplified Chinese without login requirement	Step 2 Step 5-6

3.10.1.1 Implementing (1) GET: An Online Service endpoint to receive authCode & RefID from external browser



This endpoint is to be implemented by Online Service. The implementation reuses “Callback with authCode to Online Service Server” (Section 6.4.2 of “iAM Smart” API Specification).

Pre-conditions

- “iAM Smart” user has initiated Online Service website via direct login in “iAM Smart” Mobile App. “iAM Smart” Mobile App has verified that there is a supported browser installed in the same mobile phone.

Post-conditions

- authCode will expire in 30 seconds and Online Service can only use the authCode once for requesting the accessToken from “iAM Smart” System.

Error conditions

- This API is a HTTP GET request. If parameter “error_code” is returned, it means the direct login request is failed.

Error Code	Error Description	Suggested Action
error_code – D20008	unregistered redirectURI	Check value is registered with “iAM Smart” System
error_code – D20012	insufficient scope	Check authorisation scope(s) requested is/are sufficient for invoking the corresponding “iAM Smart” API
error_code – D20015	Scope mismatch	Online service shall verify the scopes approved for the client id in

Error Code	Error Description	Suggested Action
		ESP and the scopes configured in Online Service catalogue.

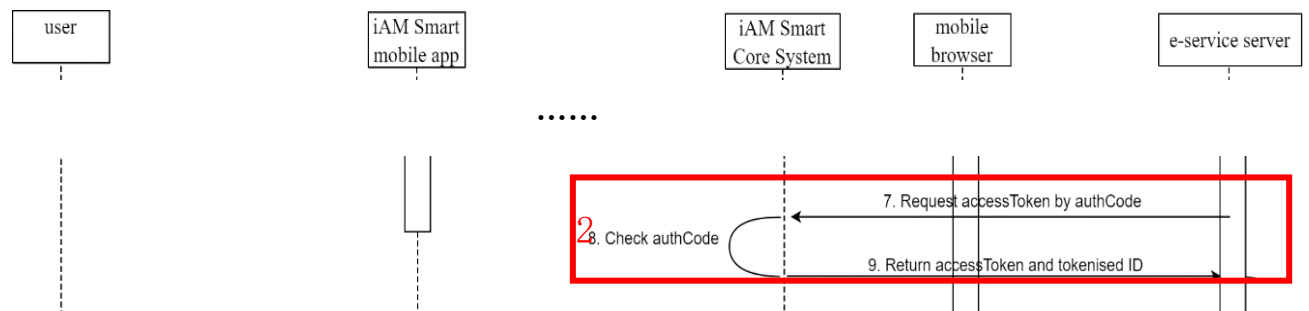
Callback Parameters

- Please refer to Section 6.4.2 of “iAM Smart” API Specification.

Notes

- Callback parameter “code”: It is the authCode for requesting the accessToken from “iAM Smart” System. Its validity lasts for 1 minute and can only be used once.
- If errors occur, “error_code” instead of “code” will be returned.
- RefID is a unique value for troubleshooting when necessary.
- Online Service should check if any login session exists and perform session management properly according to business needs.

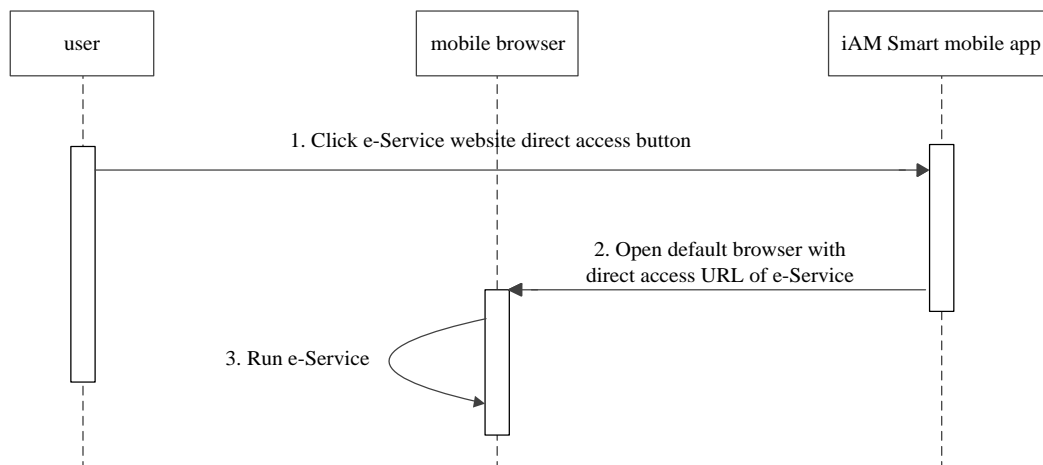
3.10.1.2 Implementing (3) POST: Request accessToken & tokenised ID



Please refer to Section 3.4.1.3.

3.10.2 Scenario 2: Direct Access with Online Service Website

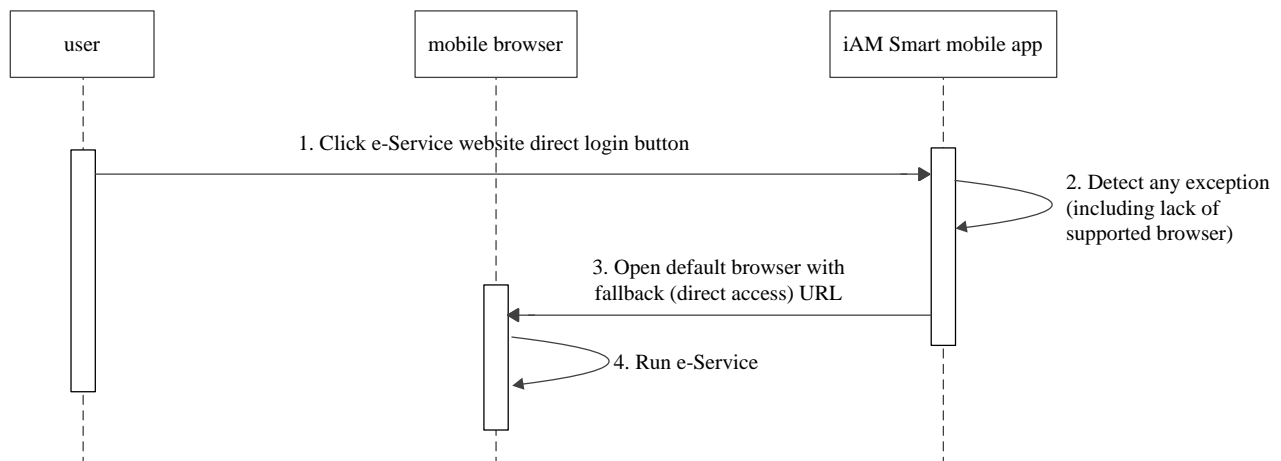
The sequence diagram below shows how users initiate Direct Access in the “iAM Smart” Mobile App.



- Step 1. User opens “iAM Smart” Mobile App, browses Online Service catalogue or views the detailed information of Online Service and clicks the button to invoke the Online Service.
- Step 2. “iAM Smart” Mobile App opens the default browser with direct access URL of Online Service. The three direct access URLs (or fallback URLs, for the English, Traditional Chinese, and Simplified Chinese languages) for the Online Service must be registered in advance in the “iAM Smart” Platform. “iAM Smart” Mobile App chooses the URL based on its current UI display language.
- Step 3. User starts using Online Service website in the default browser.

3.10.3 Scenario 3: Direct Login with Online Service Website (Fallback Mechanism)

The sequence diagram below shows how users initiate Direct Login in that “iAM Smart” Mobile App but fallback to direct access because there is no supported browser or exception happens.



- Step 1. User opens “iAM Smart” Mobile App, browses Online Service catalogue or views the detailed information of Online Service and clicks the button to invoke the Online Service.
- Step 2. “iAM Smart” Mobile App checks if any exception is detected. For example, “iAM Smart” Mobile App checks and does not find browser that support the direct login process.
- Step 3. “iAM Smart” Mobile App opens the fallback URL with the default browser. The three fallback URLs (or direct access URLs, for the English, Traditional Chinese, and Simplified Chinese languages) for the Online Service must be registered in advance in the “iAM Smart” Platform. “iAM Smart” Mobile App chooses the URL based on its current UI display language.
- Step 4. User starts using Online Service website with the default browser.

3.10.4 Scenario 4: Direct Login with Online Service App (Direct Login v2)

The sequence diagram below shows how users initiate Direct Login in the “iAM Smart” Mobile App and then automatically log in to the Online Service app installed in the same mobile device.

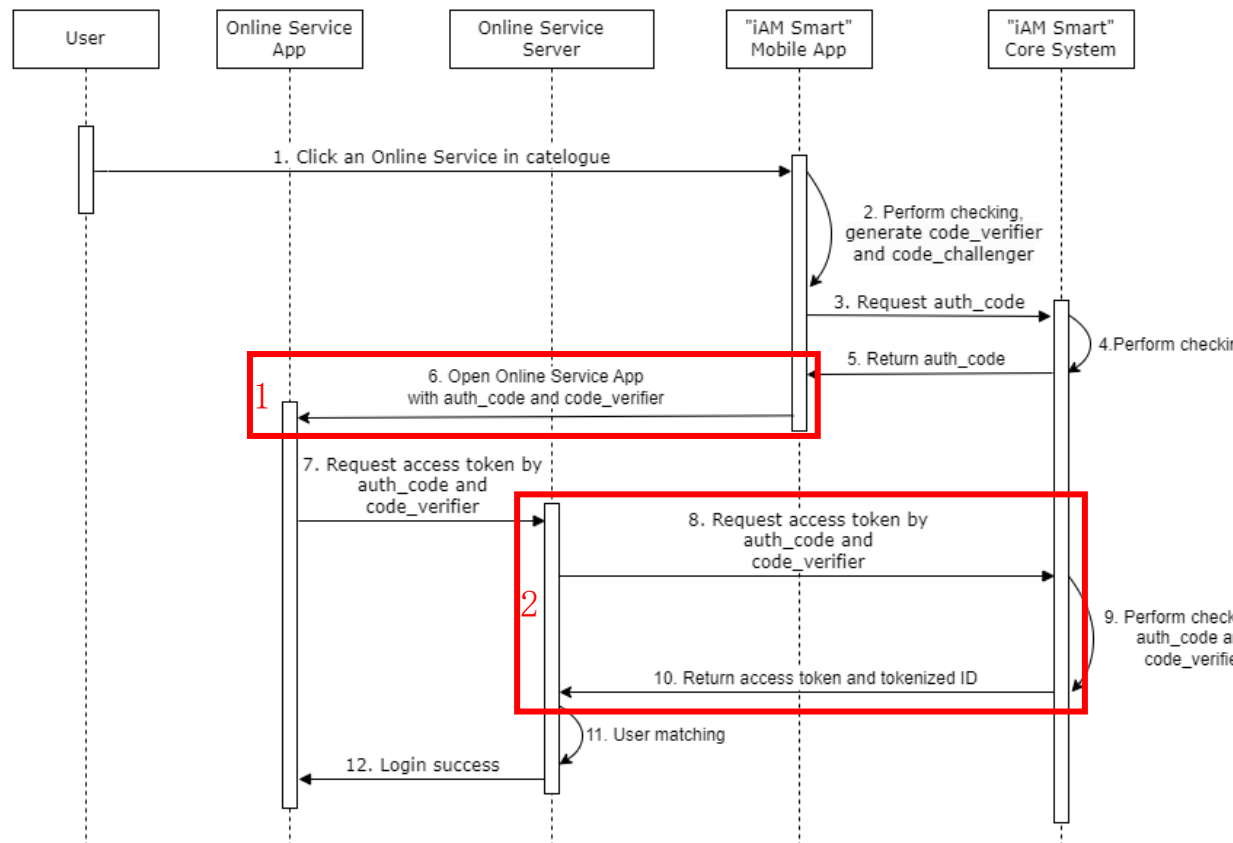


Figure-25 Direct Login with Online Service App

Pre-conditions: The Online Service App is registered to iAM Smart team with the following details,

- Basic information, including the service name, scope
- Package name (Submission should be made 3 months in advance and should be registered in iAM Smart App)
- iOS Universal Link and Android app fingerprint of the signing certificate correspond to package name in ESP

Step 1. User opens “iAM Smart” Mobile App, browses Online Service catalogue or views the detailed information of Online Service and clicks the “(Direct Login) App” button;

Step 2-5. “iAM Smart” Mobile App submits the login request to “iAM Smart” Core System. “iAM Smart” Core System verifies the necessary information of the login request and returns login result and authCode to “iAM Smart” Mobile App;

Step 6. “iAM Smart” Mobile App invokes Online Service App with required parameters using Universal Link (iOS) or package name (Android) according to the operating system;

“iAM Smart” API (URL scheme: Callback with authCode to Online Service App)

For iOS platform, the system browser will open the Universal Link and forward the request to Online Service server. Upon receiving the browser request, Online Service is recommended to instruct user to download and install the Online Service App.

For Android platform, “iAM Smart” will validate the Online Service App. If the Online Service App is installed and both the fingerprint of signing certificate and package name are valid, it will be launched by package name and the login request data, including the authCode and code_verifier will be forwarded. Otherwise, an alert dialog will be shown, the user can open the fallback URL via system browser by choosing the “More” option. Upon receiving the browser request, Online Service is recommended to instruct the user to download and install the Online Service App.

Please note that the Online Service app must check if another login session is already exists and perform session management properly according to the business needs. Online Service is strongly recommended to terminate the existing login session before it proceeds with the rest of the Direct Login workflow. The Online Service is recommended to ask the user if he/she confirms to logout the existing session and use “iAM Smart” Mobile App to login. Please refer to Section 3.10.4.1 for implementation details of the Online Service app on how to receive callback for direct login.

Step 7. Online Service App sends authCode, code_verifier, etc. to Online Service Server;

Step 8. Online Service Server provides the authCode and code_verifier to invokes the “iAM Smart” API to obtain the accessToken and the tokenised ID (i.e., openID) of the “iAM Smart” user with the selected Online Service. API data encryption is required.

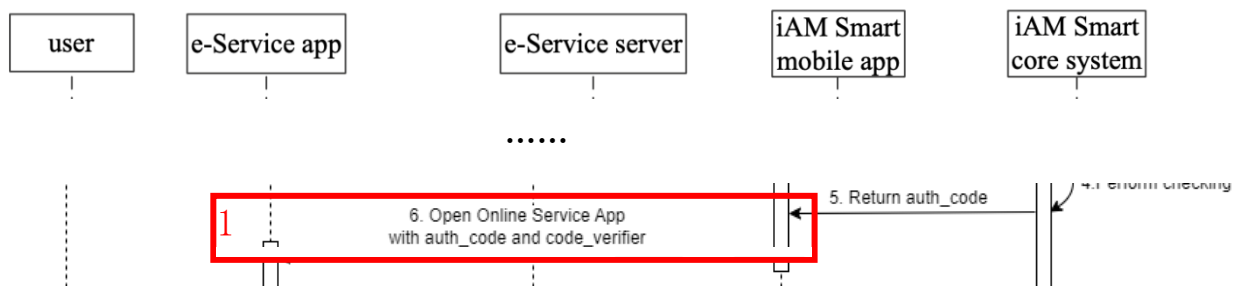
“iAM Smart” API (POST: Request accessToken & Tokenised ID)

Step 9-10. “iAM Smart” Core System verifies if the code_verifier received from Online Service Server and the code_challenge stored on the system are valid. “iAM Smart” Core System return the AccessToken and the tokenised ID (i.e., openID) of the “iAM Smart” user with the selected Online Service to Online Service Server.

Step 11. Online Service then performs user matching using the openID with its user repository and determines the result of this login request;

Step 12. Online Service App shows the login result (e.g., successful when Tokenised ID is matched with a user account of Online Service).

3.10.4.1 Implementing (1) Callback with authCode and code_verifier to Online Service App



Pre-conditions

- “iAM Smart” user has initiated Online Service app via direct login in “iAM Smart” Mobile App.
- Online Service has set up Universal Link and/or Android package name with fingerprint of signing certificate for the Online Service App.

Post-conditions

- authCode will be expired in 30 seconds and Online Service can only use the

authCode once for requesting the accessToken from “iAM Smart” System.

- **Callback Parameters**

Parameter	Type	Presence	Description
code	String	Required (Conditional)	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 value of code_verifier is generated by “iAM Smart” System.

- **Example URL Scheme (iOS Only)**

```
// Line breaks are for legibility only.
<Universal Link><landing location>
?code=0ad186353c424c64897fcc00445c9ba1
&code_verifier=3LpK1f7Qs8wNcIxRyO5JvD2Am9Vbhz0Zt6G4BdFgH1TjXeYkSpWqEuMrCnU
iVolx7ZaQ9s8W7m6D4b3F2h5J9n0B8v7c6x5Z2A1s3D4f7G8h9J0k1L2q3v5B4n6Mcis8
```

- **Example Package Name (Android Only)**

```
Intent ii=new Intent(<package name>, <activity name>);
ii.putExtra("code", "0ad186353c424c64897fcc00445c9ba1");
ii.putExtra("code_verifier", "DxjdFp0vKYCY4F0DE6M3eEadLxcJBTh6k4LZ9J5z");
ii.putExtra("activityParams", <activity params>);
startActivity(ii);
```

- **Set up Universal Link**

For the steps to set up Universal Link on iOS, please refer to Guides and Reference for iOS in <https://developer.apple.com/ios/universal-links/>.

Please note that the Universal Link may not be triggered by “iAM Smart” Mobile App only.

- **Set up for Android**

For the steps to enable “iAM Smart” to open Online Service app via Android

Package Name, modification of android files are required.

An activity must be appended in AndroidManifest.xml. Please note that the name of activity can be adjust according to the requirement of Online Service App.

```
<activity
    android:name = ".PackageNameIntentActivity"
    android:launchMode="singleTask" // Example only
    android:exported = "true" // Example only
    android:screenOrientation = "portrait">
</activity>
```

Within the `PackageNameIntentActivity` class, the `authCode` and `code_verifier` should be captured and appropriately handled.

```
val authCodeId = intent.getStringExtra("code")
val codeVerifier = intent.getStringExtra("code_verifier")
val activityParams = intent.getStringExtra("activityParams")
```

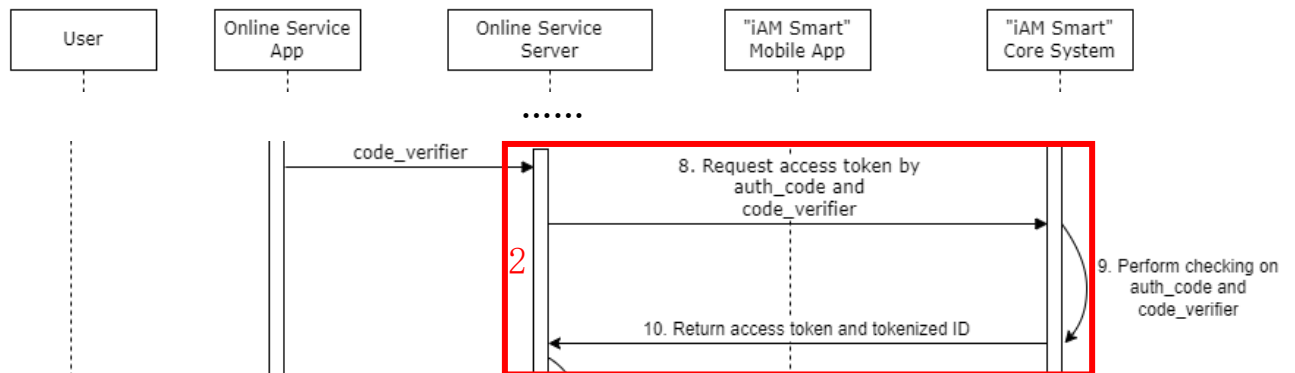
● Session Management

The Online Service app must check if another login session already exists and perform session management properly according to the business needs. Online Service is strongly recommended to terminate the existing login session before it proceeds with the rest of the Direct Login workflow. As the Universal Link may not be triggered by “iAM Smart” Mobile App, the Online Service app should take into this consideration to design the session management.

