ADSS RAS Developers Guide

ASCERTIA LTD

FEBRUARY 2021

Document Version – 6.8.0.2

CONTENTS

1 Introduction

1.1 Scope

This document provides information on how to integrate mobile applications and business applications with ADSS Server RAS Service for remote signature authorisation.

The integration uses REST architectural style APIs only. These calls are sent over HTTPS from the mobile device to the ADSS Server RAS Service.

1.2 Intended Readership

This guide is intended for developers who are integrating mobile applications with ADSS Server for remote signature authorisation. The document assumes a reasonable knowledge of web application development, specifically RESTful Web services and ADSS Server.

1.3 Conventions

The following typographical conventions are used in this guide to help locate and identify information:

- **Bold** text identifies menu names, menu options, items you can click on the screen, file names, folder names, and keyboard keys.
- Courier New font identifies code and text that appears on the command line.
- **Bold Courier New** identifies commands that you are required to type in.
- Courier New font identifies Ajax request/response in HTTP message body.

1.4 Technical support

If technical support is required, Ascertia has a dedicated support team. Ascertia Support can be contacted in the following ways:

Support Website	www.ascertia.com/support
Support Email	support@ascertia.com
Knowledge base	http://kb.ascertia.com/display/AKBS/Ascertia+Knowledge+base

In addition to the free support service describe above, Ascertia provides formal support agreements with all product sales. Please contact sales@ascertia.com for more details.

A Product Support Questionnaire should be completed to provide Ascertia Support with further information about your system environment. When requesting help, it is always important to confirm:

- System Platform details.
- ADSS Server version number and build date.
- Details of specific issue and the relevant steps taken to reproduce it.
- Database version and patch level.
- Product log files

2 ADSS Server RAS Service Overview

ADSS Server RAS Service is the client-facing component of the ADSS Server remote signing solution. It acts as a gateway controlling access to the ADSS Server Signature Activation Module (SAM) which performs the actual remote signing operation. For brevity the ADSS Server RAS Service will be referred to as ADSS RAS throughout this document.

The purpose of ADSS RAS is to manage:

- RAS registration services:
  - Register users for remote signing. This involves not only registering the user details (e.g. Name, email and phone number) but also requesting their signing key pair generation inside the ADSS Server SAM's HSM and then ensuring the corresponding public key certificate is issued by communicating with various ADSS Server components (and optionally any external CAs).
  - Register user's mobile devices for remote signing. It is possible for a user to register multiple devices.
- RAS signing services:
  - Receiving signing requests from business applications on behalf of users. Note business applications can either communicate with the ADSS Signing Service component which acts as a Signature Creation Application (SCA) which then passes the Data To Be Signed/Represented (DTBS/R) to ADSS RAS or they can directly interact with RAS Service.
  - Request authorisation of the remote signature from the user, by conducting a Signature Activation Protocol (SAP) with the user's registered mobile device.

Note for both registration and signing ADSS RAS is not the end-point, it acts as a front-end management service for the ADSS Server SAM service.

ADSS RAS has an Ascertia-defined API for user registration, device registration and certificate management and follows the industry-defined Cloud Signature Consortium[1] protocol for signing operations. The Signature Activation Protocol (SAP) interface with the user's mobile device for authorising the remote signature is also Ascertia-defined.

[1] See http://www.cloudsignatureconsortium.org/ for more details

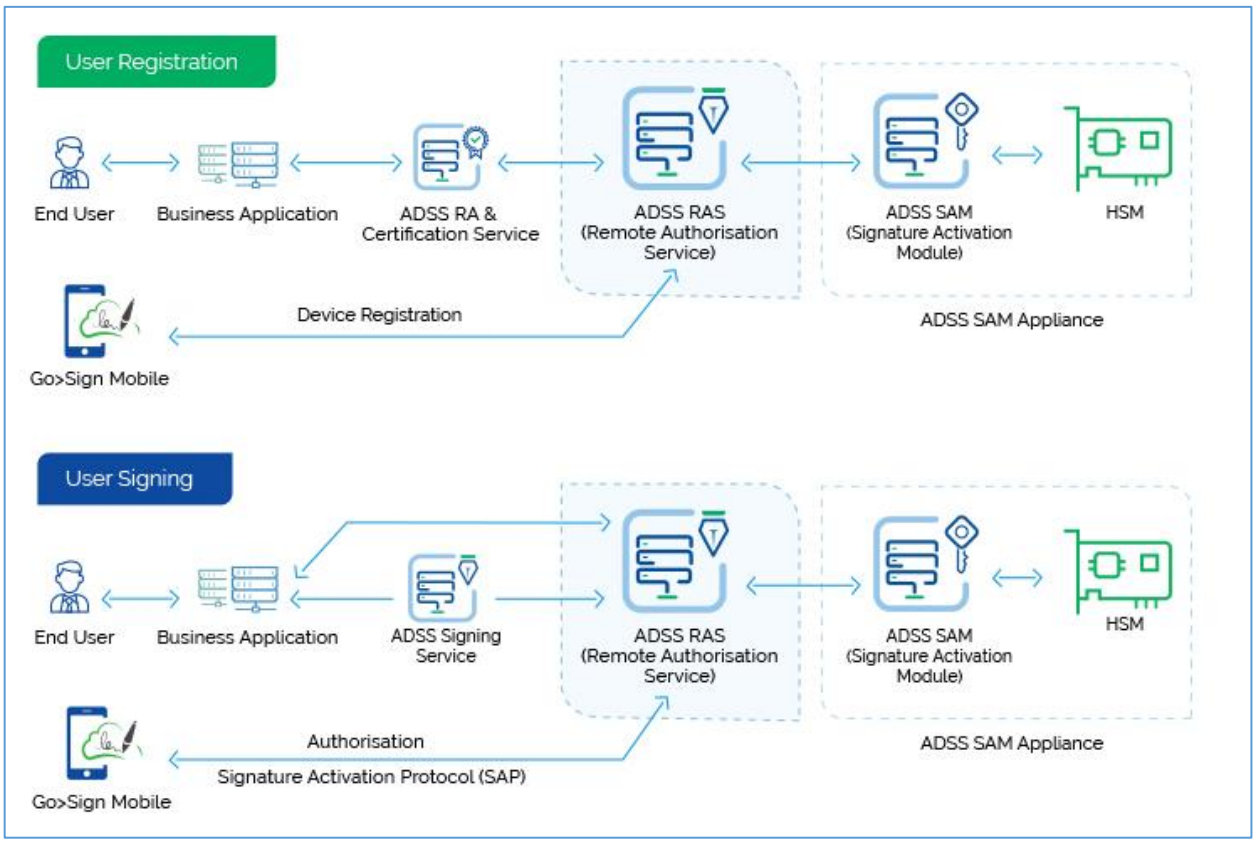

Figure 1 - RAS Service & Business Application Interaction

ADSS RAS receives all the benefits of the well-proven, robust architecture of ADSS Server. The ADSS Server Architecture & Deployment Guide describes how to implement a high availability and fault tolerant solution.

Calls to ADSS Services, including the RAS Service, use standard ADSS Server Tomcat HTTPS Listeners/Connectors. Port 8778 is used to communicate with ADSS Server over server-side TLS v1.2.

3 Business Application Interfaces

ADSS RAS has a number of APIs aimed at business applications which initiate user registrations and signing operations. We can categorise the APIs in two sections:

- Ascertia APIs
- CSC APIs

The details of both APIs is given below.

3.1 Ascertia APIs

The APIs implemented by Ascertia for ADSS RAS Service is given below:

3.1.1 Register User

Creates a user in SAM Service. When a new user is created then response status '201' is returned. A business application will register its users using this interface.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Request Body	{ "client_id": "samples_test_client", "user_id": "johnDoe12", "app_name": "Application_01", "user_name": "John Doe", "user_password": "password", "user_email": "john.doe@ascertia.com", "user_mobile": "00448007720442", "profile_id": "profile-001" }	
Status Code	**Message**	**Response Body**
201	Created	
200	OK	
400	Bad Request	
401	Unauthorised	
403	Forbidden	
500	Internal Server Error	

Table 1 – Register User

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS RAS Console > Client Manager
user_id	User ID identifying the registered user in RAS service (max. 50 characters and allowed characters are **a-zA-Z0-9_.@-**)
app_name	(Optional) Some business entity that requested to register the user. (max. 50 characters). Later would be used as search filter in Get Users API.
user_name	(Optional) User name as friendly name for the registered user in RAS service (max. 50 characters) Following languages are supported for username • Norwegian Characters • Slovenian Characters • Czech & Slovak Characters • Icelandic Characters • Arabic Characters • Latvian Characters
user_password	(Optional) Password for the registered user in RAS service (max. 50 characters)
user_email	Email for the registered user in RAS service. It will be used to send OTP for device registration etc. (max. 100 characters)
user_mobile	Mobile number for the registered user in RAS service. It will be used to send OTP for device registration etc. (max. 100 characters)
profile_id	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
error_code	The error code.
error_description	A string with the description of the error_code.

3.1.2 Update User

Updates user information by SAM Service. When a user is updated then response status ‘200’ is returned. A business application will update its user’s information using this interface.

Error! Hyperlink reference not valid./{client_id}/{user_id}	
HTTP Verb	PUT
Content-Type	application/json
Accept	application/json
Request Body	{ "user_name": "John Doe", "user_email": "john.doe@ascertia.com", "user_mobile": "00448007720442", "profile_id": "profile-001", "status": "INACTIVE"

	}	
Status Code	**Message**	**Response Body**
201	Created	
200	OK	
400	Bad Request	
401	Unauthorised	
403	Forbidden	
500	Internal Server Error	

Table 2 – Update User

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS RAS Console > Client Manager
user_id	User ID identifying the registered user in RAS service (max. 50 characters)
user_name	(Optional) User name as friendly name for the registered user in RAS service (max. 50 characters)
user_email	Email for the registered user in RAS service. It will be used to send OTP for device registration etc. (max. 100 characters)
user_mobile	Mobile number for the registered user in RAS service. It will be used to send OTP for device registration etc. (max. 100 characters)
status	Status of the user in RAS Service. The status of a user can be updated using the values (ACTIVE/INACTIVE/BLOCKED).
profile_id	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
error_code	The error code.
error_description	A string with the description of the error_code.

3.1.3 Delete User

Deletes a user in RAS Service identified by {user_id}. This interface will be used by a business application to remove a user.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/{client_id}/{user_id}?profile_id=xyz		
HTTP Verb	DELETE	
Accept	application/json	
Request Body		
Status Code	**Message**	**Response Body**
200	OK	
404	Not Found	
403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{client_id}	Client ID which is configured in ADSS Console > Client Manager
{user_id}	User ID identifying the registered user in RAS service
profile_id	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
error	The error code
error_description	Error description message

Table 3 - Delete User

3.1.4 Get User

Returns a user's information registered in RAS Service identified by {user_id}. A business application will use this interface to get a user's information.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/{client_id}/{user_id}?profile_id=xyz		
HTTP Verb	GET	
Content-Type		
Accept	application/json	
Request Body		
Status Code	**Message**	**Response Body**
200	OK	{ "user_id": "johnDoe12" "user_name": "John Doe", "app_name": "Application_01", "user_email": "john.doe@ascertia.com", "user_mobile": "00448007720442", "status": "ACTIVE",

		"created_at": "2020-12-15 12:19:39", "last_updated_at": "2020-12-15 12:22:19" "profile_id": "adss:sam:profile:001", }
404	Not Found	
403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{client_id}	Client ID which is configured in ADSS Console > Client Manager
{user_id}	User ID identifying the registered user in RAS service
profile_id	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
user_id	User ID identifying the registered user in RAS service (max. 50 characters)
user_name	(Optional) User name as friendly name for the registered user in RAS service (max. 50 characters)
user_email	Email for the registered user in RAS service. It will be used to send OTP for device registration etc. (max. 100 characters)
user_mobile	Mobile number for the registered user in RAS service. It will be used to send OTP for device registration etc. (max. 100 characters)
status	Status of the user in RAS Service. The status of a user can be updated using the values (ACTIVE/INACTIVE/BLOCKED).
created_at	It returns the date on which user is created.
last_updated_at	It returns the date on which user is modified.
error	The error code
error_description	Error description message

Table 4 - Get User

3.1.5 Get Users

Returns users information registered in RAS Service identified by {user_id}. A business application will use this interface to get users information.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/{client_id}/{start_pointer}/{fetch_size}?{query_params}	
HTTP Verb	GET
Content-Type	

Accept	application/json	
Request Body		
Response Headers		
x-total-records	2	
Status Code	**Message**	**Response Body**

200	OK	[{
		"user_name": "John Doe",
		"user_id": "johnDoe12",
		"app_name": "Application_01",
		"user_email": "john.doe@ascertia.com",
		"user_mobile": "00448007720442",
		"status": "ACTIVE",
		"created_at": "2017-10-10 10:30:00",
		"last_updated_at": "2017-10-10 10:30:00"
		"profile_id": "adss:sam:profile:001"
		},{
		"user_name": "Peter Doe",
		"user_id": "peterDoe12",
		"app_name": "Application_01",
		"user_email": "peter.doe@ascertia.com",
		"user_mobile": "00448007720442",
		"status": "ACTIVE",
		"created_at": "2017-10-10 10:30:00",
		"last_updated_at": "2017-10-10 10:30:00"
		"profile_id": "adss:sam:profile:001"
		}]

404	Not Found	
403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{client_id}	Client ID which is configured in ADSS Console > Client Manager
{user_id}	User ID identifying the registered user in RAS service
{query_params}	
profile_id	RAS Profile ID that will be used to communicate with RAS service.
client_id	Client ID that will be used in listing of users.
app_name	Application name to be used by business application for listing of users.

Response Parameters	
user_id	User ID identifying the registered user in RAS service (max. 50 characters)
user_name	(Optional) User name as friendly name for the registered user in RAS service (max. 50 characters)
user_email	Email for the registered user in RAS service. It will be used to send OTP for device registration etc. (max. 100 characters)
user_mobile	Mobile number for the registered user in RAS service. It will be used to send OTP for device registration etc. (max. 100 characters)
status	Status of the user in RAS Service. The status of a user can be updated using the values (ACTIVE/INACTIVE/BLOCKED).
created_at	It returns the date on which user is created.
last_updated_at	It returns the date on which user is modified.
error	The error code
error_description	Error description message

Table 5 - Get Users

3.1.6 Change Password

This interface is used to change the password of a user. The user provides the old password and new password in request. The RAS verifies the old password and after successful verification, it will change the old password with the new one.