Notes

- Please refer to Section 3.4.2.2.

3.10.4.2 Implementing (2) POST: Request accessToken & tokenised ID



Pre-conditions

- Online Service should request the accessToken using a valid authCode. authCode is valid for 30 seconds and should not be used more than once.
- API data encryption is required

Post-conditions

- Please refer to Section 3.4.1.3.

Error conditions

- Please refer to Section 3.2.3.

● Request and Response Parameters

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 and Response 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)	The same code_verifier value returned by “iAM Smart” Core System should be used. Required for Direct Login v2.
isDirectLoginV2	Boolean	Optional	<p>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.</p> <p>Online service must submit the parameter "isDirectLoginV2" and set this value to <i>true</i> if the authCode is received by the callback endpoint for Direct Login v2.</p> <p>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.</p> <p>The default value is "false".</p>
grantType	String	Required	The value MUST be set to <code>authorization_code</code> .

● Example Request

```
// Line breaks are for legibility only.
// Please refer to section 錯誤！找不到參照來源。 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",
"code_verifier": "3LpK1f7Qs8wNcIxRyO5JvD2Am9VbhZ0Zt6G4BdFgHlTjXeYkSpWqEuMr
CnUiVo1x7ZaQ9s8W7m6D4b3F2h5J9n0B8v7c6x5Z2A1s3D4f7G8h9J0k1L2q3v5B4n6Mcis8",
"isDirectLoginV2": "true"
"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	<div><p>The datetime of the user complete registration at “iAM Smart” System. The value will be updated when either of the following is valid:-</p><p>(1) If any one of the following verified data are changed.</p><table><tr><td>English name</td></tr><tr><td>Chinese name (* not applicable if it was marked as unverified during registration)</td></tr><tr><td>Gender</td></tr><tr><td>Date of birth</td></tr></table><p>(2) User re-register “iAM Smart” after “iAM Smart” de-registration.</p><p>The modification time will be expressed in the number of milliseconds since January 1, 1970 00:00:00 GMT.</p></div>	English name	Chinese name (* not applicable if it was marked as unverified during registration)	Gender	Date of birth
English name							
Chinese name (* not applicable if it was marked as unverified during registration)							
Gender							
Date of birth							

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 錯誤! 找不到參照來源。
// 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
// The decrypted body
{
  "txID": "<T=938ffb193b4b4370b6c2584372c6a588>",
  "code": "D40004",
  "message": "authCode not exist or expired",
}
```

Notes

- Please refer to Section 3.4.1.3, except for the following parameters:
- Request parameter “code_verifier”: code_verifier and hash of “code_verifier” (code_challenge) are used to ensure the request validity and prevent the code interception attack.

3.10.5 Scenario 5: Direct Login with Online Service App (Direct Login v1)

The sequence diagram below shows how users initiate Direct Login in the “iAM Smart” Mobile App and then automatically log in to the Online Service app installed in the same mobile device.

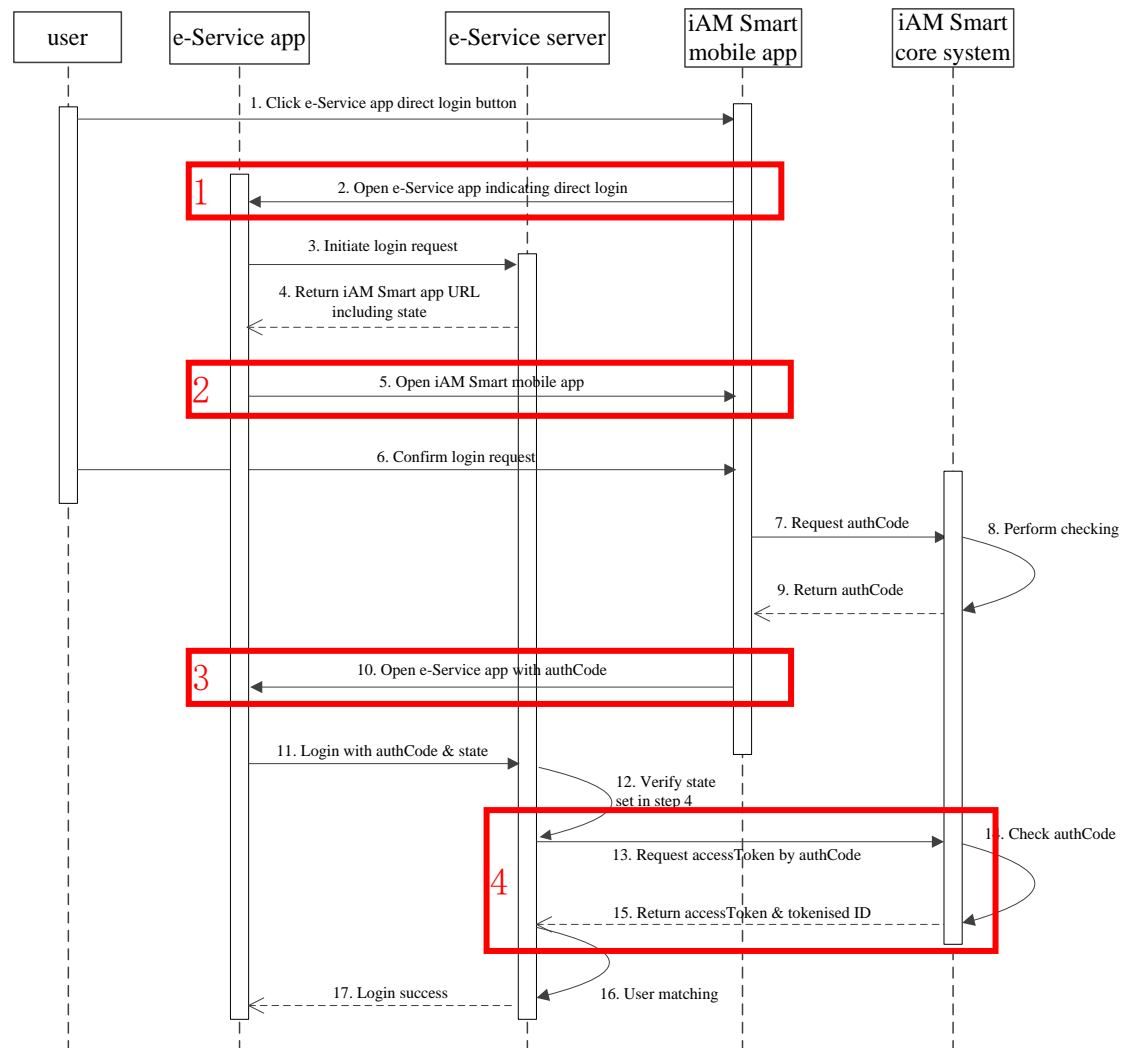


Figure-26 Direct Login with Online Service App

- Step 1. User opens “iAM Smart” Mobile App, browses Online Service catalogue or views the detailed information of Online Service and clicks the “(Direct Login) App” button;
- Step 2. “iAM Smart” Mobile App opens Online Service app through Universal Link URL (iOS) or App Links URL (Android) that has been predefined in the “iAM Smart” Platform for opening the Online Service app as direct login. If the Online Service app is not installed in the same mobile device, the system browser will

open the URL and forward the request to Online Service server. Upon receiving this browser request, Online Service is recommended to redirect the request to Online Service app detail page in the App Store (iOS) or Play Store (Android).

The Online Service app must check if another login session already exists and perform session management properly according to the business needs. Online Service is strongly recommended to terminate the existing login session before it proceeds with the rest of the Direct Login workflow. The Online Service is recommended to ask the user if he/she confirms to logout existing session and use “iAM Smart” Mobile App to login. Please refer to Section 3.10.4.1 for implementation details of the Online Service app on how to receive request for direct login.

Step 3-4. Online Service App requests Online Service Server to prepare the request parameters including “state”, “clientId”, “redirectURI”, “scope”, “source” (set as App_Scheme or App_Link), etc. and constructs the URL Scheme to invoke “iAM Smart” Mobile App by Online Service App;

Step 5. Online Service App invokes “iAM Smart” Mobile App using URL Scheme with request parameters (set <Context> as “auth” in URL Scheme);

“iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Authentication)

Step 6-7. “iAM Smart” user logs in the “iAM Smart” Mobile App and confirms the Online Service login request and then “iAM Smart” Mobile App submits the login request to “iAM Smart” Server;

Step 8-9. “iAM Smart” Server verifies the necessary information of the login request and returns login result to “iAM Smart” Mobile App;

Step 10. “iAM Smart” Mobile App invokes Online Service App with required parameters using URL Scheme or Universal Link/App Link;

Online Service Callback API (Callback with authCode to Online Service App)

Step 11. Online Service App sends authCode, state, etc. to Online Service Server;

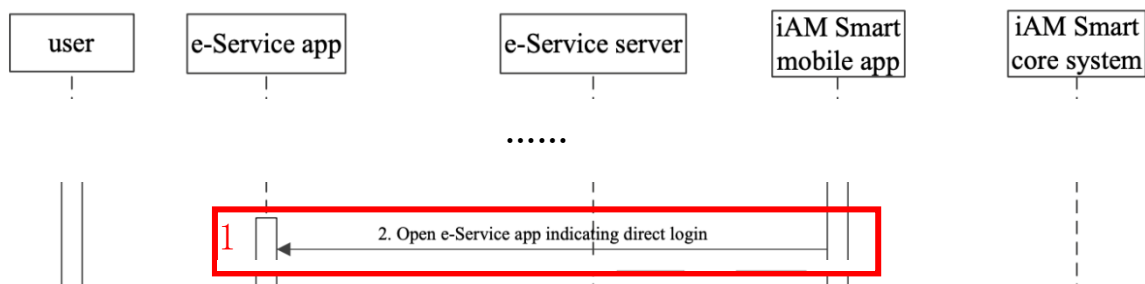
Step 12-15. When Online Service Server gets the authCode, it shall verify the state with the one generated in step 4 and then invokes the “iAM Smart” API to obtain the accessToken and the tokenised ID (i.e., openID) of the “iAM Smart” user with the selected Online Service. API data encryption is required;

“iAM Smart” API (POST: Request accessToken & tokenised ID)

Step 16. Online Service then performs user matching using the openID with its user repository and determines the result of this login request;

Step 17. Online Service App shows the login result (e.g., successful when Tokenised ID is matched with a user account of Online Service).

3.10.5.1 Implementing (1) GET: Online Service app to receive request for direct login



Pre-conditions

- “iAM Smart” user has initiated Online Service app via direct login in “iAM Smart” Mobile App.
- Online Service has set up Universal Link and/or App Links for its app.

Post-conditions

- Online Service app is opened. It will request Online Service server to set a state value that will be used to compare with the state returned with authCode in “iAM Smart” request (i.e., step 10) during verification in step 12.

Error conditions

- Online Service needs to implement this GET request. If Universal Link or App Links has not been set up, Online Service is recommended to redirect the request to Online Service app detail page in the App Store (iOS) or Play Store (Android).

Implementation Details

● API Description

Name	Description
Service Full Name	Open Online Service app indicating direct login
Universal Link URL or App Link URL	<code>https://<Online_Service_domain>/<Online_Service_context>/<app_direct_login_endpoint>?platform=<platform>&scope=<scope></code>
Request Type	GET
Service Version	1.0
Description of Service	Online Service should register this endpoint in the “iAM Smart” Platform and implement it as Universal Link URL (iOS) or App Link URL (Android). If the Online Service app is installed in the same mobile device, the Online Service app will be opened when “iAM Smart” Mobile App invokes this URL. Otherwise, the URL will be opened with system browser and we recommend Online Service to redirect the browser request to the Online Service app detail page in the App Store (iOS) or Play Store (Android).

● Request Parameters

Parameter	Type	Condition	Description
platform	String	Required	Either iOS or Android. It helps Online Service to decide where the request should be redirected to (i.e., App Store or Play store) when Online Service app has not been installed in the same mobile device.
scope	String	Required	The previously agreed scope (e.g., eidapi_auth eidapi_sign). It will be URL encoded.

● Example Request

```
GET
https://esd-isit.staging-
eid.gov.hk/eservice/directLogin?platform=iOS&scope
=eidapi_auth
```

- **Set up Universal Link or App Link**

For the steps to set up Universal Link on iOS, please refer to Guides and Reference for iOS in <https://developer.apple.com/ios/universal-links/>.

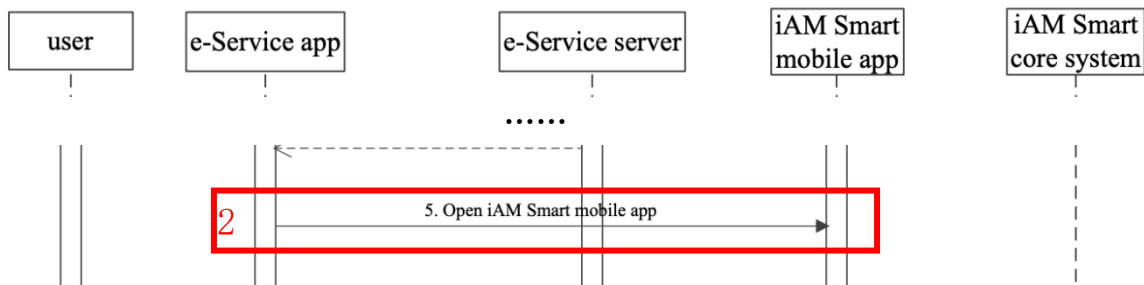
For the steps to set up App Link on Android, please refer to the Guides for Android in <https://developer.android.com/training/app-links>.

Please note that the Universal Link / App Link may not be triggered by “iAM Smart” Mobile App only.

- **Session Management**

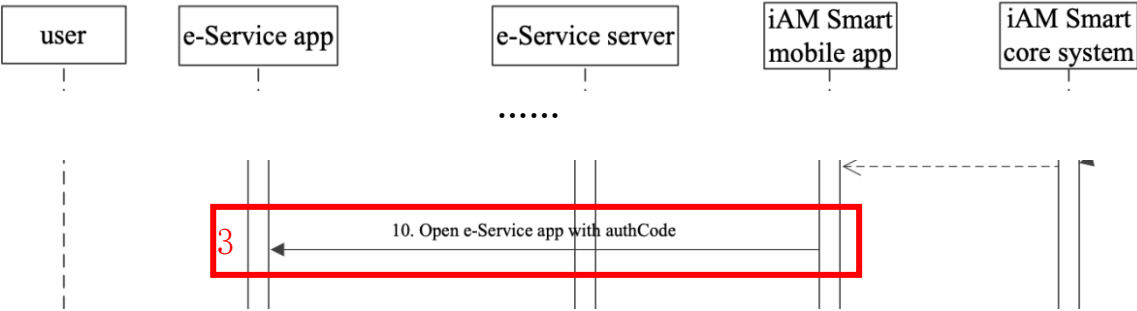
The Online Service app must check if another login session already exists and perform session management properly according to the business needs. Online Service is strongly recommended to terminate the existing login session before it proceeds with the rest of the Direct Login workflow. As the Universal Link / App Link may not be triggered by “iAM Smart” Mobile App, the Online Service app should take into this consideration to design the session management. Online Service is strongly recommended to ask the user if he/she confirms to logout existing session and use “iAM Smart” Mobile App to login.

3.10.5.2 Implementing (2) URL Scheme: Open “iAM Smart” Mobile App for Authentication



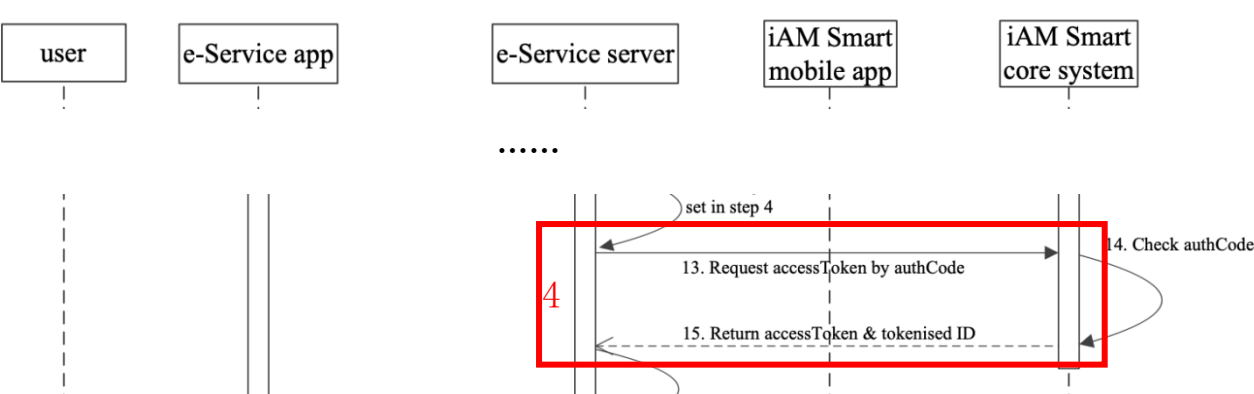
Please refer to Section 3.4.4.1.

3.10.5.3 Implementing (3) Callback with authCode to Online Service App



Please refer to Section 3.4.4.2.

3.10.5.4 Implementing (4) POST: Request accessToken & tokenised ID



Please refer to Section 3.4.4.3.

3.11 WORKFLOWS FOR BULK DIGITAL SIGNING WITH SERVICE LOGIN

There are different use cases for “iAM Smart” System and Online Services. Interactions among user, Online Service and “iAM Smart” System can be summarised into the following sequence diagrams.