Note: This interface will only be used if a password was provided at the time of user registration, otherwise it is of no use and the server will return error 'Unauthorized' as there will be no password stored against the user.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/change/password		
HTTP Verb	PUT	
Content-Type	application/json	
Accept	application/json	
Request Body	{ "client_id": "samples_test_client", "user_id": "johnDoe12", "profile_id": "profile-001", "user_password": "old-password", "user_new_password": "new-password" }	
Status Code	**Message**	**Response Body**
200	OK	
400	Bad Request	
401	Unauthorized	
404	Not Found	If URL does not contain the {Client_id} or {user_id}

403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in RAS service
profile_id	RAS Profile ID that will be used to communicate with RAS service.
user_password	Old password of the user that he wants to change
user_new_password	New password of the user
Response Parameters	
error	The error code
error_description	Error description message

Table 6 - Change Password

3.1.7 Recover Password

Initiates user password recovery process. If a user forgets his/her password, this interface can be used to recover/reset a password. Password recovery is done in two steps; first the business application will call this interface to initiate the process, then RAS will send either one or two OTPs to user's mobile and email according to the RAS Profile settings. The client will send these OTPs in a separate call using another interface discussed in next section.

Note: if the user was registered without password, this interface can also be used to set a password for that user.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/password/recover		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	

Request Body	{
	"client_id": "samples_test_client",
	"user_id": "johnDoe12",
	"profile_id": "profile-001"
	}	

Status Code	**Message**	**Response Body**

200	OK	If two OTPs will be sent to user:
		[
		{
		"type": "EMAIL_OTP",
		"sent_to": "john.doe@sample.som",
		},
		{
		"type": "SMS_OTP",
		"sent_to": "+448007720442"
		}
		]

		If one OTP will be sent on user email:
		{
		"type": "EMAIL_OTP",
		"sent_to": "john.doe@sample.som",
		}

		If one OTP will be sent to user's mobile:
		{
		"type": "SMS_OTP",
		"sent_to": "+448007720442"
		}

400	Bad Request	
403	Forbidden	
404	Not Found	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in SAM service.
profile_id	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
type	As OTP can be sent on both mediums i.e email/mobile so it defines the type.
sent_to	It could be the mobile number or email of the user depends upon the profile configuration.
error	The error code
error_description	Error description message

Table 7 – Recover Password

3.1.8 Confirm Recover Password

Completes user password recovery process. The business application will send the OTPs and new password in request, and the RAS will first validate the OTP and after successful validation, change the old password with the new password.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/password/recoverconfirm		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Request Body	{ "client_id": "samples_test_client", "user_id": "johnDoe12", "profile_id": "profile-001", "sms_otp": "225665", "email_otp": "654456", "user_password": "P@$$w0rD!@" } If only one OTP will be received by user either on SMS or email, the request would contain only “sms_otp” or “email_otp”.	
Status Code	**Message**	**Response Body**
200	OK	
400	Bad Request	
401	Unauthorized	
403	Forbidden	
404	Not Found	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in SAM service.
profile_id	RAS Profile ID that will be used to communicate with RAS service.
sms_otp	Received on the user’s registered mobile number
email_otp	Received on the user’s registered email

user_password	New password for the registered user in SAM service (max. 50 characters)
Response Parameters	
type	As OTP can be sent on both mediums i.e email/mobile so it defines the type.
sent_to	It could be the mobile number or email of the user depends upon the profile configuration.
error	The error code
error_description	Error description message

Table 8 – Confirm Recover Password

3.1.9 Change User Email

This interface will be used by a business application to change a user's email. The change email process completes in two steps. In first step, the business application will send the user ID and new email address on this interface, and the RAS will send the OTP(s) to the user (one on mobile and one on the email address according to RAS Profile). The business application will then send these OTPs in another request using another interface, that is discussed in next section.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/email/change		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Request Body	{ "client_id": "samples_test_client", "user_id": "johnDoe12", "user_email": "john.doe@ascertia.com", "profile_id": "profile-001" }	
Status Code	**Message**	**Response Body**
200	OK	If two OTPs will be sent to user: [{ "type": "EMAIL_OTP", "sent_to": "john.doe@sample.som", }, { "type": "SMS_OTP", "sent_to": "+448007720442" }]
		If one OTP will be sent on user email: { "type": "EMAIL_OTP", "sent_to": "john.doe@sample.som",

		}
		
		If one OTP will be sent to user's mobile:
		{
		"type": "SMS_OTP",
		"sent_to": "+448007720442"
		}

400	Bad Request	
403	Forbidden	
404	Not Found	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in SAM service.
profile_id	RAS Profile ID that will be used to communicate with RAS service.
user_email	New Email for the registered user in SAM service. It will be used to send OTP for device registration etc. (max. 100 characters)
Response Parameters	
type	Type of the OTP e.g. SMS/Email
sent_to	Mobile number or email of the user where OTP is sent
error	The error code
error_description	Error description message

Table 9 - Change User Email

3.1.10 Confirm Change User Email

Once the OTPs are received by the user for change email, the business application will provide these OTPs to RAS by calling this interface. The RAS will first validate the both OTPs and then change the old email with the new email address.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/email/changeconfirm	
HTTP Verb	POST
Content-Type	application/json
Accept	application/json

Request Body	{
	"client_id": "samples_test_client",
	"user_id": "johnDoe12",
	"sms_otp": "225665",
	"email_otp": "654456",
	"profile_id": "profile-001"
	}
	
	If only one OTP will be received by user either on SMS or email, the request would contain only “sms_otp” or “email_otp”.

Status Code	Message	Response Body
200	OK	
400	Bad Request	
401	Unauthorized	
403	Forbidden	
404	Not Found	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in SAM service.
profile_id	RAS Profile ID that will be used to communicate with RAS service.
sms_otp	Received by the user’s registered mobile number
email_otp	Received by the user’s registered email
user_password	New password for the registered user in SAM service (max. 50 characters)
Response Parameters	
error	The error code
error_description	Error description message

Table 10 – Confirm Change User Email

3.1.11 Change User Mobile

A business application will call this interface of RAS in order to change a user’s mobile number. Like ‘Change Email’, this process also completes in two steps. The business application will send the user ID and new mobile number on this interface and RAS will send the OTPs (one on user’s email and another on the provided new mobile number).

After receiving the OTPs, the business application will call another interface discussed in next section to complete the process.

Exposed for: Business Applications

<table>
<tr><td colspan="3">https://server:8778/adss/service/ras/v1/users/mobile/change</td></tr>
<tr><td>HTTP Verb</td><td colspan="2">POST</td></tr>
<tr><td>Content-Type</td><td colspan="2">application/json</td></tr>
<tr><td>Accept</td><td colspan="2">application/json</td></tr>
<tr><td>Request Body</td><td colspan="2">{
"client_id": "samples_test_client",
"user_id": "johnDoe12",
"user_mobile": "+448007720442",
"profile_id": "profile-001"

}</td></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><td>Status Code</td><td>Message</td><td>Response Body</td></tr>
<tr><td rowspan="3">200</td><td rowspan="3">OK</td><td>If two OTPs will be sent to user:
[
{
"type": "EMAIL_OTP",
"sent_to": "john.doe@sample.som",
},
{
"type": "SMS_OTP",
"sent_to": "+448007720442"
}
]</td></tr>
<tr><td>If one OTP will be sent on user email:
{
"type": "EMAIL_OTP",
"sent_to": "john.doe@sample.som",
}</td></tr>
<tr><td>If one OTP will be sent to user’s mobile:
{
"type": "SMS_OTP",
"sent_to": "+448007720442"
}</td></tr>
<tr><td>400</td><td>Bad Request</td><td></td></tr>
<tr><td>403</td><td>Forbidden</td><td></td></tr>
<tr><td>404</td><td>Not Found</td><td></td></tr>
</table>