3.11.1 Scenario 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.

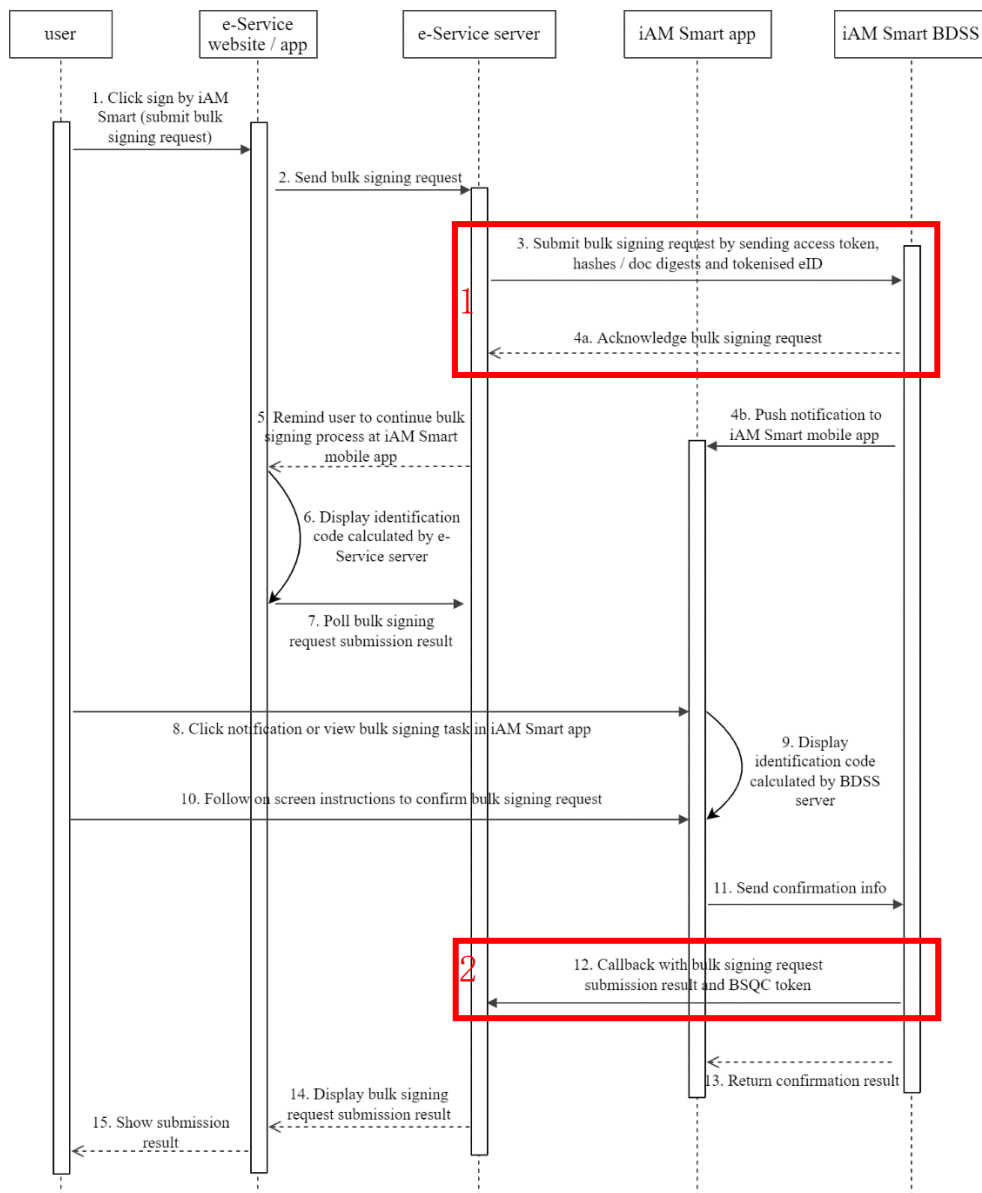
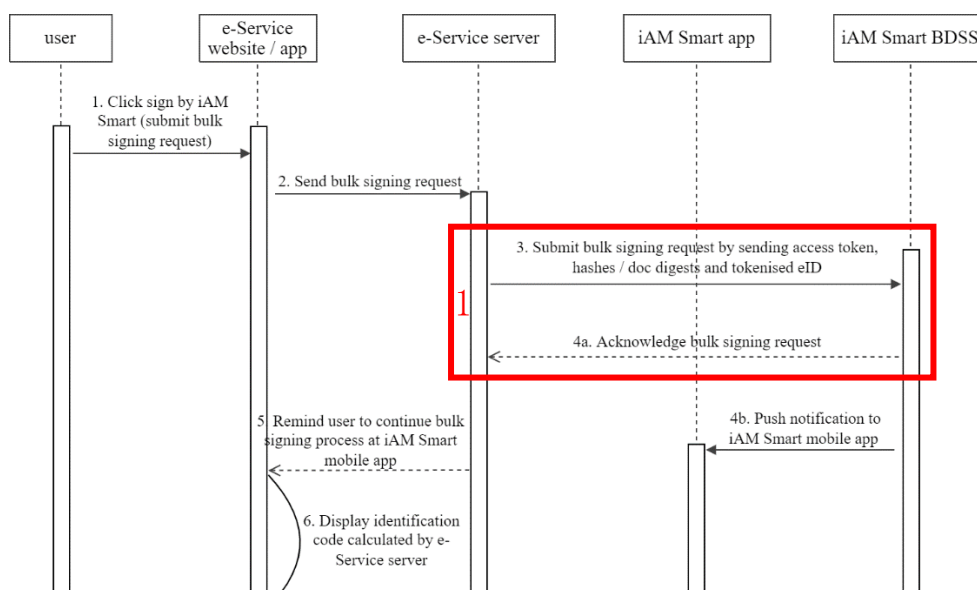


Figure-27 Bulk Digital Signing (Online Service Website/App in Different Device)

- Step 1. User clicks the “Sign by iAM Smart” button in Online Service Website/App;
- Step 2-3. Online Service initiates bulk digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, accessToken, openID (Tokenised ID), source, redirectURI, callbackResultURI, state, HKICHash, department, serviceName, requestName, maxCallbackSigs, documents, etc.
- The request parameter “documents” contains the Document Hash (hashCode) / PDF digest (docDigest) of all documents, signature algorithm (only for document). The request parameter “source” will be the browser's user agent value (for Online Service Website) or “App_Scheme”/“App_Link” (for Online Service App) in this scenario. API encryption is required;
- “iAM Smart” API (POST: Request Bulk Signing)*
- Step 4a. “iAM Smart” System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the digital signing request to Online Service server by returning POST response with parameter “authByQR” (set to “true”) in this scenario;
- Step 4b. “iAM Smart” System pushes a notification message to the “iAM Smart” Mobile App based on the Tokenised ID;
- Step 5-6. Online Service server concatenates all document hashes and PDF digests and hash of Tokenised ID to calculate a 6-digit identification code and instructs Online Service Website/App to display the instructions with the 6-digit identification code to inform “iAM Smart” user to process the signing authorisation request in “iAM Smart” Mobile App;
- Step 7. Online Service Website/App should keep synchronising with Online Service server for the digital signing processing result (e.g. polling);
- Step 8. “iAM Smart” user opens “iAM Smart” Mobile App, reviews the digital signing authorisation request (e.g. continue or reject the request).
- Step 9. “iAM Smart” user verifies the 6-digit identification code with Online Service Website/App;

- Step 10-11. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 12. “iAM Smart” BDSS invokes Online Service callback API to return the BSQCToken corresponding with openID of the digital signing request. API data encryption by AES;
- “iAM Smart” API (POST: Callback to Receive BSQC Token)*
- Step 13. “iAM Smart” BDSS saves the signing request to MQ queue and returns the result with the same “businessID” of the signing request.
- Step 14-15. Online Service server returns the submission result (status of Pending for Digital Signing) to Online Service Website/App, and then Online Service Website/App returns the result to user. Response to Step 7.

3.11.1.1 Implementing (1) POST: Request Bulk Digital Signing



Pre-conditions

- Online Service must possess a valid accessToken of the “iAM Smart” user with authorisation scope “bulk signing authorisation”.
- Online Service Website/App and “iAM Smart” Mobile App are in different devices in this scenario.
- The “iAM Smart” user who signs the document must have a sign version “iAM Smart”.
- Online Service generates a “businessID” for this bulk signing request and uses this identifier to match the callback result returned from “iAM Smart” BDSS.
- Online Service generates the Document Hash as “hashCode” parameter from the original documents to be signed using a hash algorithm that can suit the specific business need. It is recommended to use a hash algorithm that is at least SHA-256 or equivalent.
- Online Service generates the digest as the parameter “docDigest” from the pdf document to be signed using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter.
- API data encryption is required.

Post-conditions

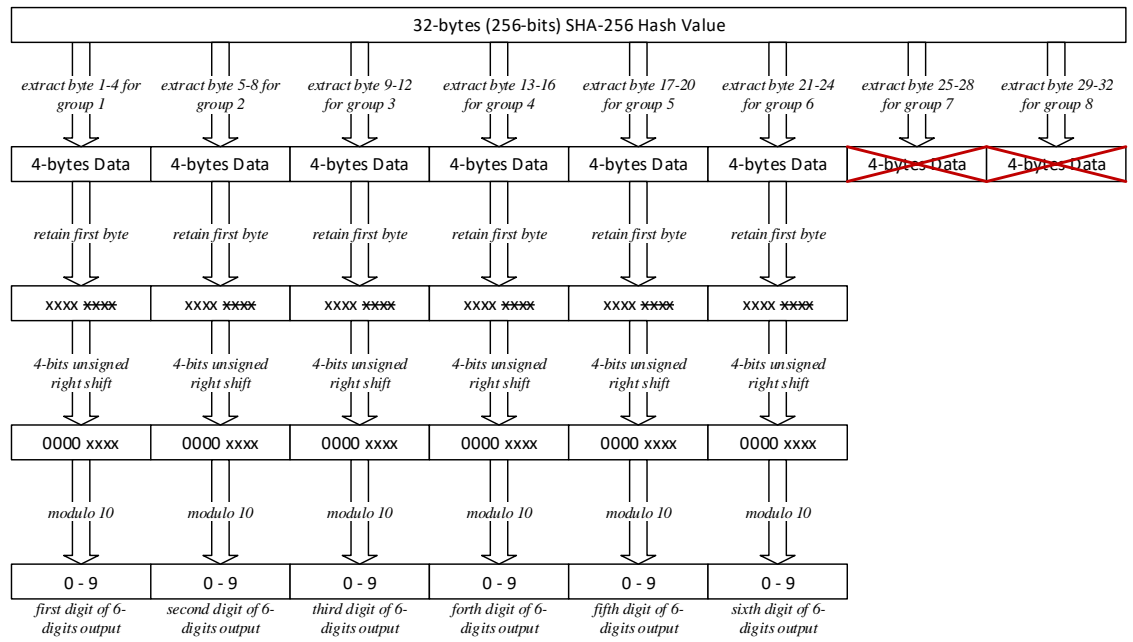
- Online Service server should check the response parameter “authByQR” and “ticketID” and determine the next action. For details, please refer to Section

3.2.1. “ticketID” will not be provided by “iAM Smart” System in this scenario.

- Online Service has to calculate a 6-digit identification code using all Document Hashes and PDF digests and Tokenised ID hash value. Online Service Website/App should show the 6-digit identification code and instruction to inform “iAM Smart” user to process the digital signing request in “iAM Smart” Mobile App when “authByQR” is “true”.

6-digit identification code generation algorithm is as follows:

1. Concatenate all hashes and document digests following the submission order to “iAM Smart” System.
i.e.: First Hash / Document Digest Bytes + Second Hash / Document Digest Bytes + Third Hash / Document Digest Bytes + ...
2. Calculate SHA-512 hash value on the output of combining concatenated value from Step 1 with SHA-512 hash value of Open ID (for non-anonymous digital signing).
i.e.: For non-anonymous: SHA-512(Concatenated Value from Step 1 + SHA-512(Open ID))
3. Calculate SHA-256 hash value on the output of hashing from Step 2.
i.e.: SHA-256(Output Value from Step 2)
4. Manipulate the output of hashing from Step 3 as follow to obtain the 6-digits identification code.
 - a) Divide the 256-bits SHA-256 hash value into 8 groups of 4-bytes data. Drop the last 2 groups of data (i.e., group 7 and 8).
 - b) Retain the only first byte for each group of 4-bytes data for the first 6 groups.
 - c) Perform 4-bits unsigned right shift on each byte.
 - d) Perform modulo 10 operations on each byte to get a value from 0 to 9.
 - e) Combine each 0 to 9 value resulted together to form the 6-digits identification code. The value from group 1 as the first digit, the value from group 2 as the second digit, and so on.



Pseudo code for generating 6-digit identification code

```
private static final char hexDigits[] = {'0', '1', '2', '3', '4',
'5', '6', '7', '8', '9'};
private static String get6DigitsSignCode(byte[] combineHash,
byte[] openID) {
    if (null == combineHash || null == openID) {
        return null;
    }
    char code[] = new char[6];
    byte[] source = new byte[combineHash.length +
openID.length];
    System.arraycopy(combineHash, 0, source, 0,
combineHash.length);
    System.arraycopy(openID, 0, source, combineHash.length,
openID.length);
    byte[] inData =
DigestUtil.digestToByte(DigestUtil.digestToByte(source,
Alg.SHA512), Alg.SHA256);
    try {
        for (int i = 0; i < 6; i++) {
            code[i] = hexDigits[(inData[i * 4] >>> 4 &
0xf) % 10]; // 无符号右移 4 位, 再取高位
        }
        return new String(code);
    } catch (Exception e) {
        log.error(e.getMessage());
        if (log.isDebugEnabled()) {
            log.debug(e.getMessage(), e);
        }
    }
    return null;
}
```

- Online Service Website/App should keep synchronising with Online Service server for the bulk digital signing result returned from “iAM Smart” System.
- API data decryption is required.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D71004	user is not allowed to sign	Inform user that “iAM Smart” bulk digital signing is not allowed for default “iAM Smart” and suggest user to upgrade to “iAM Smart” with bulk digital signing function
code - D71005	Inconsistent HKIC number	Hash of HKIC number provided is not matched with “iAM Smart” user. Check hash of HKIC number or inform user to authorise the digital signing using the correct “iAM Smart” account

Request and Response Parameters

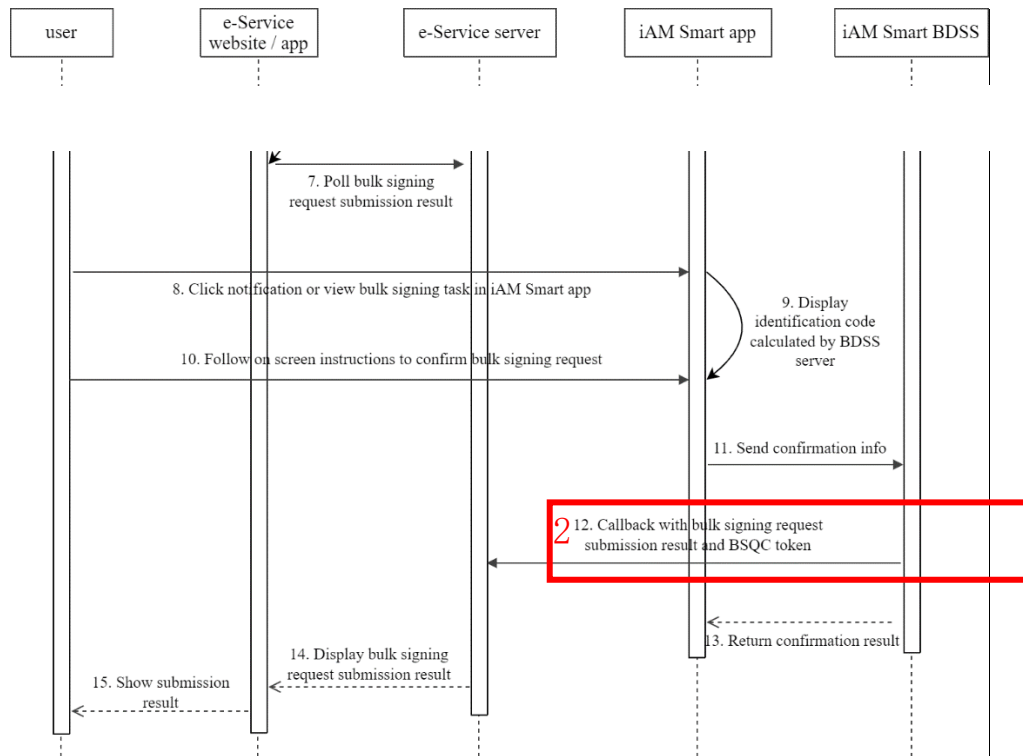
- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Request parameter “source”: Value should be matched with Online Service client terminal (e.g. “App_Scheme”/“App_Link” or browser’s user agent value).
- Request parameter “openID”: It is the Tokenised ID of the “iAM Smart” user. Online Service should ensure the accessToken and Tokenised ID are in pair and belongs to the target “iAM Smart” user for the request. The target “iAM Smart” user must have a sign version “iAM Smart”.
- Request parameter “redirectURI”: Callback URI for the endpoint that Online Service uses to receive BSQC Token. It must be in the same value as provided during Online Service registration.
- Request parameter “callbackResultURI”: Value should equal to URI of “Callback to Receive Bulk Digital Signing Result” Online Service callback API. It must be in the same value as provided during Online Service registration.
- Request parameter “hashCode”: It is the Document Hash to be signed. SHA-256 or equivalent is recommended.

- Request parameter “docDigest”: It is the PDF digest to be signed. SHA-256 or equivalent is recommended.
- Request parameter “sigAlgo”: It is the signature algorithm to be used by “iAM Smart”. The default value is "SHA256withRSA". While using "NONEwithRSA", hashCode provided must be hashed with SHA-256.
- Request parameter “HKICHash”: “iAM Smart” System will verify the HKIC hash of the “iAM Smart” user with the “HKICHash” provided by Online Service, and proceed the digital signing only when they matched. Online Service should convert the HKIC number into hash value using SHA-256 before sending to “iAM Smart” System. Only the identifier of HKIC number will be hashed, no check digit is needed.
- Request parameters “department”, “serviceName”, “documents”: They are the information of documents to be signed and will be shown in “iAM Smart” Mobile App for “iAM Smart” user's reference with the 6-digit identification code.
- Response parameter “authByQR”: The value returned by “iAM Smart” System will be “true” in this scenario.

3.11.1.2 Implementing (2) POST: Callback to Receive BSQC Token



Pre-conditions

- “iAM Smart” user accept the bulk digital signing request.
- “iAM Smart” user can also reject the bulk digital signing request.

Post-conditions

- API data decryption is required.
- Online Service server should match the callback result BSQC Token with corresponding Online Service Website/App using the “businessID”.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D71000	User cancelled signing request	Inform user the “iAM Smart” digital signing request is cancelled
code - D71001	User rejected signing request	Inform user the “iAM Smart” digital signing request is rejected

Error Code	Error Description	Suggested Action
code - D71002	Failed to request signing	Inform user the “iAM Smart” digital signing request is failed and retry later
code - D71003	Signing request timeout	Inform user the “iAM Smart” digital signing request is timeout and provide way for user to retry
code - D71005	inconsistent HKIC number	Hash of HKIC number provided is not matched with “iAM Smart” user. Check hash of HKIC number or inform user to authorise the digital signing using the correct “iAM Smart” account

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Callback parameter “businessID”: It is a unique identifier for Online Service to differentiate different request. Online Service can use the businessID to relate the BSQC Token with the original request.
- Callback parameter “state”: 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.
- Callback parameter “signature”: It can be used by the corresponding Online Service to enquire the status of the submitted bulk digital signing request or cancel the corresponding request.

3.11.2 Scenario 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.

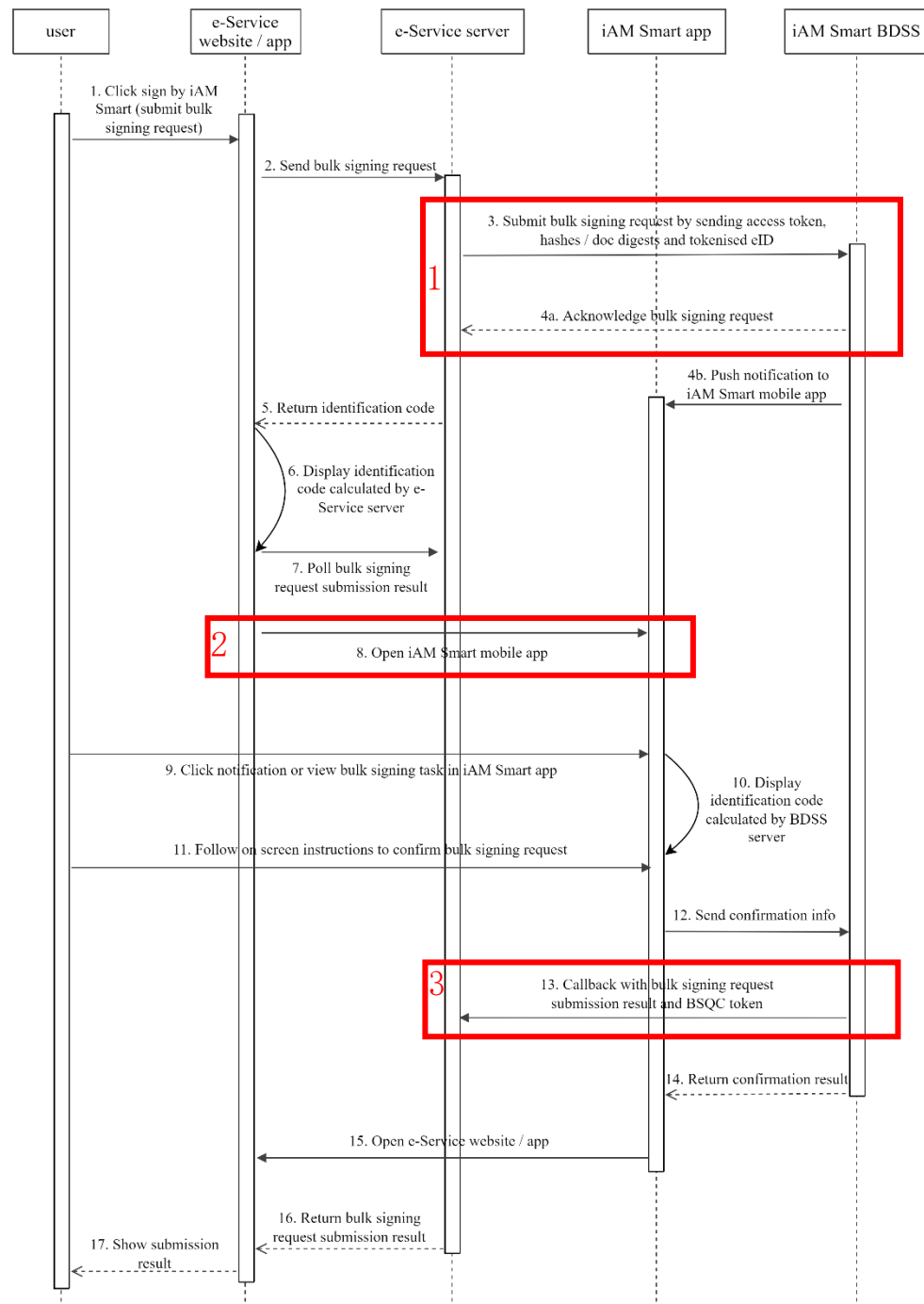


Figure-28 Bulk Digital Signing with Service Login (Online Service Website/App in Same Device)

Step 1. User clicks the “Sign by iAM Smart” button in Online Service Website/App;

Step 2-3. Online Service initiates bulk digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, accessToken, openID (Tokenised ID), source, redirectURI, callbackResultURI, state, HKICHash, department, serviceName, requestName, maxCallbackSigs, documents, etc.

The request parameter “documents” contains the Document Hash (hashCode) / PDF digest (docDigest) of all documents, signature algorithm (only for document). The request parameter “source” will be the browser's user agent value (for Online Service Website) or “App_Scheme”/“App_Link” (for Online Service App) in this scenario. API encryption is required;

“iAM Smart” API (POST: Request Bulk Signing)

Step 4a. “iAM Smart” System verifies the validity of accessToken, Tokenised ID, other parameters and confirms receipt of the digital signing request to Online Service server by returning POST response with parameter “authByQR” (set to “false”) in this scenario;

Step 4b. “iAM Smart” System pushes a notification message to the “iAM Smart” Mobile App based on the Tokenised ID;

Step 5-6. Online Service server concatenates all document hashes and PDF digests and hash of Tokenised ID to calculate a 6-digit identification code and instructs Online Service Website/App to display the instructions with the 6-digit identification code to inform “iAM Smart” user to process the signing authorisation request in “iAM Smart” Mobile App;

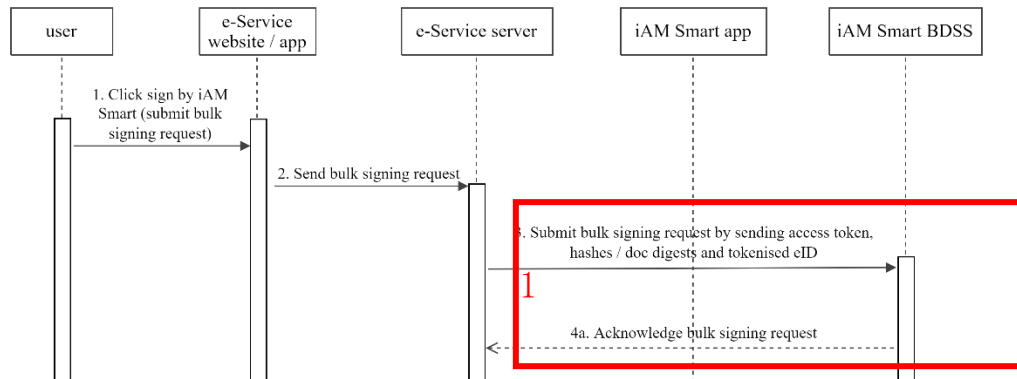
Step 7. Online Service Website/App should keep synchronising with Online Service server for the digital signing processing result (e.g. polling);

Step 8. Online Service Website invokes “iAM Smart” Mobile App using URL Scheme with “ticketID”. Other parameters of this API are not used in this scenario;

“iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Getting Context)

- Step 9. “iAM Smart” user opens “iAM Smart” Mobile App, reviews the digital signing authorisation request (e.g. continue or reject the request).
- Step 10. “iAM Smart” user verifies the 6-digit identification code with Online Service Website/App;
- Step 11-12. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 13. “iAM Smart” BDSS invokes Online Service callback API to return the BSQCToken corresponding with openID of the digital signing request. API data encryption by AES;
- “iAM Smart” API (POST: Callback to Receive BSQC Token)*
- Step 14. “iAM Smart” BDSS saves the signing request to MQ queue and returns the result with the same “businessID” of the signing request.
- Step 15. “iAM Smart” BDSS instructs “iAM Smart” Mobile App to invoke the Online Service App using URL Scheme, or Universal Link/App Link (“iAM Smart” System queries the information registered at Online Service registration depending on the “source” submitted in Step 3);
- Step 16-17. Online Service server returns the submission result (status of Pending for Signing) to Online Service Website/App, and then Online Service Website/App returns the result to user. Response to Step 7.

3.11.2.1 Implementing (1) POST: Request Bulk Digital Signing



Pre-conditions

- Please refer to Section 3.7.1.1 except:
 - Online Service Website/App and “iAM Smart” Mobile App are in the same device in this scenario.

Post-conditions

- Please refer to Section 3.7.1.1 except:
 - Response “authByQR” is “false”, “ticketID” will be provided by “iAM Smart” System in this scenario.

Error conditions

- Please refer to Section 3.7.1.1.

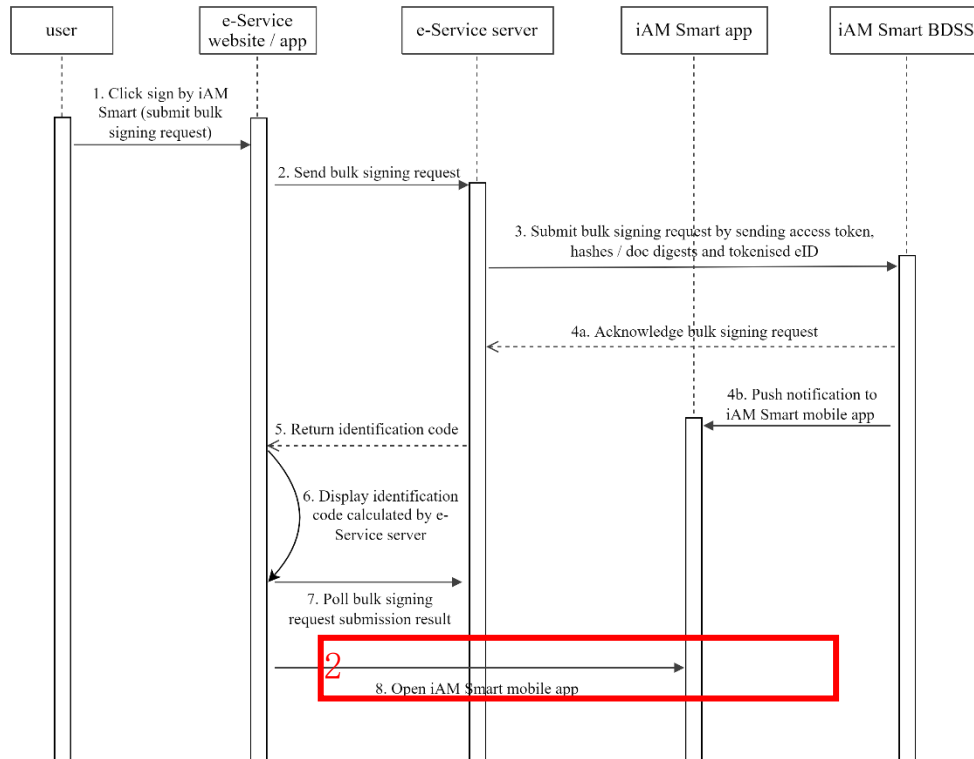
Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Please refer to Section 3.7.1.1, except for the following parameters:
 - Response parameter “authByQR”: The value is “false” in this scenario.
 - Response parameter “ticketID”: It will be provided by “iAM Smart” System in this scenario. It is a 36-byte (or less) UUID number (ASCII character set).

3.11.2.2 Implementing (2) URL Scheme: *Open “iAM Smart” Mobile App for Getting Context*



Pre-conditions

- Online Service Website and “iAM Smart” Mobile App are in the same device.
- Online Service has the “ticketID” provided by “iAM Smart” System for the digital signing request;
- Other parameters of this API are not used in this scenario.

Post-conditions

- “iAM Smart” Mobile App will be launched.
- Online Service Website should keep synchronising with Online Service server for the callback response of the digital signing request from “iAM Smart” System.

Error conditions

- Nil

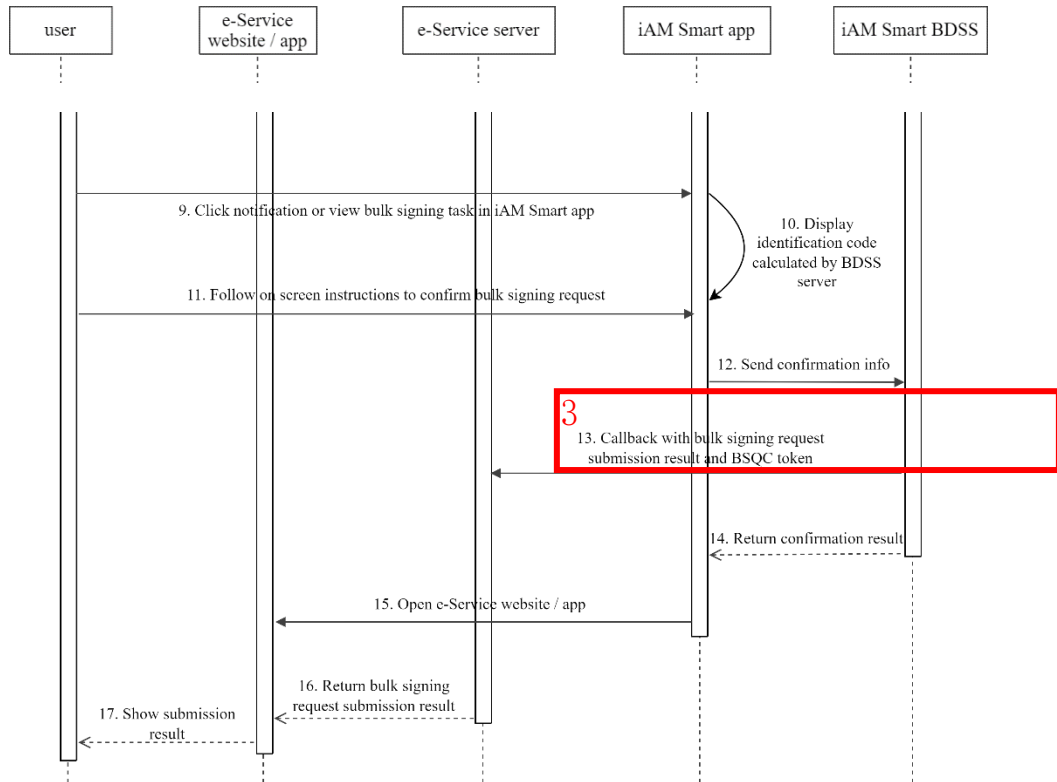
Request Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- Nil

3.11.2.3 Implementing (3) POST: *Callback to Receive BSQC Token*

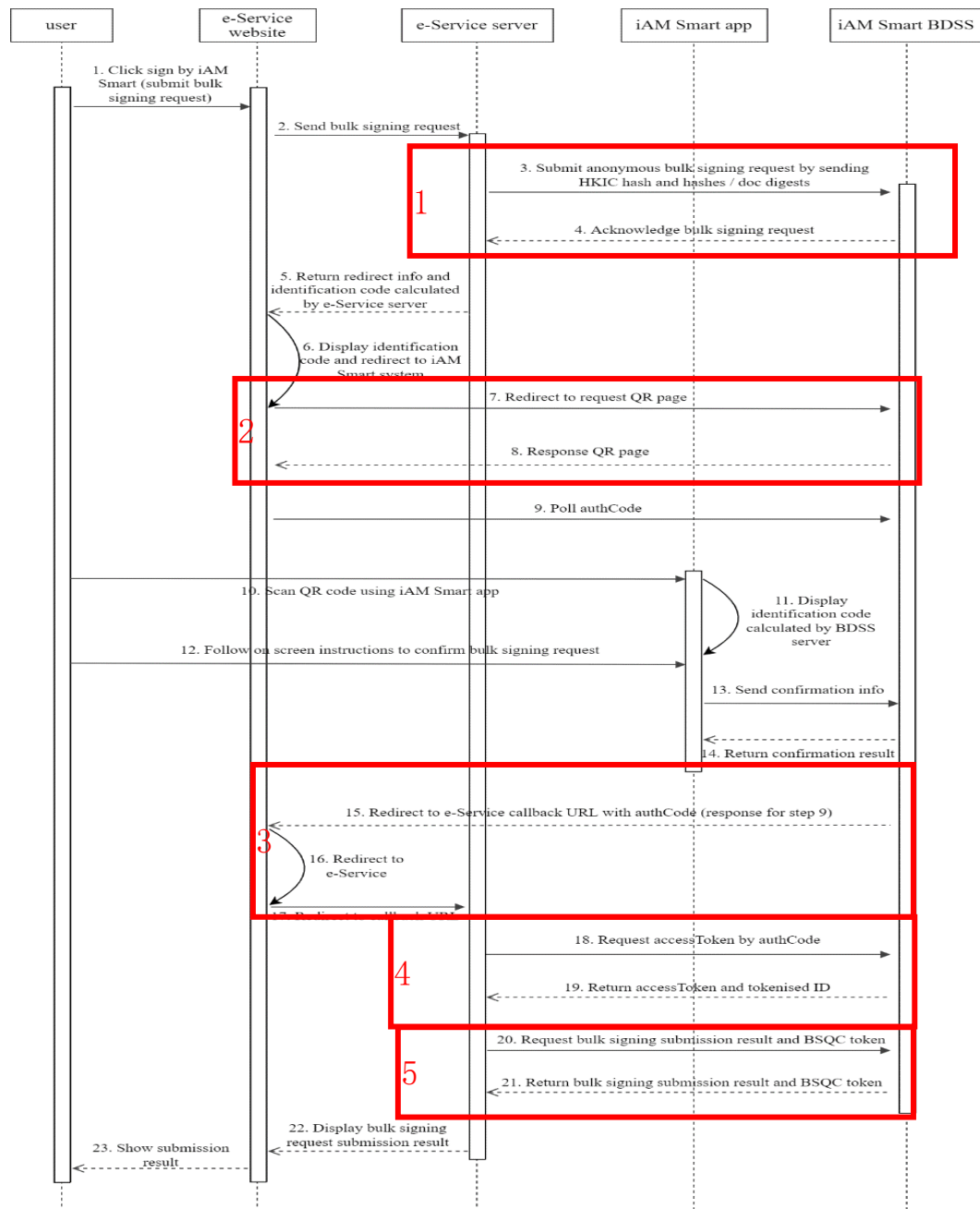


Please refer to Section 3.11.1.2.

3.12 WORKFLOWS FOR BULK DIGITAL SIGNING WITHOUT SERVICE LOGIN (AKA ANONYMOUS BULK DIGITAL SIGNING)

3.12.1 Scenario 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 different devices.



- Step 1. After entering HKIC number and choose documents, the user clicks the “Anonymous bulk digital signing” button in Online Service Website;
- Step 2. Online Service Website initiates anonymous bulk digital signing request to Online Service server with browser's user agent name (use as value for request parameter “source”);
- Step 3. Online Service server initiates an anonymous bulk digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, Document Hash / PDF digest, HKICHash, signature algorithm (only for hash document), etc. API encryption is required;
- ”iAM Smart” API (POST: Request Anonymous Bulk Digital Signing)*
- Step 4. “iAM Smart” BDSS verifies and confirms receipt of the anonymous bulk digital signing request to online service Server by returning POST response with parameter “ticketID”;
- Step 5-7. Online Service Server uses the Document Hash / PDF digest, HKICHash and hash of clientID to calculate a 6-digit identification code and instructs Online Service Website to display the instructions with the 6-digit identification code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;
- At the same time, the page polls Online Service for the submission status at step 6, Online Service server prepares the request parameters such as “clientID”, “redirectURI”, “source” (set as browser’s user agent value), “scope”, “ticketID”, etc. and constructs the GET request to invoke the “iAM Smart” API using browser redirection;
- ”iAM Smart” API (GET: Request QR Page)*
- Step 8. “iAM Smart” System returns the broker page (if Online Service has set “brokerPage” to “true”) and after the broker page fails to find the “iAM Smart” Mobile App, it will request QR page to be displayed in browser. If Online Service does not request for broker page (i.e. no broker page will be returned), the browser will directly display QR Code page;

- Step 9. QR Code page of “iAM Smart” System polls “iAM Smart” System for the QR Code status;
- Step 10. “iAM Smart” user logs in “iAM Smart” Mobile App to scan the QR Code;
- Step 11. “iAM Smart” Mobile App requests and displays 6-digit identification code from “iAM Smart” BDSS. “iAM Smart” user reviews the signing authorisation request (e.g. continue or reject the request) and verifies the 6-digit identification code with Online Service Website;
- Step 12. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 13-14. “iAM Smart” BDSS verifies the validity of QR Code and other necessary information, and return the authorisation result to “iAM Smart” Mobile App;
- Step 15-17. QR Code page of “iAM Smart” System receives the polling result returned by “iAM Smart” BDSS (i.e. response for the polling in Step 9) which includes authCode and businessID or any error code (e.g. “iAM Smart” user rejects the request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

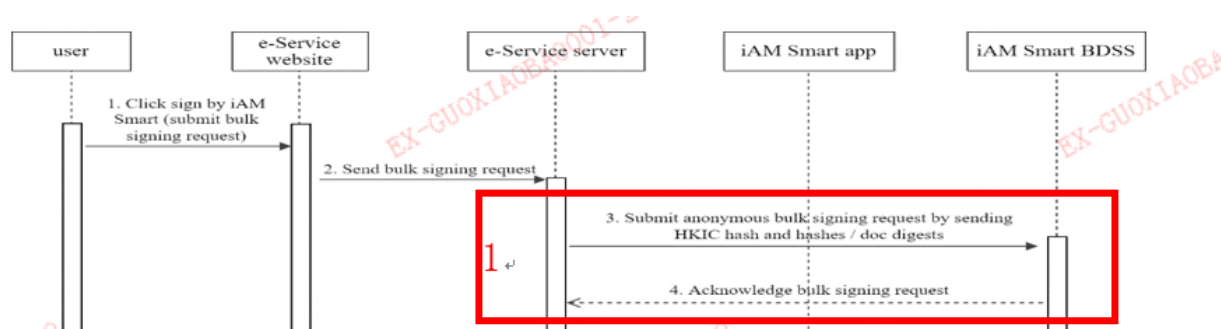
- Step 18-19. When Online Service server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e. openID) of the “iAM Smart” user. API data encryption is required;

“iAM Smart” API (POST: Request accessToken and Tokenised ID)

- Step 20-21. Online Service server invokes the “iAM Smart” API with the accessToken and openID to obtain BSQCToken, then Online Service invokes the “iAM Smart” API with BSQCToken and openID to obtain the bulk digital signing result. API data encryption is required;

“iAM Smart” API (POST: Request BSQC Token)

3.12.1.1 Implementing (1) POST: Request Anonymous Bulk Digital Signing



Pre-conditions

- Online Service Website/App and “iAM Smart” Mobile App are in different devices in this scenario.
- The “iAM Smart” user who signs the document must have a sign version “iAM Smart”.
- Online Service generates a “businessID” for this bulk signing request and uses this identifier to match the callback result returned from “iAM Smart” BDSS.
- Online Service generates the Document Hash as “hashCode” parameter from the original documents to be signed using a hash algorithm that can suit the specific business need. It is recommended to use a hash algorithm that is at least SHA-256 or equivalent, or Online Service generates the Document Hash as “docDigest” parameter from the original documents to be signed using the Adobe.PPKLite filter and the adbe.pkcs7.detached subfilter.
- Online Service should convert the HKIC number (no check digit is needed) into hash value using SHA256.
- API data encryption is required.