500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in SAM service.
profile_id	RAS Profile ID that will be used to communicate with RAS service.
user_mobile	New mobile number for the registered user in SAM service. It will be used to send OTP for device registration etc. (max. 100 characters)
Response Parameters	
type	As OTP can be sent on both mediums i.e email/mobile so it defines the type.
sent_to	It could be the mobile number or email of the user depends upon the profile configuration.
error	The error code
error_description	Error description message

Table 11 - Change User Mobile

3.1.12 Confirm Change User Mobile

Once the OTPs to change user mobile are received by the user. The business application will call this interface providing the both OTPs to SAM. The SAM will first validate the OTPs and after successful verification, it will change the old mobile number with the new one.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/mobile/changeconfirm	
HTTP Verb	POST
Content-Type	application/json
Accept	application/json

Request Body	{
	"client_id": "samples_test_client",
	"user_id": "johnDoe12",
	"sms_otp": "225665",
	"email_otp": "654456",
	"profile_id": "profile-001"
	
	}
	
	If only one OTP will be received by user either on SMS or email, the request would contain only “sms_otp” or “email_otp”.

Status Code	**Message**	**Response Body**
200	OK	
400	Bad Request	
401	Unauthorized	
403	Forbidden	
404	Not Found	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in SAM service.
profile_id	RAS Profile ID that will be used to communicate with RAS service.
sms_otp	Received by the user's registered mobile number
email_otp	Received by the user's registered email
Response Parameters	
error	The error code
error_description	Error description message

Table 12 – Confirm Change User Mobile

3.1.13 Get Registered Devices

Get a list of all the registered devices against a user. The business application will call this interface to get all the registered devices of the logged in user.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/devices	
HTTP Verb	POST
Content-Type	
Accept	application/json
Request Body	{ "client_id": "samples_test_client", "profile_id": "profile-001",

	"user_id": "johnDoe12" }	
Status Code	**Message**	**Response Body**

200	OK	[
		{
		"device_id": "2eb1846d-81d8-40d0-86ba-d20bdf7ac5e0",
		"device_name": "IPhone",
		"secure_element": true,
		"biometric": true,
		},
		{
		"device_id": "3fc29573-92e9-40d0-86ba-d20bdf7ac5e0",
		"device_name": "Samsung",
		"secure_element": true,
		"biometric": true,
		
		}
		]

404	Not Found	
403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in SAM service
profile_id	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
device_id	Device ID of which is created at the time of device registration
device_name	Device name which is set at the time of device registration
device_certificate	Device Certificate
secure_element	“True” if device has secure element/enclave.
biometric	“True” if device has biometric feature available on the device. It can be TouchID, FaceID, Fingerprint etc. It can be used when the Device Registration is done from the mobile device.

status	It shows the status of device.
registered_at	It returns the registration time
user_id	User ID identifying the registered user in SAM service
error	The error code
error_description	Error description message

Table 13 – Get Registered Devices

3.1.14 Delete Device

Deletes a user's device in RAS Service identified by {user_id} and {device_id}. A business application would use this interface to delete a user's device.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/users/devices/{client_id}/{user_id}/{device_id}?{profile_id}=xyz		
HTTP Verb	DELETE	
Accept	application/json	
Request Body		
Status Code	**Message**	**Response Body**
200	OK	
404	Not Found	
403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{client_id}	Client ID which is configured in ADSS Console > Client Manager
{user_id}	User ID identifying the registered user in RAS service
{profile_id}	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
error	The error code
error_description	Error description message

Table 14 - Delete Device

3.1.15 Generate Key Pair

Creates a key pair for the user in RAS Service. When a new key pair is created the response status '201' is returned. The business applications will use this interface to generate the qualified key-pair for a user.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/keypairs		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	

Request Body	{
	"client_id": "samples_test_client",
	"user_id": "johnDoe12",
	"app_name": "Application_01",
	"user_password": "*****",
	"key_alias": "sample_key_alias",
	"profile_id": "adss:ras:profile:001"
	}	

Status Code	**Message**	**Response Body**
201	Created	
200	OK	
400	Bad Request	
404	Not Found	
403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in RAS service.
app_name	(Optional) Some business entity that requested to generate the key-pair. (max. 50 characters). Later would be used as search filter in Get User's Certificates API.
user_password	(Optional) password will only be required if key wrapping with Dynamic KEK is enabled in Hardware Crypto Profile.
key_alias	Key Alias of the key pair
profile_id	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
error	The error code.
error_description	Error description message

Table 15 – Generate Key Pair

3.1.16 Delete Key Pair

Deletes a user's keypair in RAS Service identified by {user_id} and {key_alias}. The business applications will call this interface to delete a key-pair of a user.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/keypairs/{client_id}/{user_id}/{key_alias}?profile_id={profile_id}		
HTTP Verb	DELETE	
Accept	application/json	
Request Body		
Status Code	**Message**	**Response Body**
200	OK	
404	Not Found	
403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{client_id}	Client ID which is configured in ADSS Console > Client Manager
{user_id}	User ID identifying the registered user in RAS service
{key_alias}	Key Alias of key pair that is going to be deleted.
{profile_id}	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
error	The error code.
error_description	Error description message

Table 16 – Delete Key Pair

3.1.17 Get CSR

Returns the base64 encoded CSR (Certificate Signing Request i.e. PKCS#10) of the key pair generated for the provided user.

The business applications will call this interface to get a CSR after generating a key-pair for a user.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/keypairs/csr	
HTTP Verb	POST

Content-Type	application/json	
Accept	application/json	

Request Body	{
	"client_id": "samples_test_client",
	"user_id": "johnDoe12",
	"user_password": "password12",
	"key_alias": "sample_key_alias",
	"profile_id": "profile-001"
	
	}	

Status Code	**Message**	**Response Body**

200	OK	{
		"csr": " MIICUzCCATsCAQAwDjEMMAoGA1……KJh"
		}

400	Bad Request	
404	Not Found	
403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in RAS service.
user_password	(Optional) used for unwrapping user key wrapped with dynamic KEK
key_alias	Key Alias of key pair for which CSR to be generated.
profile_id	RAS Profile ID that will be used to communicate with RAS service.
Response Parameters	
csr	Base 64 encoded CSR
error	The error code.
error_description	Error description message

Table 17 - Get CSR

3.1.18 Import Certificate

Uploads or import the user's certificate and certificate chain related to a key of the user.

The business applications will call this interface to import the certificate and its chain related to the key-pair of the user.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/keypairs/cert		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	

Request Body	{
	"client_id": "samples_test_client",
	"user_id": "johnDoe12",
	"key_alias": "sample_key_alias",
	"profile_id": "profile-001",
	"certificate": "HyguhugyCATsCAQAwDjEMMAoGA1……jhgjh=",
	"certificate_chain": [
	"HyguhugyCATsCAQAwDjEMMAoGA1……jhgjh=",
	"HyguhugyCATsCAQAwDjEMMAoGA1……jhgjh=",
	…
	],
	"p7b": "HyguhugyCATsCAQAwDjEMMAoGA1……jhgjh="
	}	

Status Code	**Message**	**Response Body**
200	OK	
400	Bad Request	
403	Forbidden	
404	Not Found	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID which is configured in ADSS Console > Client Manager
user_id	User ID identifying the registered user in RAS service.
key_alias	Key Alias of key pair for which certificate is to be imported.
profile_id	RAS Profile ID that will be used to communicate with RAS service.
certificate	Base 64 encoded string representing end entity certificate

certificate_chain	Array containing certificates chain in Base 64 encoded string
p7b	Certificate chain can also be provided in p7b format as Base 64 encoded string. certificate_chain and p7b can be provided alternatively. If both are present p7b will override certificate_chain.
Response Parameters	
error	The error code.
error_description	Error description message

Table 18 - Import Certificate

3.1.19 Get User's Certificates

Returns a list of all the certificates (with chains) for the provided registered user.

Exposed for: Business Applications

https://server:8778/adss/service/ras/v1/keypairs/cert/{client_id}/{user_id}?profile_id=adss:ras:profile:01		
HTTP Verb	GET	
Content-Type		
Accept	application/json	
Request Body		
Status Code	**Message**	**Response Body**

200	OK	[{
		"user_id": "Alice",
		"key_alias": "sample_cert_alias_01",
		"app_name": "Application_01"
		"key_status": "ACTIVE",
		"certificate_chain": [
		{"HyguhugyCATsCAQAwDjEMMAoGA1……jhgjh="},
		{"HyguhugyCATsCAQAwDjEMMAoGA1……jhgjh="}
		…
		]
		},
		{
		"user_id": "johnDoe12",
		"key_alias": "sample_cert_alias_02",
		"app_name": "Application_01"
		"key_status": " ACTIVE ",
		"certificate_chain": [
		{"HyguhugyCATsCAQAwDjEMMAoGA1……jhgjh="},
		{"HyguhugyCATsCAQAwDjEMMAoGA1……jhgjh="}
		…
		]
		}]

404	Not Found	

403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{client_id}	Client ID which is configured in ADSS Console > Client Manager
{user_id}	User ID identifying the registered user in RAS service.
{query_params}	Currently only supported parameter are “client_id “, “profile_id” and “app_name”. Response will contain all the certificates that contain this app_name provided as query parameter. e.g. …/service/ras/v1/keypairs/cert/list/my_client_01/user_01?client_id=abc&profile_id=xyz&app_name=application01
Response Parameters	
user_id	User ID identifying the registered user in SAM service.
key_alias	Key Alias of key pair for which certificate is to be imported.
profile_id	RAS Profile ID that will be used to communicate with RAS service.
certificate	Base 64 encoded string representing end entity certificate
certificate_chain	Array containing certificates chain in Base 64 encoded string
p7b	Certificate chain can also be provided in p7b format as Base 64 encoded string. certificate_chain and p7b can be provided alternatively. If both are present p7b will override certificate_chain.
error_description	Error description message

Table 19 - Get User Certificates

3.1.20 Application Meta Information

This call returns the meta information and the list of endpoints implemented by the service.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Authorization	No Auth	
Request Body		
Status Code	**Message**	**Response Body**

200	OK	{
		"specs": "1.0.3.0",
		"name": "Ascertia RAS",
		"logo": "https://localhost:8777/images/logo.png",
		"region": "GB",
		"lang": "en-gb",
		"description": "RAS - CSC Service provides remote authorization service implementing protection profiles",
		"oauth2BaseURI": "http://localhost:8777/adss/service/ras/csc/v1",
		"authType": [
		"basic",
		"oauth2code",
		" oauth2client "
		],
		"methods": [
		"auth/login",
		"auth/revoke",
		"credentials/list",
		"credentials/info",
		"credentials/authorize",
		"signatures/signHash",
		"oauth2/authorize",
		"oauth2/token",
		"oauth2/revoke"
		],
		}

404	Not Found	HTTP Status 404 – Not Found

Table 20 – Application Meta Information

3.1.21 Authentication/Login without Password

User can be registered in SAM without password so for that API will be used to generate the access token.

Error! Hyperlink reference not valid.	
HTTP Verb	POST
Content-Type	application/json
Accept	application/json

Request Body	{
	"client_id": “adss…client”,
	"client_secret": “fj49kl…….oOpQS”,
	"profile_id": “ADSS RAS Profile 001”,
	"user_id": jhon.wick
	}

Status Code	Message	Response Body

200	OK	{
		"access_token": "eyJhbGciOiJIUzI1NsInN…Pcxcz2hM",
		"expires_in": 3600
		}

400	Bad Request	{
		"error": "58071",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

401	Unauthorised	{
		"error": "59033",
		"error_description": "Failed to process request - user ID or password is invalid"
		}

Table 21 – Authentication/Login

3.2 CSC APIs

Ascertia has implemented CSC protocol to perform remote authorised signing. The Cloud Signature Consortium (CSC) is a group of industry and academic organizations committed to building new standards for cloud-based digital signatures that will support web and mobile applications and comply with the most demanding electronic signature regulations in the world. Below is the list of CSC APIs:

3.2.1 Authentication/Login

It is a username and password based authentication call which after successful authentication returns an access token and optionally refresh token based on input parameter in request.

Error! Hyperlink reference not valid.	
HTTP Verb	POST
Content-Type	application/json
Accept	application/json
Authorization	Basic c2FuOnBhc3N3b3Jk This is the base64 encoded value of Username:UserPassword.

Request Body	{
	"client_id": “adss…client”,
	"client_secret": “fj49kl…….oOpQS”,
	"profile_id": “ADSS RAS Profile 001”,
	"rememberMe": true
	}

Status Code	Message	Response Body

200	OK	{
		"access_token": "eyJhbGciOiJIUzI1NsInN…Pcxcz2hM",
		"refresh_token": "eyJJpc3MinRpYSN1YiI6InN…PCgvAI",
		"expires_in": 3600
		}

400	Bad Request	{
		"error": "58039",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

401	Unauthorised	{
		"error": "59033",
		"error_description": "Failed to process request - user ID or password is invalid"
		}

Table 1 – Authentication/Login

3.2.2 Authentication/Revoke

Revoke a service access token or refresh token that was obtained from the Remote Service, as described in RFC 7009. This method exists to enforce the security of the Remote Service. When the Signature Application needs to terminate a session, it is recommended to invoke this method to prevent further access by reusing the token.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Authorization	Bearer _TiHRG-bA H3XlFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw	

Request Body	{
	"token": "_TiHRG-bA H3XlFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw",
	"token_type_hint": "refresh_token"
	}	

Status Code	**Message**	**Response Body**
200	OK	
400	Bad Request	{ "error": "invalid_request",

		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed." }

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid string parameter token_type_hint"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Missing (or invalid type) string parameter token"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid string parameter token"
		}

Table 3 – Authentication/Revoke

3.2.3 Credentials/List

Returns the list of credentials associated with a user identifier. A user may have one or multiple credentials associated within a single Remote Signature Service Provider.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Authorization	Bearer _TiHRG-bA H3XlFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw	
Request Body		
Status Code	**Message**	**Response Body**

200	OK	{
		"credentialIDs": [
		"johnDoe"
		]
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

Table 4 – Credentials/List

3.2.4 Credentials/Info

Retrieve the credential and return the main identity information and the public key certificate or the certificate chain associated to it.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Authorization	Bearer _TiHRG-bA H3XlFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw	
Request Body	{ "credentialID": "JohnDoe", "certificates": "chain", "certInfo": true, "authInfo": true }	
Status Code	**Message**	**Response Body**
200	OK	{ "description": "Go>Sign mobile based implicit credential authorization", "key": { "status": "ENABLED", "algo": ["1.2.840.113549.1.1.1"], "len": 2048 }, "cert": { "status": "valid", "certificates": ["Base64-encoded X.509 end entity certificate", "Base64-encoded X.509 intermediate CA certificate", "Base64-encoded X.509 issuer CA certificate"], "issuerDN": "Issuer DN printable string", "SerialNumber": "5AAC41CD8FA22B953640", "subjectDN": "Subject DN printable string", "validFrom": "20180709132216+0000", "validTo": "20190709132216+0000" }, "authMode": "implicit", "SCAL": "2", "multisign": true, "lang": "en-gb }