Post-conditions

- Online Service server should check the response parameter “ticketID” and determine the next action. “ticketID” will not be provided by “iAM Smart” System in this scenario.
- Online Service has to calculate a 6-digit identification code using the Document Hash and Tokenised ID hash value. Online Service Website/App should show the 6-digit identification code and instruction to inform “iAM Smart” user to process the bulk digital signing request in “iAM Smart” Mobile App when QR Code is scanned.
- 6-digit identification code generation algorithm refer to Section 3.11.1.1.
- Online Service Website/App should keep synchronising with Online Service server for the bulk digital signing result returned from “iAM Smart” BDSS.

- API data decryption is required.

Error conditions

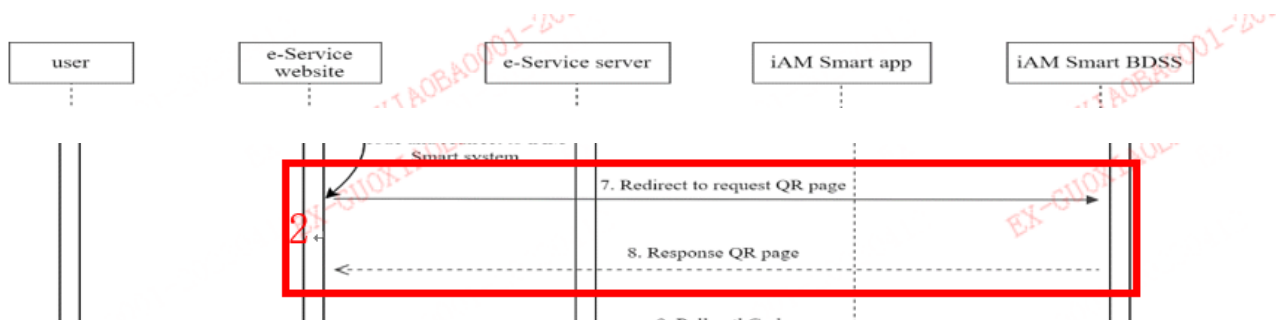
- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D70002	Failed to request signing	Inform user the “iAM Smart” digital signing request is failed and retry later

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

3.12.1.2 Implementing (2) GET: Request QR Page



Pre-conditions

- Please refer to Section 3.4.1.1 except:
 - Online Service has the “ticketID” provided by “iAM Smart” System for the anonymous request in step 4.

Post-conditions

- Please refer to Section 3.4.1.1.

Error conditions

- Please refer to Section 3.4.1.1.

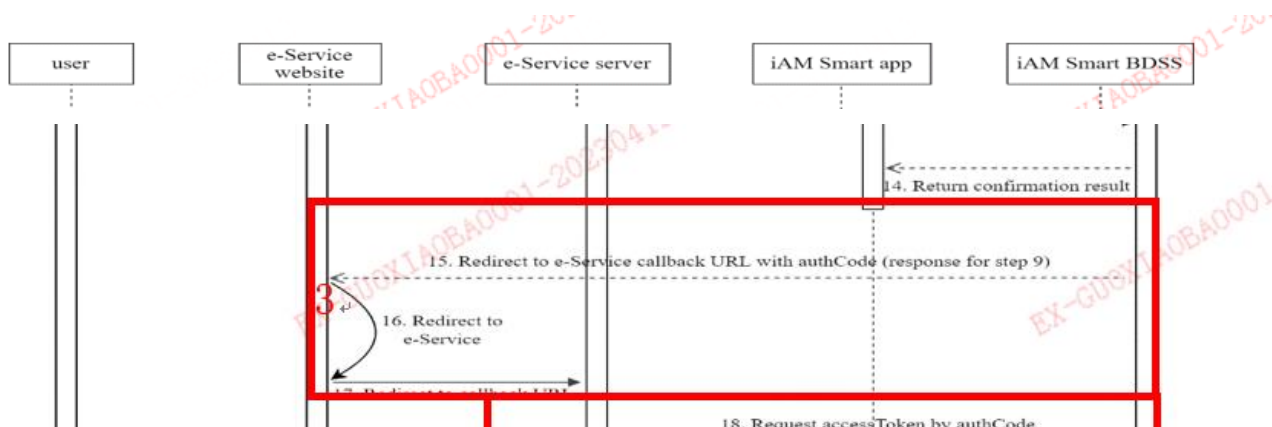
Request Parameters

- Please refer to Section 3.4.1.1.

Notes

- Please refer to Section 3.4.1.1 except the following parameter:
 - Request parameter “ticketID”: Value provided by “iAM Smart” System for the anonymous bulk digital signing request.

3.12.1.3 Implementing (3) GET: Callback with authCode to Online Service Server



Pre-conditions

- Please refer to Section 3.4.1.2.

Post-conditions

- Please refer to Section 3.4.1.2 except:
 - Online Service server should match the callback result with corresponding Online Service client terminal using the “businessID”.

Error conditions

- This Online Service callback API is a HTTP GET request. If parameter “error_code” is returned, it means the request is failed.

Error Code	Error Description	Suggested Action
error_code - D71001	User rejected signing request	Inform user the digital signing request is rejected
error_code - D71002	Failed to request signing	Inform user the digital signing request is failed and retry later
error_code - D71004	User not allowed to sign	Inform user that “iAM Smart” digital signing is not allowed for default “iAM Smart” and suggest user to upgrade to “iAM Smart” with digital signing function
error_code - D71005	Inconsistent HKIC number	Hash of HKIC number provided is not matched with “iAM Smart” user. Check hash of HKIC number or inform user to authorise the digital signing using the correct “iAM Smart” account

The error_code D71003 (signing request timeout) does not appear in this scenario. For the QR Code timeout or request confirmation timeout in the “iAM Smart” Mobile App, message will be prompted in the corresponding user interface.

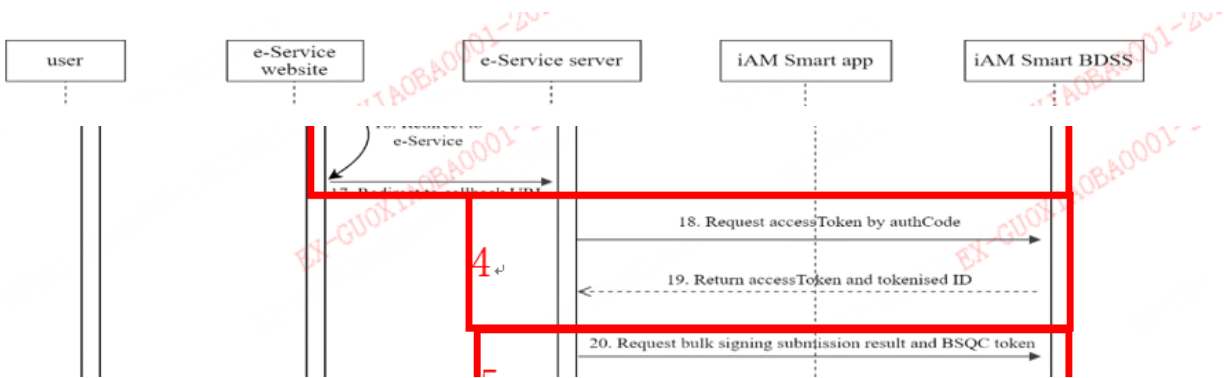
Callback Parameters

- Please refer to Section 3.4.1.2.

Notes

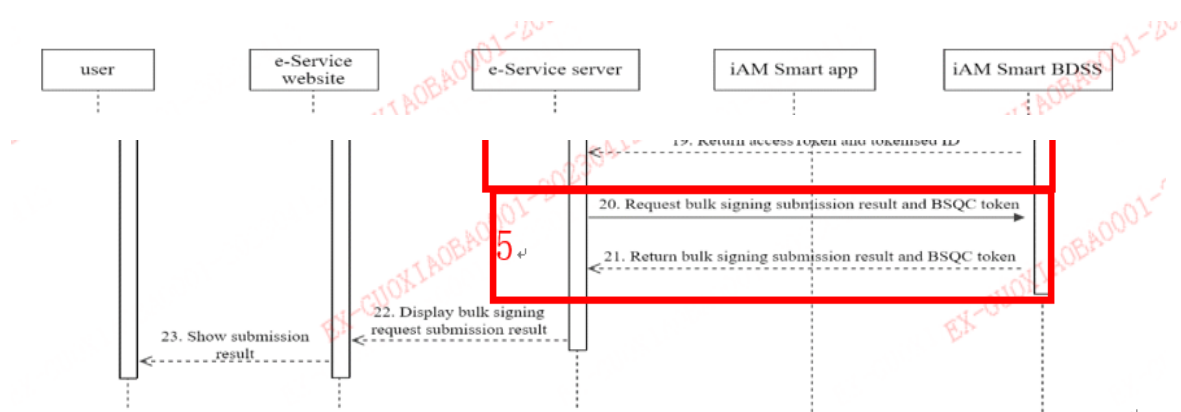
- Please refer to Section 3.4.1.2 except the following parameter:
 - Callback parameter “businessID”: Value returned by “iAM Smart” System should be the same one as Online Service submitted in the anonymous request.

3.12.1.4 Implementing (4) POST: Request accessToken and Tokenised ID



Please refer to Section 3.6.1.4

3.12.1.5 Implementing (5) POST: Request BSQC Token



Pre-conditions

- Online Service provides parameters “accessToken” and “openID”.
- API data encryption is required.

Post-conditions

- Online Service has obtained “accessToken” and “openID” by previous request

Error conditions

Error Code	Error Description	Suggested Action
error_code - D20012	insufficient scope	Check authorisation scope(s) requested is/are sufficient for invoking the corresponding “iAM Smart” API

Request Parameters

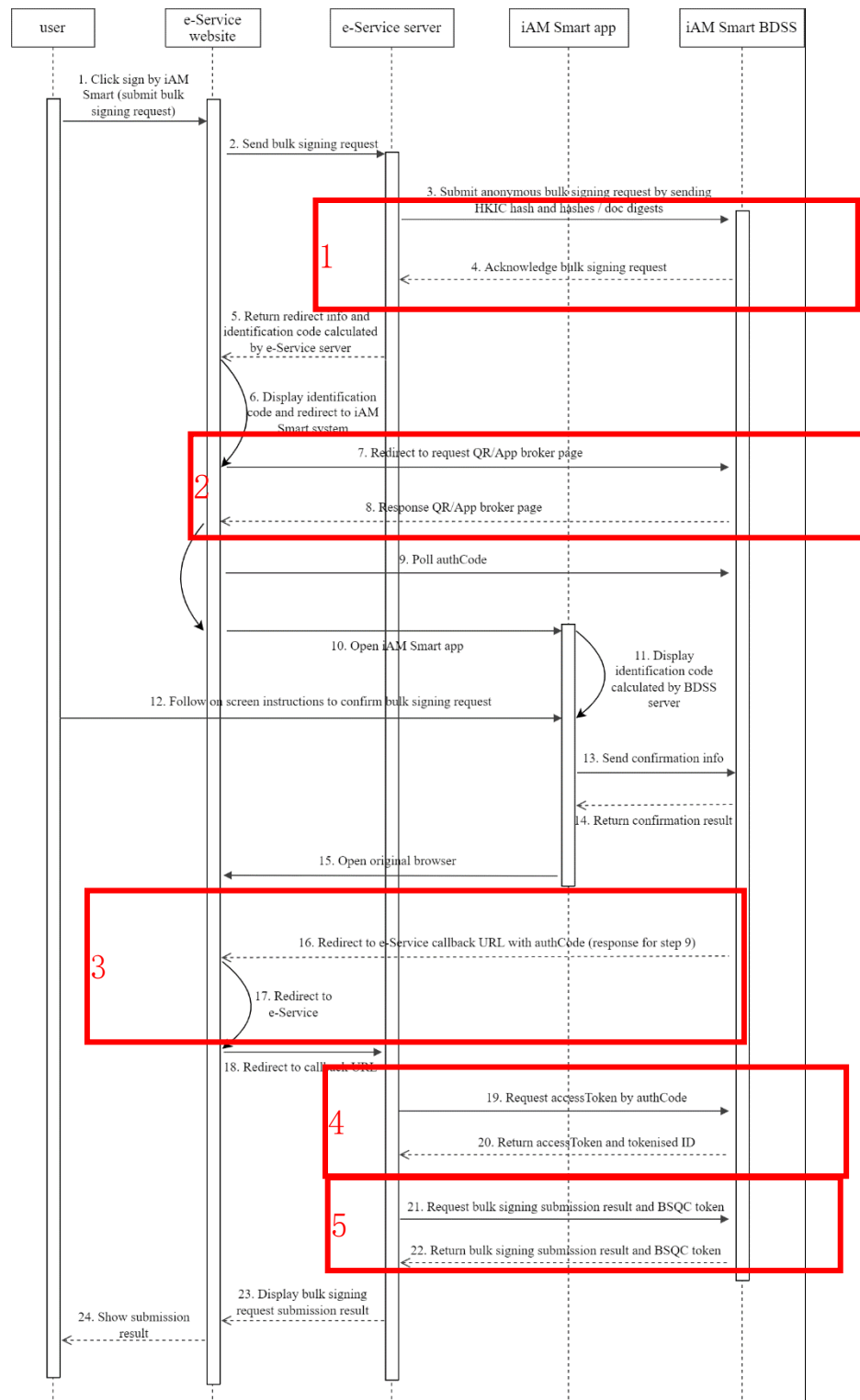
Parameter	Type	Presence	Description
accessToken	String	Required	accessToken value
openID	String	Required	Tokenised ID value

Response Parameters

Parameter	Type	Presence	Description
BSQCToken	String	Required	BSQCToken can be used by the corresponding Online Service to enquire or cancel the submitted bulk digital signing request

3.12.2 Scenario 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.



- Step 1. After entering HKIC number and choose documents, the user clicks the “Anonymous bulk digital signing” button in Online Service Website;
- Step 2. Online Service Website initiates anonymous bulk digital signing request to Online Service server with browser's user agent name (use as value for request parameter “source”);
- Step 3. Online Service server initiates an anonymous bulk digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, Document Hash / PDF digest, HKICHash, signature algorithm (only for hash document), etc. API encryption is required;
- ”iAM Smart” API (POST: Request Anonymous Bulk Digital Signing)*
- Step 4. “iAM Smart” BDSS verifies and confirms receipt of the anonymous bulk digital signing request to Online Service server by returning POST response with parameter “ticketID”;
- Step 5-7. Online Service server uses the Document Hash / PDF digest, HKICHash and hash of clientID to calculate a 6-digit identification code and instructs Online Service Website to display the instructions with the 6-digit identification code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;
- At the same time, the page polls Online Service for the submission status at step 6, Online Service server prepares the request parameters such as “clientID”, “redirectURI”, “source” (set as browser’s user agent value), “scope”, “ticketID”, etc. and constructs the GET request to invoke the “iAM Smart” API using browser redirection;
- ”iAM Smart” API (GET: Request QR Page)*
- Step 8. “iAM Smart” System returns the broker page to the Online Service Website;
- Step 9. Broker page of “iAM Smart” System polls “iAM Smart” System for further action;
- Step 10. Broker page invokes the “iAM Smart” Mobile App in the same device automatically and sends it the relevant parameters.