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Missing (or invalid type) string parameter credentialID"
		}

400	Bad Reqeust	{
		"error": "58100",
		"error_description": "Invalid parameter credentialID"
		}

Table 5 – Credentials/Info

3.2.5 Credentials/Authorize

Authorize the access to the credential for remote signing, according to the authorization mechanisms associated to it. This method returns the Signature Activation Data (SAD) required to authorize the signatures/signHash method. PIN and/or OTP values collected from the user shall be present in the request according to the requirements specified by the credentials/info method.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Authorization	Bearer _TiHRG-bA H3XlFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw	

Request Body	{
	"credentialID": "JohnDoe",
	"numSignatures": 2,
	"documents": [{
	"document_id": 123,
	"document_name": “Document Name 123”,
	},{
	"document_id": 456,
	"document_name": “Document Name 456”,
	}],
	"hash":
	[
	"sTOgwOm+474gFj0q0x1iSNspKqbcse4IeiqlDg/HWuI=",
	"c1RPZ3dPbSs0NzRnRmowcTB4MWlTTnNwS3FiY3NlNEllaXFsRGcvSFd1ST0="
	]
	}	

Status Code	**Message**	**Response Body**

200	OK	{
		"SAD": "_TiHRG-bA4/CKN69L8gdSYp5_pw"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Missing (or invalid type) string parameter credentialID"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid parameter credentialID"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Missing (or invalid type) integer parameter numSignatures"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid parameter numSignatures"
		}

Table 6 – Credentials/Authorize

3.2.6 Signatures/signHash

Calculate the remote digital signature of one or multiple hash values provided as an input. This method requires providing credential authorization in the form of Signature Activation Data (SAD).

Error! Hyperlink reference not valid.	
HTTP Verb	POST
Content-Type	application/json
Accept	application/json
Authorization	Bearer _TiHRG-bA H3XlFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw

Request Body	{
	"credentialID": "JohnDoe",
	"SAD": "_TiHRG-bAH3XlFQZ3ndFhkXf9P24/CKN69L8gdSYp5_pw",
	"documents": [{
	"document_id": 123,
	"document_name": “Document Name 123”,
	},{
	"document_id": 456,
	"document_name": “Document Name 456”,
	}],
	"hash":

	["sTOgwOm+474gFj0q0x1iSNspKqbcse4IeiqlDg/HWuI="], "hashAlgo": "2.16.840.1.101.3.4.2.1", "signAlgo": "1.2.840.113549.1.1.1" }

Status Code	Message	Response Body

200	OK	{
		"signatures":
		[
		"KeTob5gl26S2tmXjqN…MRGtoew=="
		]
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Missing (or invalid type) string parameter SAD"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid parameter SAD"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Missing (or invalid type) string parameter credentialID "
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid parameter credentialID"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Missing (or invalid type) array parameter hash"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Empty hash array"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid Base64 hash string parameter"
		}

400	Bad Request	{ "error": "invalid_request", "error_description": "Missing (or invalid type) string parameter signAlgo"

		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Missing (or invalid type) string parameter hashAlgo"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid parameter hashAlgo"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid parameter signAlgo"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid digest value length"
		}

400	Bad Request	{
		"error": "invalid_otp",
		"error_description": "The OTP is invalid"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Signing certificate 'O=[organization],CN=[common_name]' is expired."
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "Invalid parameter clientData "
		}

Table 7 – Signatures/SignHash

3.2.7 OAuth2/Authorize

It does not specify a regular CSC API method, but rather the URI path component of the address of the web page allowing the user to sign-in to the remote service to authorize the signature application or to authorize a credential. The complete URL to invoke the OAuth 2.0 authorization server is obtained by adding oauth2/authorize to the base URI of the authorization server as returned in the oauth2 parameter by the info method and it does not necessarily include the base URI of the remote service API.

Error! Hyperlink reference not valid.authorize	
HTTP Verb	GET
Content-Type	
Accept	
Parameters	**//Service Auhorisation** response_type=code&

	client_id=samples_test_client& redirect_uri =http://localhost:8777& scope=service& lang=en-UK& state=123456& profile_id=adss:ras:profile:001 **// Credentials Authorisation** response_type=code& client_id=samples_test_client& redirect_uri =http://localhost:8777& scope=credential& credentialID=sample-key& numSignatures=2& hash= MTIzNDU2Nzg5MHF3ZXJ0enVpb3Bhc2RmZ2hqa2zDtnl4& state=12345

Status Code	Message	Response Body
302	Found	Location: <OAuth2_redirect_uri> ? code=12234&state=121212

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

Table 8 – OAuth2/Authorize

3.2.8 OAuth2/Token – Authorization Code Flow

Obtain an OAuth 2.0 bearer access token from the authorization server by passing the authorization code or refresh token returned by the authorization server after a successful user authentication, along with the client ID and client secret in possession of the signature application.

Error! Hyperlink reference not valid.	
HTTP Verb	POST

Content-Type	application/x-www-form-urlencoded
Accept	application/json
Request Header	
profile_id	adss:ras:profile:001
Request Body	grant_type=authorization_code& client_id=samples_test_client& client_secret=jr67gj0h76gr83nf8734nj59g4he895jh87nr& code=ssd34343& redirect_uri=http://locahost:8777

Status Code	Message	Response Body

200	OK	// Service Authorisation Response
		{
		"access_token":"KeTob5gl26S2tmXjqN…MRGtoew=="
		"refresh_token":"KeTob5gl26S2tmXjqN…MRGtoew=="
		"token_type":"Bearer"
		"expires_in":"3600"
		}
		// Credentials Authorisation Response
		{
		"access_token":"KeTob5gl26S2tmXjqN…MRGtoew=="
		"token_type":"SAD"
		"expires_in":"3600"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

Table 9 – OAuth2/token

3.2.9 OAuth2/Token – Client Credentials Flow

Obtain an OAuth 2.0 bearer access token from the authorization server by passing the client credentials which is pre-assigned by the authorization server to the signature application along with the client ID and client secret in possession of the signature application.

Error! Hyperlink reference not valid.

HTTP Verb	POST	
Content-Type	application/x-www-form-urlencoded	
Accept	application/json	
Request Headers		
profile_id	adss:ras:profile:001	
user_id	Jhon Wick	
Request Body	grant_type=client_credentials& client_id=samples_test_client& client_secret=jr67gj0h76gr83nf8734nj59g4he895jh87nr	
Status Code	**Message**	**Response Body**

200	OK	// Service Authorisation Response
		{
		"access_token":"KeTob5gl26S2tmXjqN…MRGtoew=="
		"refresh_token":"KeTob5gl26S2tmXjqN…MRGtoew=="
		"token_type":"Bearer"
		"expires_in":"3600"
		}

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

Table 10 – OAuth2/token

3.2.10 OAuth2/Revoke

Calculate the remote digital signature of one or multiple hash values provided as an input. This method requires providing credential authorization in the form of Signature Activation Data (SAD).

Error! Hyperlink reference not valid.oauth2/revoke	
HTTP Verb	POST
Content-Type	application/x-www-form-urlencoded
Accept	application/json
Request Body	token= jr67gj0h76gr83nf8734nj59g4he895jh87nr& token_type_hint=access_token/refresh_token&

	client_id=samples_test_client& client_secret=jr67gj0h76gr83nf8734nj59g4he895jh87nr

Status Code	Message	Response Body
200	OK	

400	Bad Request	{
		"error": "invalid_request",
		"error_description": "The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed."
		}

Table 11 – OAuth2/revoke

4 Mobile Application Interfaces

A mobile app must interact with ADSS RAS to handle these services:

- Registration of the user's mobile device for remote authorisation
- Allowing the user to receive, authorise and send remote signing requests/responses

Mobile apps integrate with ADSS RAS Service using RESTful APIs. This section details each API method.

4.1 Authenticate Application

This call returns the meta information and the list of endpoints implemented by the service.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/x-www-form-urlencoded	
Accept	application/json	
Request Body	client_id=samples_test_client & client_secret=121212 & grant_type=client_credentials	
Status Code	**Message**	**Response Body**
200	OK	{ "access_token":"2YotnFZFEjr1zCsicMWpAA", "expires_in":3600 }
400	Bad Request	For Error information in client credentials request refer OAuth RFC 6749 at: https://tools.ietf.org/html/rfc6749#section-5.2

Table 1 – Authenticate Application

Item Details	
Name	**Description**
Request Parameters	
client_id	Client ID is the created by the business application in ADSS and used to authenticate the application.
client_secret	Client Secret is the created by the business application in ADSS and used to authenticate the application.
grant_type	Grant type would be the client credential.
Response Parameters	

access_token	It will return the client access token after the authentication of client ID and secret.
expires_in	It's token expiry mentioned in the seconds.

4.2 Authenticate User

This call initiates the user authentication on for client application. The following authentication methods can be configured:

- Authenticate user with OTP(s) (Either SMS or Email or Both SMS/Email)
- Authenticate user with QR Code
- No Authentication

Authenticate user with OTP(s):

If this option is enabled, it means user will be authenticated using the OTPs. RAS will send a request to SAM to generate either a single or two OTPs according to the option "SMS OTP" and "Email OTP" selected in the RAS Profile. The SAM will generate the OTP(s) and return to RAS that will send the OTP(s) to user's mobile number or email. It will also return the mobile number and email of the user to client application that will be an indication that the user will be authenticated using the OTPs and these OTPs will be verified with another RAS API.

Authenticate user with QR Code:

If this option will be selected, the RAS Service will instantly return the response to client application with authType:qrCode. That will be an indication that the user will be authenticated using a QR Code so the client app will ask the user to go to QR code page and scan the QR code. Once the mobile app scans the QR Code, it will send this to RAS for verification by calling another API (Verify QR Code).

No Authentication:

In this case, the RAS will send a request to SAM to check if user ID is registered. After getting confirmation from SAM the RAS will generate the access and refresh tokens for this user and return to client application. The presence of access token in response will be an indication for client app that the user has been authenticated.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Authorization	Bearer {application_access_token}	

Request Body	{
	"user_id": "John_Doe",
	}	

Response Header		
authentication_methods	true	
Status Code	**Message**	**Response Body**
200	OK	If OTP Authentication is configured in RAS Profile.

		If both SMS and Email OPTs are sent to user:
		{
		"auth_type": "OTP",
		"otp_info": [
		{
		"otp_type": "EMAIL_OTP",
		"sent_to": "john.doe@sample.som",
		},
		{
		"otp_type": "SMS_OTP",
		"sent_to": "+448007720442"
		}
		]
		}

		If one OTP will be sent on user email:
		{
		"auth_type": "OTP",
		"otp_info": [
		{
		"otp_type": "EMAIL_OTP",
		"sent_to": "john.doe@sample.som",
		}
		]
		}

		If one OTP will be sent to user's mobile:
		{
		"auth_type": "OTP",
		"otp_info": [
		{
		"otp_type": "SMS_OTP",
		"sent_to": "+448007720442"
		}
		]
		}

		If QR Code authentication is configured:
		{
		"auth_type": "QR_CODE",
		}

		If no authentication is configured:
		{
		"auth_type": "NO_AUTHENTICATION",
		"token_info": {
		"access_token":"eyJhbGciOiJIUzIda……96RDo",
		"refresh_token":"eyJhbGciOiJIUzsac……ymjGp-E",
		"token_type":"bearer",
		"expires_in":3600
		}

		}
400	Bad Request	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
user_id	User ID as registered by the business application in ADSS Server SAM.
Response Parameters	
auth_type	Authentication type configured in RAS Profile. The following values can be found in this parameter: - OTP - QR_CODE - NO_AUTHENTICATION **Note:** The values OTP, QR_CODE and NO_AUTHENTICATION are case-sensitive
otp_info	Contains information related to the types of OTPs (Email/SMS) and the mobile number and email of the user.
token_info	Contains the OAuth access & refresh tokens and the expiry.
error_code	The error code.
error_description	Error description message.

Table 2 – Authenticate User

4.3 Verify OTPs

If the OTP authentication will be enabled in RAS Service, the user will receive either one or two OTPs on his mobile number or email. The user will provide these OTPs to this API. After successful OTPs verification, access and refresh tokens are returned.

https://server:8778/adss/service/ras/v1/authentication/otp/verify	
HTTP Verb	POST
Content-Type	application/json
Accept	application/json
Authorization	Bearer {application_access_token}
Request Body	If user had received two OTPs: { "user_id": "User ID",

	"otp_info": [{
	"otp": "258456987",
	"otp_type": "SMS_OTP"
	},{
	"otp": "258456987",
	"otp_type": "EMAIL_OTP"
	}]
	}	

	If user had received a single OTP on mobile:
	{
	"user_id": "User ID",
	"otp_info": [{
	"otp": "258456987",
	"otp_type": "SMS_OTP"
	}]
	}	

	If user had received one OTP via email:
	{
	"user_id": "User ID",
	"otp_info": [{
	"otp": "258456987",
	"otp_type": "EMAIL_OTP"
	}]
	}	

Status Code	**Message**	**Response Body**

200	Ok	{
		"access_token":"eyJhbGciOiJIUzI1NiIsInR5……EFRJ96RDo",
		"refresh_token":"eyJhbGciOiJIUzI1NiIsInR5……heyXmjGp-E",
		"token_type":"bearer",
		"expires_in":3600,
		}

400	Bad Request	
401	Unauthorized	

403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
user_id	User ID as registered by the business application in ADSS Server SAM.
otp_info	Array of otp(s) sent to user via SMS/EMAIL
otp	OTP received by the user on his/her mobile/email
otp_type	Type of the OTP i.e. SMS or Email
Response Parameters	
access_token	It will return the client access token after the authentication of client ID and secret.
refresh_token	Refresh token will be used to get the new access token without send the user credentials.
expires_in	It's token expiry mentioned in the seconds.
error	The error code
error_description	Error description message

4.4 Renew Access Token

This call allows the renewal of an expired access token by providing the refresh token.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/x-www-form-urlencoded	
Accept	application/json	
Request Body	grant_type=refresh_token&refresh_token=tGzv3JOkF0XG5Qx	
Status Code	**Message**	**Response Body**
200	OK	{ "access_token":"2YotnFZFEjr1zCsicMWpAA", "refresh_token":"TRVFHTHcedfJGJFLGKKJ", "expires_in":3600,