- Step 11. “iAM Smart” Mobile App requests and displays 6-digit identification code from “iAM Smart” BDSS. “iAM Smart” user reviews the signing authorisation request (e.g. continue or reject the request) and verifies the 6-digit identification code with Online Service Website;
- Step 12. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 13-15. “iAM Smart” BDSS verifies the validity of QR Code and other necessary information, and returns the authorisation result to “iAM Smart” Mobile App, “iAM Smart” Mobile App invokes the original Online Service browser;
- Step 16-18. Broker page of “iAM Smart” System receives the polling result returned by “iAM Smart” BDSS (i.e. response for the polling in Step 9) which includes authCode and businessID or any error code (e.g. “iAM Smart” user rejects the request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

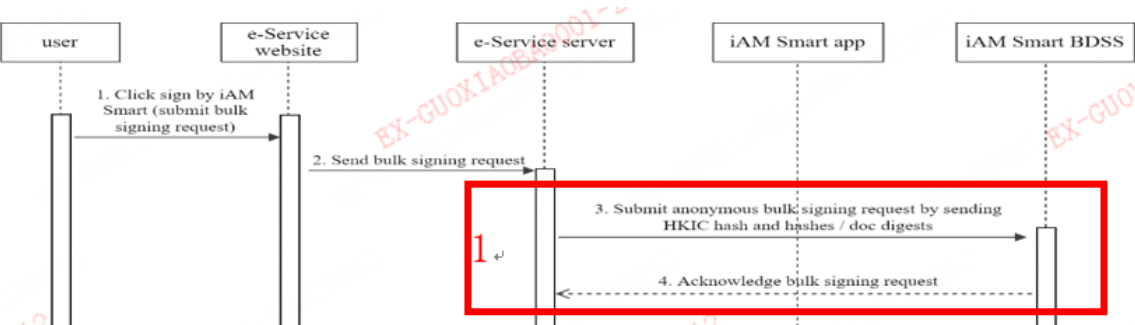
- Step 18-19. When Online Service server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invokes the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e. openID) of the “iAM Smart” user. API data encryption is required;

“iAM Smart” API (POST: Request accessToken and Tokenised ID)

- Step 20-21. Online Service server invokes the “iAM Smart” API with the accessToken and openID to obtain BSQCToken, then Online Service invokes the “iAM Smart” API with BSQCToken and openID to obtain the bulk digital signing result. API data encryption is required;

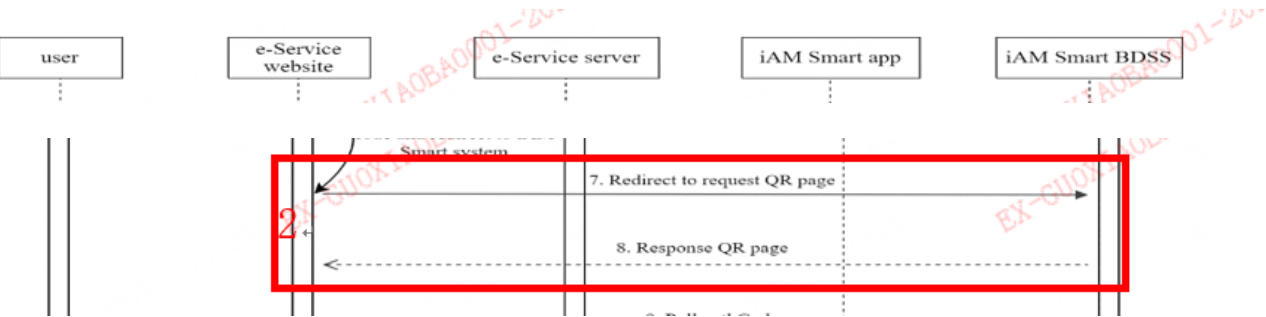
“iAM Smart” API (POST: Request BSQC Token)

3.12.2.1 Implementing (1) POST: Request Anonymous Bulk Digital Signing



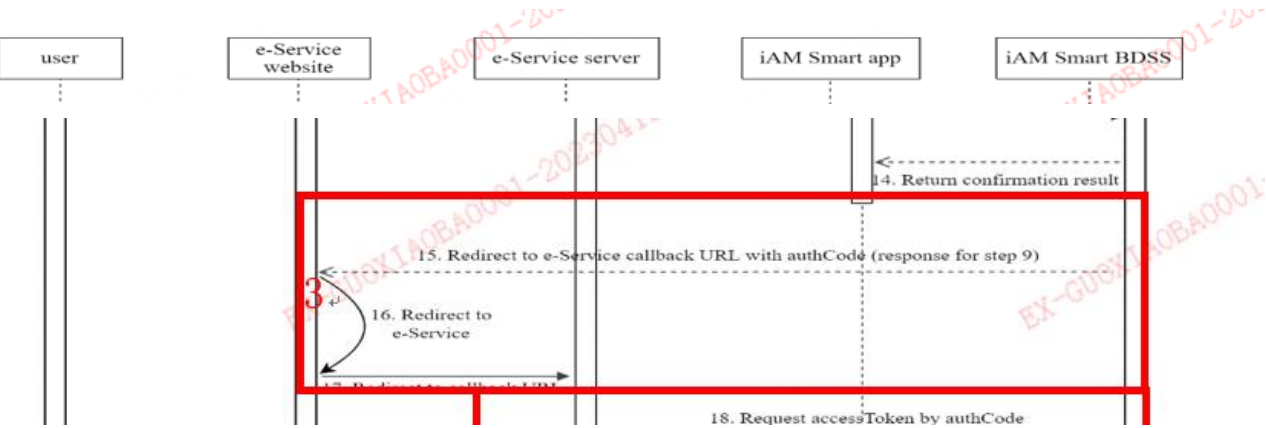
Please refer to Section 3.12.1.1

3.12.2.2 Implementing (2) GET: Request QR Page



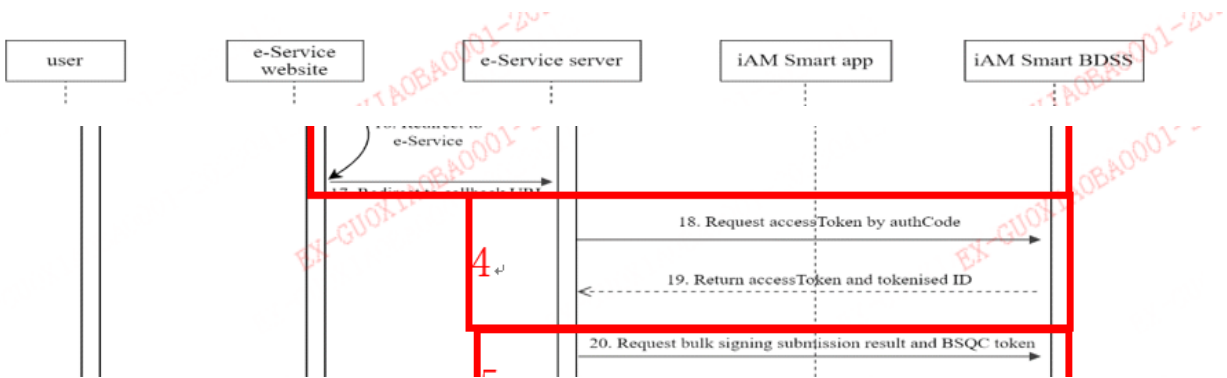
Please refer to Section 3.12.1.2

3.12.2.3 Implementing (3) GET: Callback with authCode to Online Service Server



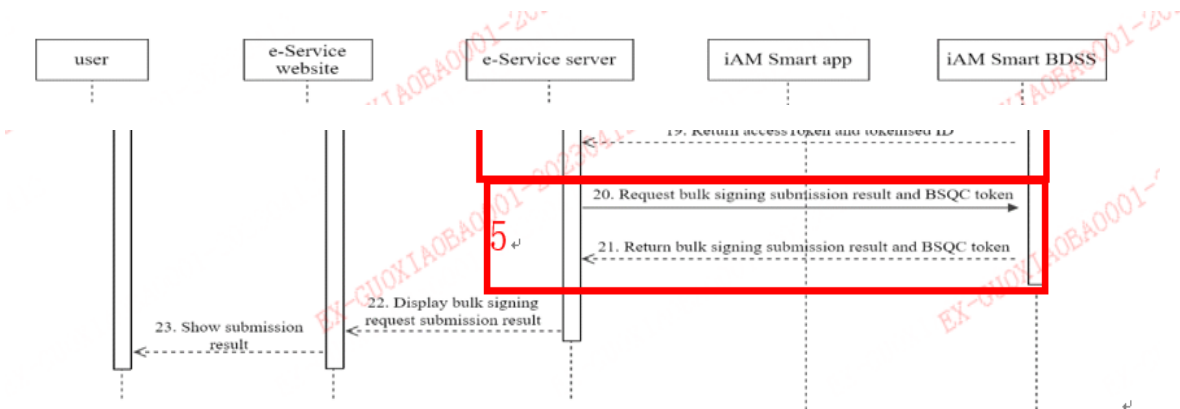
Please refer to Section 3.12.1.3

3.12.2.4 Implementing (4) POST: Request accessToken and Tokenised ID



Please refer to Section 3.6.1.4

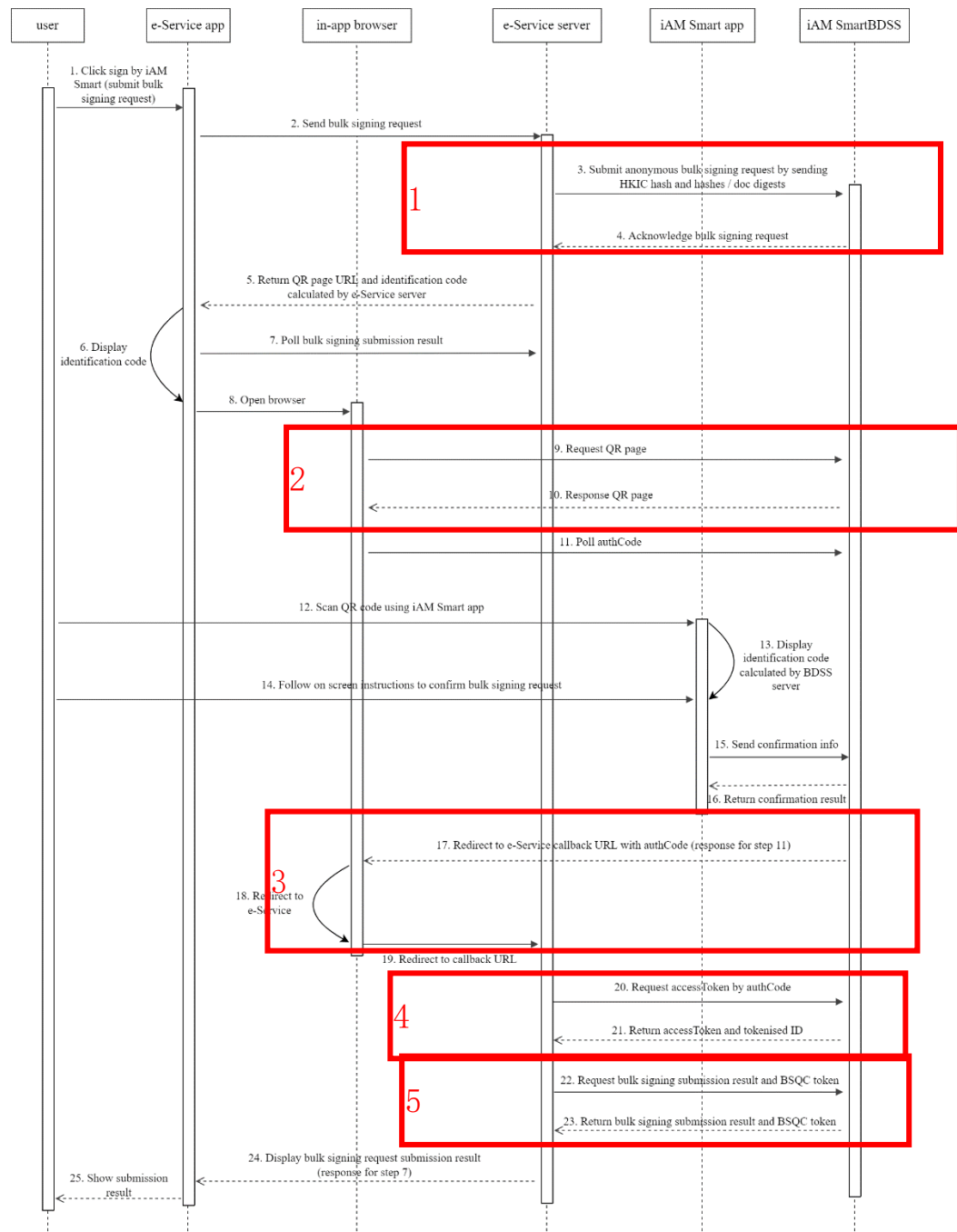
3.12.2.5 Implementing (5) POST: Request BSQC Token



Please refer to Section 3.12.1.5

3.12.3 Scenario 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.



- Step 1. After entering HKIC number and choose documents, the user clicks the “Anonymous bulk digital signing” button in Online Service Website;
- Step 2. Online Service Website initiates anonymous bulk digital signing request to Online Service server with browser's user agent name (use as value for request parameter “source”);
- Step 3. Online Service server initiates an anonymous bulk digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, Document Hash / PDF digest, HKICHash, signature algorithm (only for hash document), etc. API encryption is required;
- ”iAM Smart” API (POST: Request Anonymous Bulk Digital Signing)*
- Step 4. “iAM Smart” BDSS verifies and confirms receipt of the anonymous bulk digital signing request to Online Service server by returning POST response with parameter “ticketID”;
- Step 5-6. Online Service server uses the Document Hash / PDF digest, HKICHash and hash of clientID to calculate a 6-digit identification code and instructs Online Service Website to display the instructions with the 6-digit identification code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;
- At the same time, the page polls Online Service for the submission status at step 6, Online Service server prepares the request parameters such as “clientID”, “redirectURI”, “source” (set as browser’s user agent value), “scope”, “ticketID”, etc. and constructs the GET request to invoke the “iAM Smart” API using browser redirection;
- “iAM Smart” API (GET: Request QR Page)*
- Step 7. Online Service App polls Online Service server for the digital signing result from “iAM Smart” BDSS;
- Step 8. Online Service App invokes in-app browser (Safari for iOS, Chrome for Android) to submit the GET request;
- Step 9-10. Browser opens the GET request to invoke the “iAM Smart” API and QR Code page will be shown;

“iAM Smart” API (GET: Request QR page)

- Step 11. QR Code page of “iAM Smart” System polls “iAM Smart” System for the QR Code status;
- Step 12. “iAM Smart” user logs in “iAM Smart” Mobile App to scan the QR Code;
- Step 13. “iAM Smart” Mobile App requests and displays 6-digit identification code from “iAM Smart” BDSS. “iAM Smart” user reviews the signing authorisation request (e.g. continue or reject the request) and verifies the 6-digit identification code with Online Service Website;
- Step 14. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 15-16. “iAM Smart” BDSS verifies the validity of QR Code and other necessary information, and return the authorisation result to “iAM Smart” Mobile App;
- Step 17-19. QR Code page of “iAM Smart” System receives the polling result returned by “iAM Smart” System (i.e. response for the polling in Step 9) which includes authCode and businessID or any error code (e.g. “iAM Smart” user rejects the request), assembles and invokes the Online Service callback API given in “redirectURI” using browser redirection;

Online Service Callback API (GET: Callback with authCode to Online Service Server)

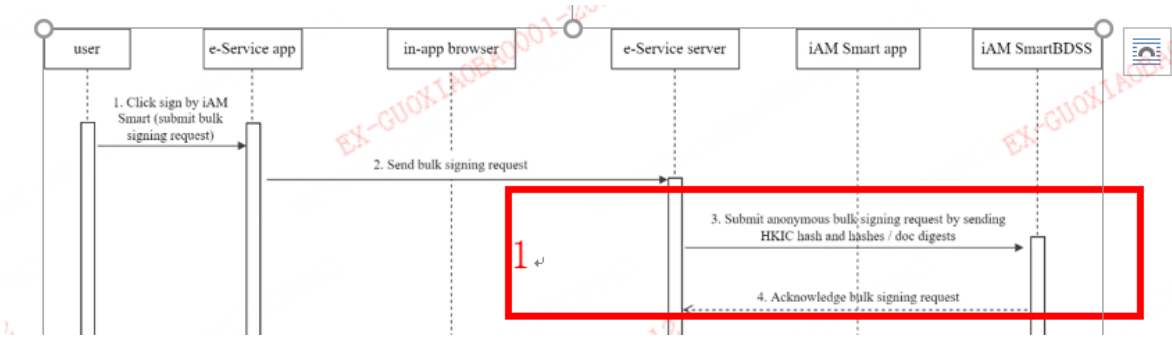
- Step 20-21. When Online Service server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invokes the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e. openID) of the “iAM Smart” user. API data encryption is required;

“iAM Smart” API (POST: Request accessToken and Tokenised ID)

- Step 22-23. Online Service server invokes the “iAM Smart” API with the accessToken and openID to obtain BSQCToken, then Online Service invokes the “iAM Smart” API with BSQCToken and openID to obtain the bulk digital signing result. API data encryption is required;

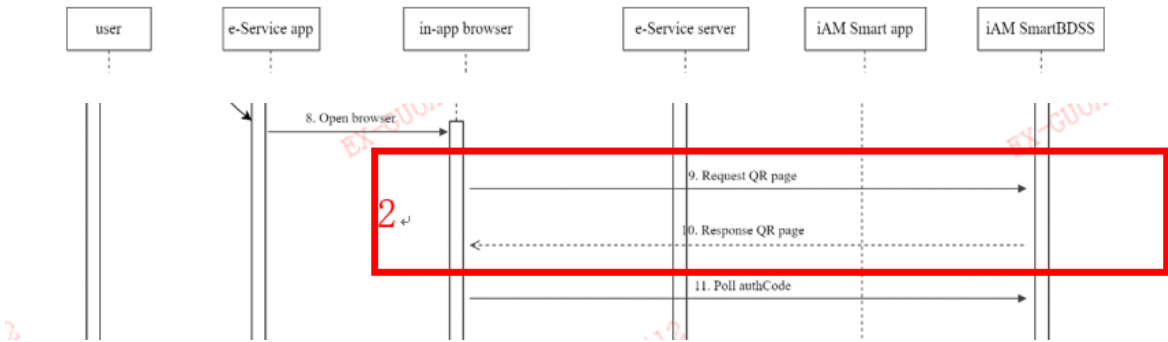
"iAM Smart" API (POST: Request BSQC Token)

3.12.3.1 Implementing (1) POST: Request Anonymous Bulk Digital Signing



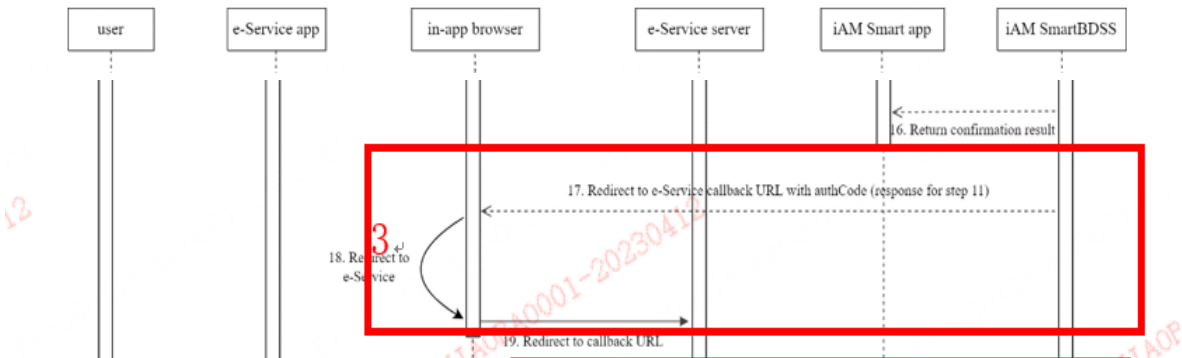
Please refer to Section 3.12.1.1

3.12.3.2 Implementing (2) GET: Request QR Page



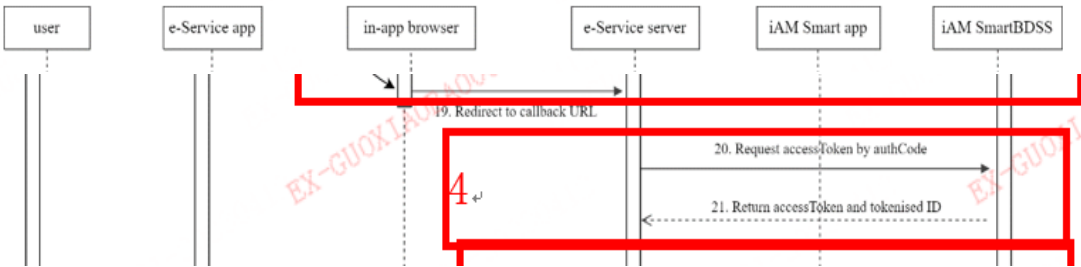
Please refer to Section 3.12.1.2

3.12.3.3 Implementing (3) GET: Callback with authCode to Online Service Server



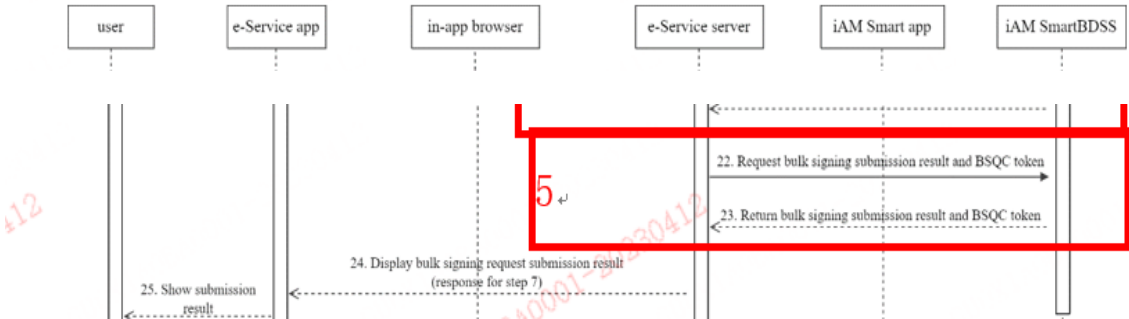
Please refer to Section 3.12.1.3

3.12.3.4 Implementing (4) POST: Request accessToken and Tokenised ID



Please refer to Section 3.6.1.4

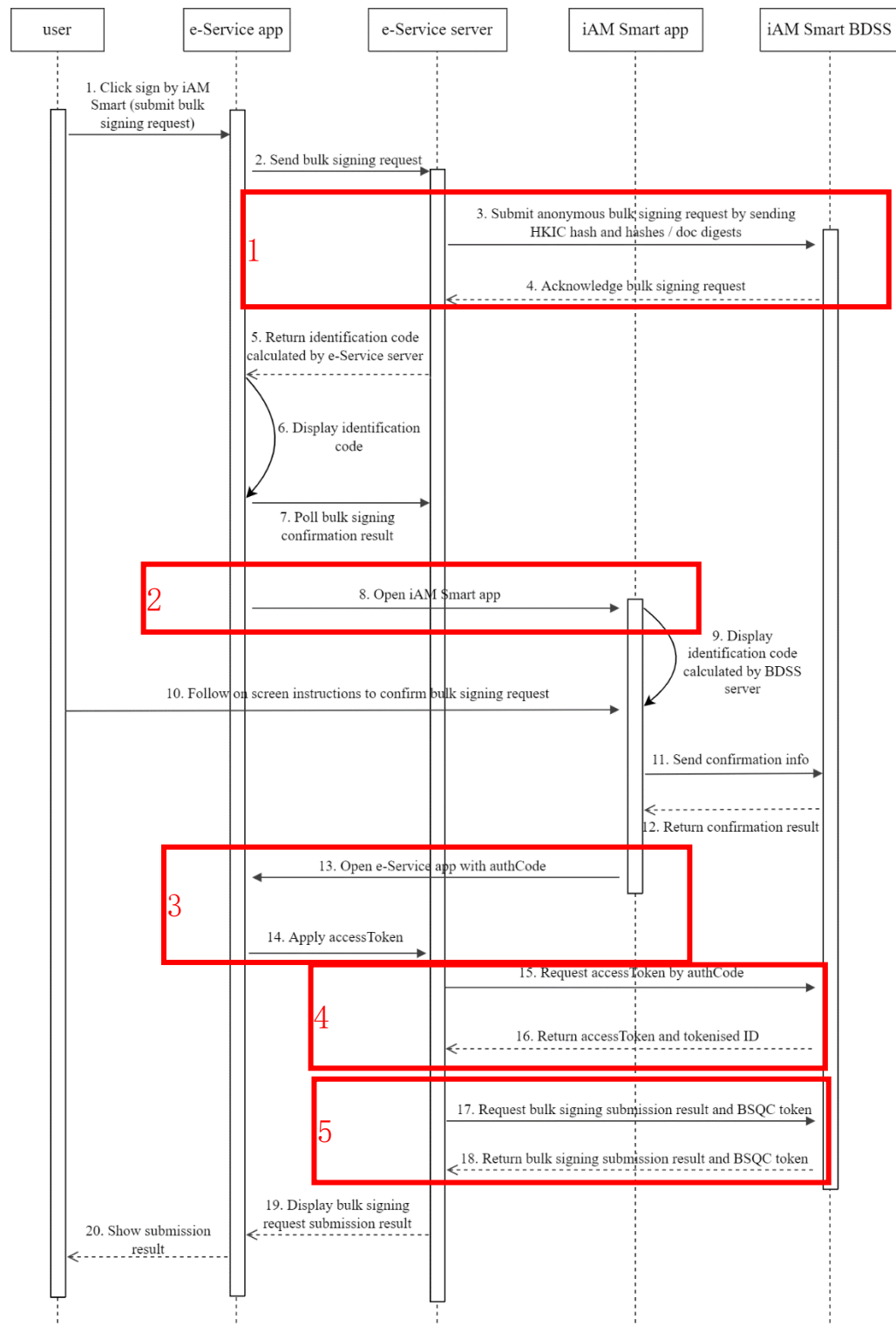
3.12.3.5 Implementing (5): POST: Request BSQC Token



Please refer to Section 3.12.1.5

3.12.4 Scenario 4: Anonymous Bulk Digital Signing (Online Service App in Same Device)

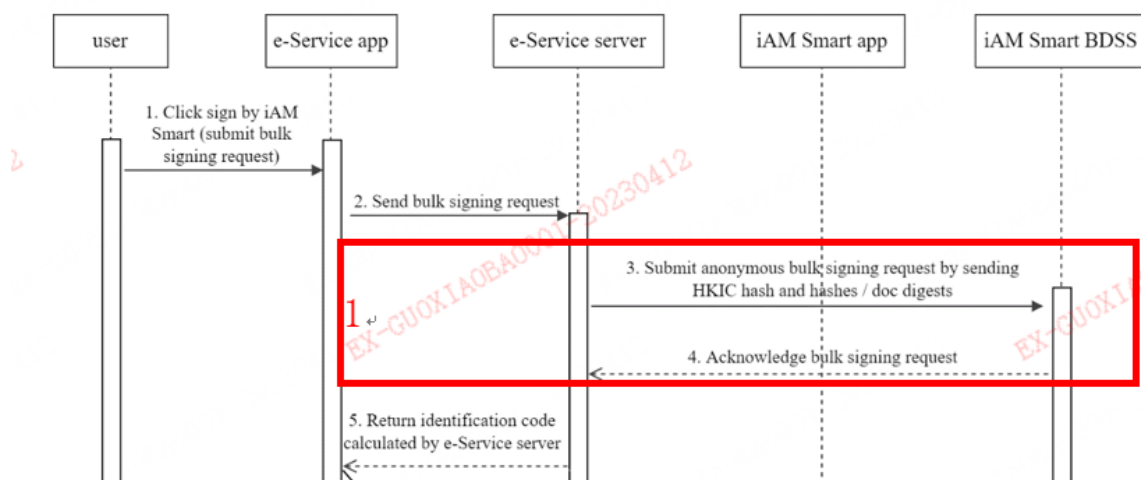
The sequence diagram below shows how an anonymous user authorises and signs multiple document hashes and/or digests when Online Service App and the “iAM Smart” Mobile App are running in the same device.



- Step 1. After entering HKIC number and choose documents, the user clicks the “Anonymous bulk digital signing” button in Online Service Website;
- Step 2. Online Service Website initiates anonymous bulk digital signing request to Online Service server with browser's user agent name (use as value for request parameter “source”);
- Step 3. Online Service server initiates an anonymous bulk digital signing request to invoke the “iAM Smart” API with the “businessID” of this request, Document Hash / PDF digest, HKICHash, signature algorithm (only for hash document), etc. API encryption is required;
- ”iAM Smart” API (POST: Request Anonymous Bulk Digital Signing)*
- Step 4. “iAM Smart” BDSS verifies and confirms receipt of the anonymous bulk digital signing request to Online Service server by returning POST response with parameter “ticketID”;
- Step 5&6. Online Service server uses the Document Hash / PDF digest, HKICHash and hash of clientID to calculate a 6-digit identification code and instructs Online Service App to display the instructions with the 6-digit identification code to inform “iAM Smart” user to process the digital signing authorisation request in “iAM Smart” Mobile App;
- Online Service server prepares necessary parameters such as “clientID”, “redirectURI” (set as Online Service App URL Scheme or Universal Link/App Link depending on “source” parameter), “scope”, “source” (set as “App_Scheme” or “App_Link”), “ticketID”, etc. and constructs the URL Scheme to invoke “iAM Smart” Mobile App by Online Service App;
- Step 7. Online Service App polls Online Service server for the digital signing result from “iAM Smart” BDSS;
- Step 8. Online Service App invokes “iAM Smart” Mobile App using URL Scheme with request parameters (set <Context> as “anon_bulk-sign” in URL Scheme);
- “iAM Smart” API (URL Scheme: Open “iAM Smart” Mobile App for Getting Context)*

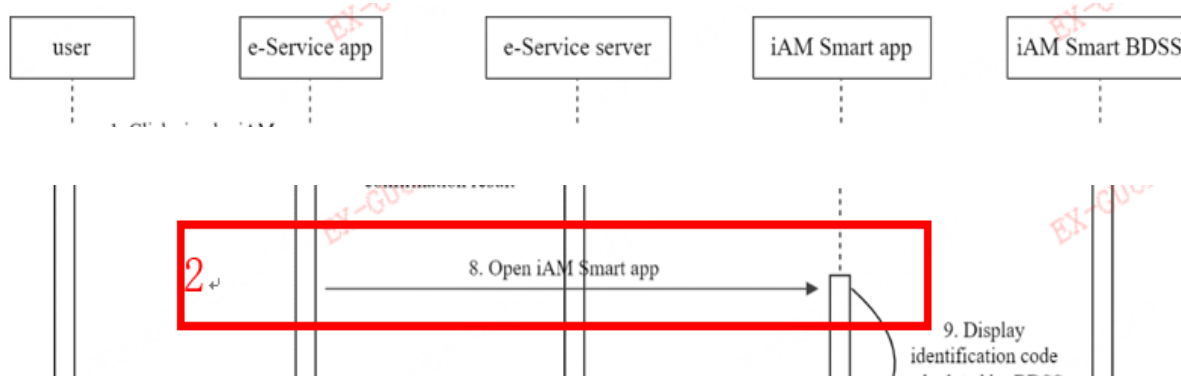
- Step 9. “iAM Smart” user logs in “iAM Smart” Mobile App, “iAM Smart” Mobile App requests and displays 6-digit identification code from “iAM Smart” BDSS. “iAM Smart” user reviews the digital signing authorisation request (e.g. continue or reject the request) and verifies the 6-digit identification code with Online Service Website;
- Step 10. “iAM Smart” user follows the instructions in “iAM Smart” Mobile App to complete digital signing operation;
- Step 11-12. “iAM Smart” BDSS verifies the validity of QR Code and other necessary information, and return the authorisation result to “iAM Smart” Mobile App;
- Step 13. “iAM Smart” Mobile App invokes Online Service App using URL Scheme or Universal Link/App Link with required parameters such as “authCode”, “businessID”;
- Online Service Callback API (GET: Callback with authCode to Online Service App)*
- Step 14. Online Service App sends authCode, businessID, etc. to Online Service server;
- Step 15-16. When Online Service server gets the authCode and businessID, it verifies businessID as generated in Step 3 and should then invoke the “iAM Smart” API to obtain the accessToken and the Tokenised ID (i.e. openID) of the “iAM Smart” user. API data encryption is required;
- “iAM Smart” API (POST: Request accessToken & Tokenised ID)*
- Step 17-8. Online Service server invokes the “iAM Smart” API with the accessToken and openID to obtain BSQCToken, then Online Service invokes the “iAM Smart” API with BSQCToken and openID to obtain the bulk digital signing result. API data encryption is required;
- “iAM Smart” API (POST: Request BSQC Token)*

3.12.4.1 Implementing (1) POST: Request Anonymous Bulk Digital Signing



Please refer to Section 3.12.1.1

3.12.4.2 Implementing (2) URL Scheme: Open “iAM Smart” Mobile App for Getting Context



Pre-conditions

- Please refer to Section 3.4.1.2.

Post-conditions

- Please refer to Section 3.4.1.2.

Error conditions

- Please refer to Section 3.4.1.2.

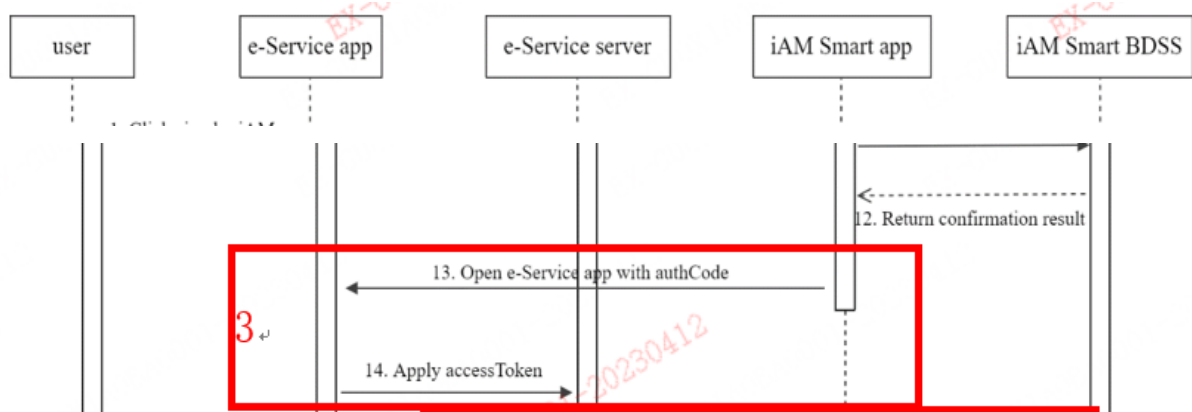
Request Parameters

- Please refer to Section 3.4.1.2.

Notes

- Please refer to Section 3.4.1.2, except:
 - Set <Context> as “anon_bulk-sign” in URL Scheme.

3.12.4.3 Implementing (3) GET: Callback with authCode to Online Service Server



Pre-conditions

- Please refer to Section 3.4.1.2.

Post-conditions

- Please refer to Section 3.4.1.2 except:
 - Online Service server should match the callback result with corresponding Online Service client terminal using the “businessID”.

Error conditions

- This Online Service callback API is a HTTP GET request. If parameter “error_code” is returned, it means the request is failed.

Error Code	Error Description	Suggested Action
error_code - D71001	User rejected signing request	Inform user the digital signing request is rejected
error_code - D71002	Failed to request signing	Inform user the digital signing request is failed and retry later
error_code - D71004	User not allowed to sign	Inform user that “iAM Smart” digital signing is not allowed for default “iAM Smart” and suggest user to upgrade to “iAM Smart” with digital signing function

Error Code	Error Description	Suggested Action
error_code - D71005	Inconsistent HKIC number	Hash of HKIC number provided is not matched with “iAM Smart” user. Check hash of HKIC number or inform user to authorise the digital signing using the correct “iAM Smart” account

The error_code D71003 (signing request timeout) does not appear in this scenario. For the QR Code timeout or request confirmation timeout in the “iAM Smart” Mobile App, message will be prompted in the corresponding user interface.

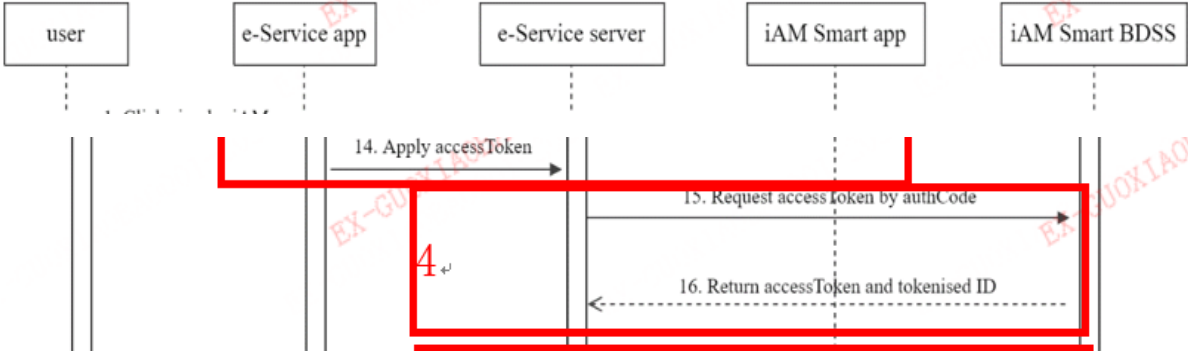
Callback Parameters

- Please refer to Section 3.4.1.2.

Notes

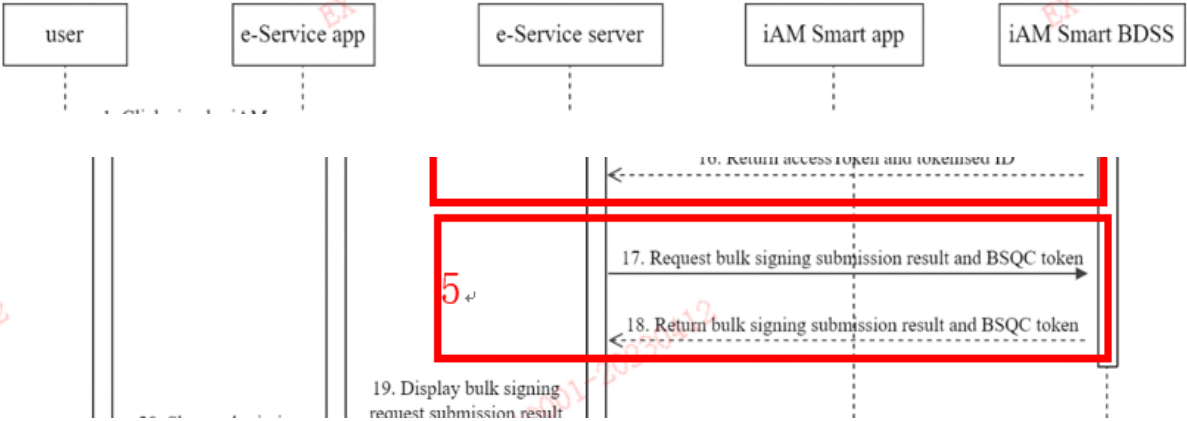
- Please refer to Section 3.4.1.2 except the following parameter:
 - Callback parameter “businessID”: Value returned by “iAM Smart” System should be the same one as Online Service submitted in the anonymous request.

3.12.4.4 Implementing (4) POST: Request accessToken and Tokenised ID



Please refer to Section 3.6.1.4

3.12.4.5 Implementing (5) POST: Request BSQC Token

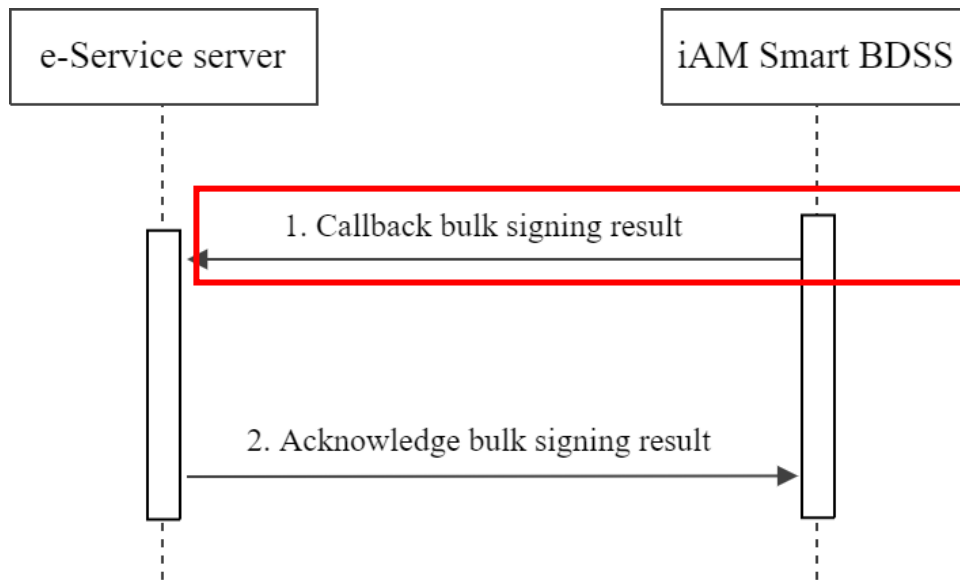


Please refer to Section 3.12.1.5

3.13 WORKFLOWS FOR CALLBACK BULK DIGITAL SIGNING RESULT

3.13.1 Workflows for Callback Bulk Digital Signing Result

3.13.1.1 Implementing (1) POST: Callback to Receive Bulk Digital Signing Result



Pre-conditions

- “iAM Smart” user can accept and complete the digital signing request.
- “iAM Smart” user can also reject the digital signing request.

Post-conditions

- API data decryption is required.
- Online Service server should match the callback result with corresponding Online Service Website/App using the “businessID”.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D71000	User cancelled signing request	Inform user the “iAM Smart” digital signing request is cancelled

code - D71001	User rejected signing request	Inform user the “iAM Smart” digital signing request is rejected
code - D71002	Failed to request signing	Inform user the “iAM Smart” digital signing request is failed and retry later
code - D71003	Signing request timeout	Inform user the “iAM Smart” digital signing request is timeout and provide way for user to retry

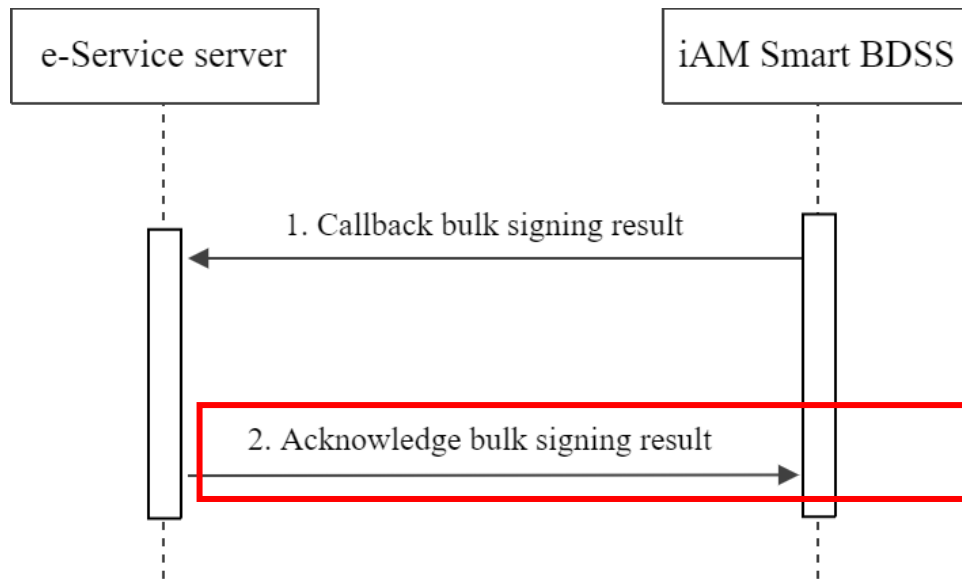
Callback Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Callback parameter “hashCode”: It is the Document Hash provided by Online Service and Online Service may use for checking purpose.
- Callback parameter “timestamp”: It is the timestamp returned by “iAM Smart” System when the digital signing operation is successful.
- Callback parameter “signature”: It is the RSA signature of the Document Hash submitted by Online Service for digital signing. The value is BASE64 encoded.
- Callback parameter “cert”: It is the “iAM Smart” e-Cert of the “iAM Smart” user. It is in X.509 format and the value is BASE64 encoded.

3.13.1.2 Implementing (2) POST: Online Service Acknowledges Bulk Digital Signing Result



Pre-conditions

- Online Service receives the “iAM Smart” bulk digital signing result and completes its document digital signing process.
- API data encryption is required.

Post-conditions

- Online Service may record the bulk digital signing acknowledgement result.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D71006	Failed to process digital signing acknowledgement	“iAM Smart” digital signing acknowledgement processing failed, retry later

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

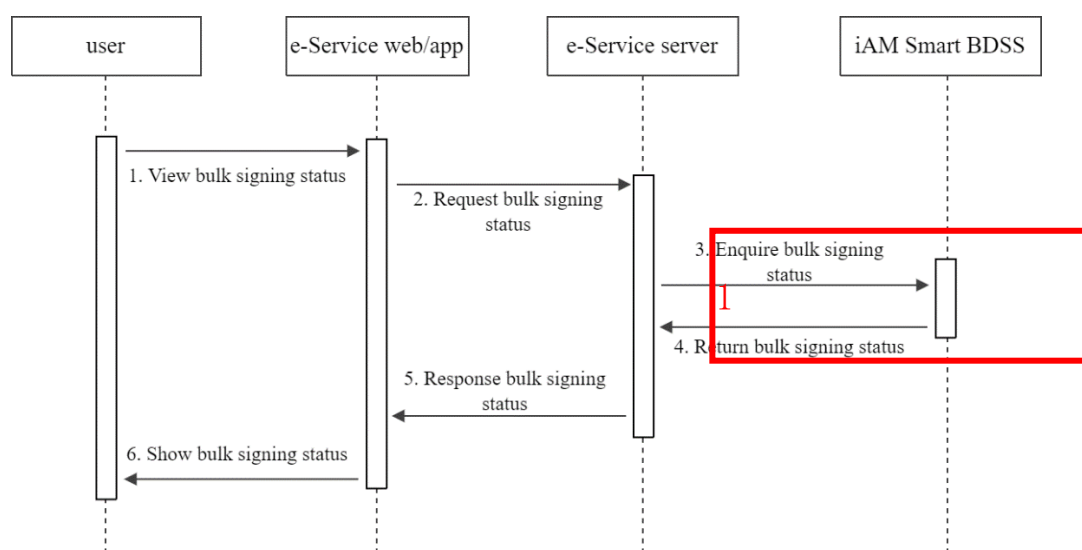
- For common parameters in request and response, please refer to Section 3.2.2.

- Request parameter “businessID”: It is the same “businessID” submitted in the digital signing request.
- Request parameter “signingResult”: It is the status of Online Service document digital signing process after receiving the “iAM Smart” digital signing result. Its value must be equal to those specified in this API.

3.14 WORKFLOWS FOR ENQUIRE BULK DIGITAL SIGNING RESULT

3.14.1 Workflows for Enquire Bulk Digital Signing Result

3.14.1.1 Implementing (1) POST: Enquire Bulk Digital Signing Status



Pre-conditions

- Online Service provides parameters “openID” and “BSQCToken”.
- API data encryption is required.

Post-conditions

- API data encryption is required.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D71008	BSQC Token not found	BSQC Token may expire

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

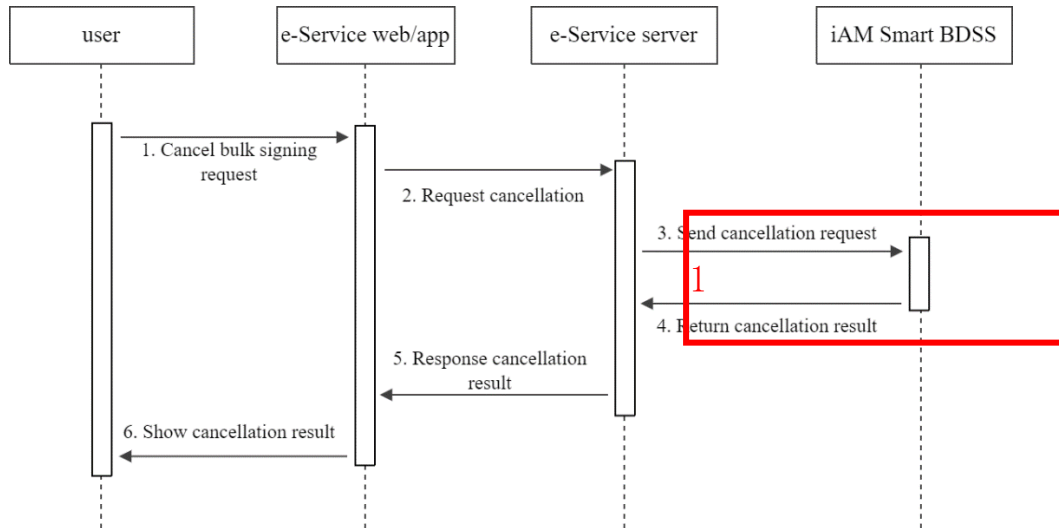
Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- The BSQC Token that's used to enquire digital signing status or cancel signing request is valid for 1 day.

3.15 WORKFLOWS FOR CANCEL BULK DIGITAL SIGNING REQUEST

3.15.1 Workflows for Cancel Bulk Digital Signing Request

3.15.1.1 Implementing (1) POST: Cancel Bulk Digital Signing Request



Pre-conditions

- Online Service provides parameters “openID” and “BSQCToken”.
- The request of which status is “Pending for Signing” can be cancelled.
- API data encryption is required.

Post-conditions

- API data encryption is required.

Error conditions

- For common errors, please refer to Section 3.2.3. Other errors of this API include:

Error Code	Error Description	Suggested Action
code - D71008	BSQC Token not found	BSQC Token may expire

Request and Response Parameters

- Please refer to “iAM Smart” API Specification.

Notes

- For common parameters in request and response, please refer to Section 3.2.2.
- Only the request of which status is “Pending for Signing” can be cancelled.

- The BSQC Token that's used to enquire digital signing status or cancel signing request is valid for 1 day.

APPENDICES

A. NOT APPLICABLE

B. NOT APPLICABLE

C. NOT APPLICABLE

D. ESSENTIAL TIPS FOR APP-TO-APP DIRECT LOGIN V2

Direct Login v2 has adopted a new approach on Android for launching the online service app. The `authCode` and related parameters are passed in the form of an Android Intent. This section outlines essential tips and common pitfalls associated with Direct Login v2, including issues where the online service app does not launch successfully and the `authCode` cannot be retrieved from the “iAM Smart” Android Intent.

D.1 Prerequisites to launch Android App via Package Name

In order to launch the online service app from the “iAM Smart” App catalogue successfully, the following configuration shall be valid:

- Package name
- Activity name
- Signing certificate fingerprint

An error message stating “An error occurred, unable to launch the mobile application” will be displayed if otherwise. Information can be gathered from the ESP, the online service app, and the submitted “iAM Smart” catalogue application form to ensure that the above configurations are **aligned and correct**.

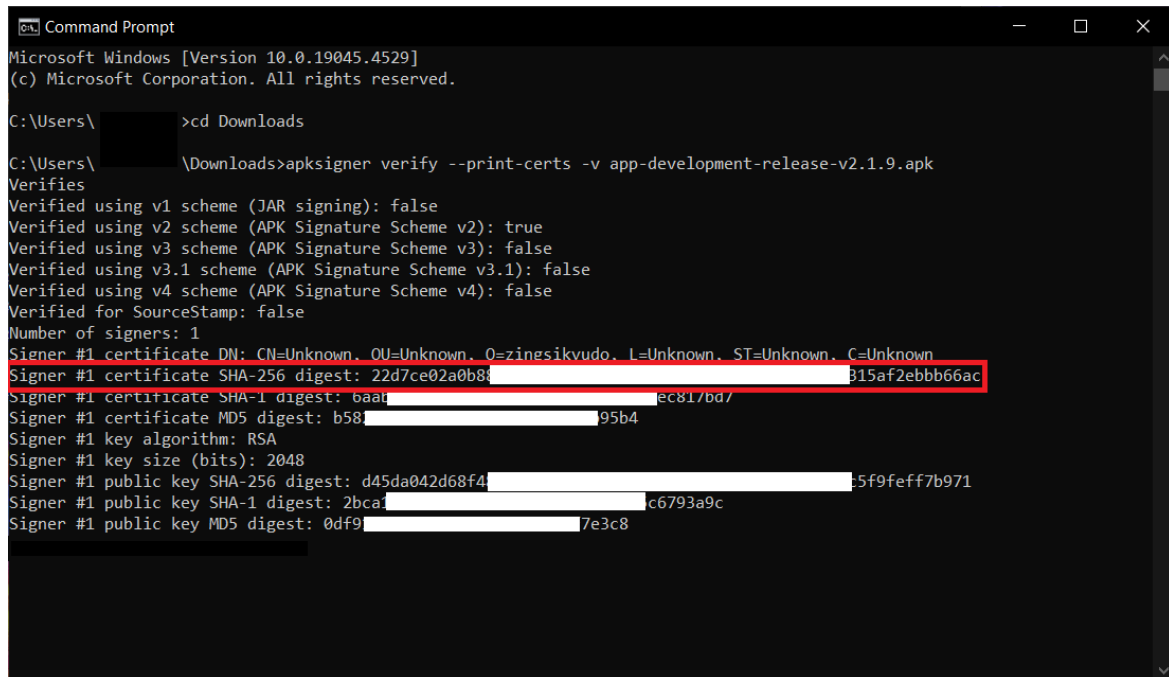
Item	Item to check	Retrieve information from online service app
Certificate Fingerprint	ESP	Refer to D.1.1 錯誤! 找不到參照來源。
	APK	
Package name	ESP	Refer to D.1.2
	APK	
	Application form	
Activity class name	APK	Refer to D.1.3
	Application form	

D.1.1 Retrieve signing certificate from online service app

To retrieve the certificate from the online service app, the Android SDK Platform-Tools are

required. After installing the Android SDK Platform-Tools, execute the command below to obtain the signing fingerprint of the app. Please note that if different certificates are used for signing the production and development apps, the fingerprints will differ..

```
apksigner verify --print-certs -v <path-to-apk-file>
```



```
Microsoft Windows [Version 10.0.19045.4529]
(c) Microsoft Corporation. All rights reserved.

C:\Users\>cd Downloads

C:\Users\>\Downloads>apksigner verify --print-certs -v app-development-release-v2.1.9.apk
Verifies
Verified using v1 scheme (JAR signing): false
Verified using v2 scheme (APK Signature Scheme v2): true
Verified using v3 scheme (APK Signature Scheme v3): false
Verified using v3.1 scheme (APK Signature Scheme v3.1): false
Verified using v4 scheme (APK Signature Scheme v4): false
Verified for SourceStamp: false
Number of signers: 1
Signer #1 certificate DN: CN=Unknown, OU=Unknown, O=zingsikvudo, L=Unknown, ST=Unknown, C=Unknown
Signer #1 certificate SHA-256 digest: 22d7ce02a0b8...b15af2ebbb66ac
Signer #1 certificate SHA-1 digest: baaf...ec817bd7
Signer #1 certificate MD5 digest: b58...95b4
Signer #1 key algorithm: RSA
Signer #1 key size (bits): 2048
Signer #1 public key SHA-256 digest: d45da042d68f4...5f9feff7b971
Signer #1 public key SHA-1 digest: 2bca...c6793a9c
Signer #1 public key MD5 digest: 0df9...7e3c8
```

The fingerprint can be found in the Signer certificate SHA-256 digest. Please note that the format may differ from what is required in the ESP. For example, the format in the ESP is 22:D7:CE...66:AC, whereas it may appear as 22d7ce...66ac in the terminal.

D.1.2 Retrieve package name from online service app

To retrieve the package name from the online service app, the Android SDK Platform-Tools are required. After installing the Android SDK Platform-Tools, execute the command below to obtain the package name of the app. Please note that the package names for the production and development apps may differ depending on the configuration of the online service app.

```
aapt dump xmltree <path-to-apk-file> AndroidManifest.xml
```

```
Command Prompt
C:\Users\...\Downloads>aapt dump xmltree esdemo-aos.apk AndroidManifest.xml
N: android=http://schemas.android.com/apk/res/android
E: manifest (line=2)
  A: android:versionCode(0x0101021b)=(type 0x10)0x1
  A: android:versionName(0x0101021c)="3.5.0" (Raw: "3.5.0")
  A: android:compileSdkVersion(0x01010572)=(type 0x10)0x1f
  A: android:compileSdkVersionCodename(0x01010573)="12" (Raw: "12")
  A: package="com. ....eservice" (Raw: "com. ....eservice")
  A: platformBuildVersionCode=(type 0x10)0x17
  A: platformBuildVersionName=(type 0x10)0xc
E: uses-sdk (line=7)
  A: android:minSdkVersion(0x0101020c)=(type 0x10)0x17
  A: android:targetSdkVersion(0x01010270)=(type 0x10)0x1f
E: uses-permission (line=11)
  A: android:name(0x01010003)="android.permission.READ_EXTERNAL_STORAGE" (Raw: "android.permission.READ_EXTERNAL_STORAGE")
E: uses-permission (line=12)
  A: android:name(0x01010003)="android.permission.WRITE_EXTERNAL_STORAGE" (Raw: "android.permission.WRITE_EXTERNAL_STORAGE")
E: uses-permission (line=13)
  A: android:name(0x01010003)="android.permission.MANAGE_EXTERNAL_STORAGE" (Raw: "android.permission.MANAGE_EXTERNAL_STORAGE")
E: uses-permission (line=14)
  A: android:name(0x01010003)="android.permission.VIBRATE" (Raw: "android.permission.VIBRATE")
E: uses-permission (line=15)
  A: android:name(0x01010003)="android.permission.INTERNET" (Raw: "android.permission.INTERNET")
E: queries (line=17)
E: intent (line=18)
  E: action (line=19)
    A: android:name(0x01010003)="android.intent.action.VIEW" (Raw: "android.intent.action.VIEW")
```

The package name can be found in “A: package=...”.

D.1.3 Retrieve activity name from online service app

Online service provider should have decided which activity to handle the incoming intent from “iAM Smart”. To retrieve the full activity name (e.g. `com.example.dev.MainActivity` instead of `MainActivity`) from the app, the Android SDK Platform-Tools are required. After installing the Android SDK Platform-Tools, execute the command below to obtain the activity name of the app.

```
aapt dump xmltree <path-to-apk-file> AndroidManifest.xml
```

The contents of the Android Manifest file will be displayed in the terminal, including the activity name that handles the intent from “iAM Smart”.

D.2 Use correct method to obtain the `authCode` and `code_verifier` from Android Intent

Since the iAM Smart has adopted a new approach on Android to pass the `authCode`, online service app is required to retrieve the `authCode` and `code_verifier` from the received intent using the following code.

```
val authCodeId = intent.getStringExtra("code")
val codeVerifier = intent.getStringExtra("code_verifier")
```

```
val activityParams = intent.getStringExtra("activityParams") //
optional
```

Please note that the code uses `getStringExtra()` method, not the `getData()` method.

For more details, refer to the section 3.10.4.1.