		}
400	Bad Request	For Error information in client credentials request refer OAuth RFC 6749 at: https://tools.ietf.org/html/rfc6749#section-5.2

Item Details	
Name	**Description**
Request Parameters	
refresh_token	Refresh token which client application already received while user authentication.
Response Parameters	
access_token	New access token
refresh_token	New refresh token to cover in-activity time by the logged-in user
expires_in	Access token expiry in seconds
error_code	The error code
error_description	Error description message

Table 4 – Renew Access Token

4.5 Device Registration

Once we get the access token we can use subsequent APIs. This API is used to register user's device for remote signature authorisation purposes and request a certificate for the device's authorisation public key. **User needs to generate the keypair in device e.g mobile device's software and hardware(Secure Enclave) and generate the CSR(Certificate Signing Request)**. Once the CSR is generated, it will be sent in this API which will return the certificate that is used in the signing the authorisation response message when a user authorises a remote signing operation.

https://<server>:8778/adss/service/ras/v1/authorization/certificate	
HTTP Verb	POST
Content-Type	application/json
Accept	application/json
Authorization	Bearer {access_token}

Request Body	{
	"csr": "MIICxDCCAawCAQAwfzELM[....]5f52oQ==",
	"device": {
	"device_id": " ASJMMN5389FF ",
	"device_name": "IPHONE X",
	"secure_element": true,
	"biometric": true,
	}
	}

Status Code	**Message**	**Response Body**

200	OK	{
		"alias": "hvcNAU+qCdXzADEA"
		"certificate": "MIIItAYJKocNA[...]ZU+qCdXzADEA"
		}

400	Bad Request	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
csr	The CSR for the device. Remove the tags in the CSR "-----BEGIN CERTIFICATE REQUEST-----" and "-----END CERTIFICATE REQUEST-----". The CSR should be in a single line.
device_name	Alias of the device. Later can be renamed.
device_id	A unique device ID, that is obtained from the mobile app. For example, UUID random number.
secure_element	Must set to "True" if device has a hardware Secure Element/Enclave. It would be used when you use this API from mobile device.
biometric	Must set to "True" if device has biometric feature available on the device. It can be TouchID, FaceID, Fingerprint etc. It can be used when you call this API from mobile device.
Response Parameters	
certificate	Certificate generated for the device in base64 encoded format.
Alias	The certificate alias.
error_code	The error code.
error_description	A string with the description of the error_code.

Table 5 – Request Device/User Certificate

4.6 List Registered Devices

This method retrieves all the devices that the user has registered for use in remote authorised signing operations.

https://<server>:8778/adss/service/ras/v1/authorization/devices?user_id={user_id}	
HTTP Verb	GET
Accept	application/json
Authorization	Bearer {application_access_token}

Request Body		
Status Code	**Message**	**Response Body**

200	OK	[
		{
		"device_id": "id-001",
		"device_name": "IPhone",
		"secure_element": true,
		"biometric": true,
		},
		{
		"device_id": "id-002",
		"device_name": "Samsung",
		"secure_element": true,
		"biometric": true,
		}
		]

400	Bad Request	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
Response Parameters	
device_id	Device ID.
device_name	Device alias.
secure_element	“True” if device has secure element/enclave.
biometric	“True” if device has biometric feature available on the device. It can be TouchID, FaceID, Fingerprint etc. It can be used when the Device Registration is done from the mobile device.
error_code	The error code.
error_description	Error description message

Table 7 – List Registered Devices

4.7 Delete Device

This API deletes a user’s device in RAS Service identified by {device_id}. A client application would use this interface to delete a user’s device.

https://server:8778/adss/service/ras/v1/authorization/devices/{device_id}	
HTTP Verb	DELETE
Accept	application/json

Access Token	Bearer {user_access_token}	
Request Body		
Status Code	**Message**	**Response Body**
200	OK	
404	Not Found	
403	Forbidden	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{device_id}	Device ID which is already registered in ADSS Server.
Response Parameters	
error	The error code
error_description	Error description message

Table 7 - Delete Device

4.8 Get Pending Authorisation Request

This method returns a pending authorisation request. That is, where the business application has requested a signing operation that requires user authorisation.

https://<server>:8778/adss/service/ras/v1/authorization/request		
HTTP Verb	GET	
Accept	application/json	
Authorization	Bearer {access_token}	
Request Body		
Status Code	**Message**	**Response Body**
200	OK	{ "transaction_id": "932469001521668267", "request": "PEFDRj48Y2VydEFs[...]9BQ0Y+", "hash_algorithm": "SHA256" }
400	Bad Request	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
Response Parameters	
request_id	Request ID of the signature.
request	Pending authorisation request in base64 form - this is the "object" together with some additional data e.g. Device ID added and signed by the client app using the authorisation key. Value must be decoded before signing operation. After decoding the request, it will be look like this. <AuthorisationData> <OriginatorID>Virtual_CSP_Client</OriginatorID> <UserID>olcayatli@gmail.com</UserID> <CertificateID>416edc72-6c63-45aa-bb34a373102234df</CertificateID> <TransactionID>980551837300673581</TransactionID> <Salt>924552495291565632</Salt> <MetaData> <DisplayText>Data to be displayed</DisplayText> <DeviceID></DeviceID> </MetaData> <Documents> <Document id="b81e040a-a4d8-4134-92ff-2d4bf5e9116d"> <Name>b81e040a-a4d8-4134-92ff-2d4bf5e9116d</Name>

	<DigestValue>ypkP9L2tZO2JdfNr4X4X5SRur529uJqykdc5q5HDSiLNiYcLrysO0S/H31yb8QZS
SOBYsFlVSj9/SKUqrhsUC5oEc/gr</DigestValue> </Document> </Documents> <ValidityPeriod> <ValidFrom>2019-12-07T18:25:37</ValidFrom> <ValidTo>2019-12-07T18:42:17</ValidTo> </ValidityPeriod> <Signature> <DigestMethod>SHA256</DigestMethod> </AuthorisationData>
hash_algorithm	Hash algorithm to be used for signing the authorized remote signing request.
error_code	The error code.
error_description	Error description message.

Table 8 – Get Pending Signature Request

4.9 Authorise a Pending Request

This method authorises a pending request by sending the signed Signature Activation Data (SAD) against the pending authorisation request received as described above. That is, the value returned in section 4.8 above (together with some additional data) must be signed on the mobile device and returned here. The returned value must be base64 encoded. The hash algorithm is as returned in section 4.8 above, and the same value is returned here in the body request.

https://<server>:8778/adss/service/ras/v1/authorization/request/{request_id}		
HTTP Verb	PUT	
Authorization	Bearer {access_token}	
Content-Type	application/json	
Accept	application/json	

Request Body	{
	"request": "PEFDRj48Y2VydEFs[...]9BQ0Y+",
	“hash_algorithm": “SHA256”
	}	

Status Code	**Message**	**Response Body**
200	OK	
400	Bad request	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{request_id}	Request ID received in section 4.8 above
request	You need to decode the request, signed with private key and add the signature in the request. This signed request also called the SAD(Signature Activation Data) <AuthorisationData> <OriginatorID>Virtual_CSP_Client</OriginatorID> <UserID>olcayatli@gmail.com</UserID> <CertificateID>416edc72-6c63-45aa-bb34a373102234df</CertificateID> <TransactionID>980551837300673581</TransactionID> <Salt>924552495291565632</Salt> <MetaData> <DisplayText>Data to be displayed</DisplayText> <DeviceID>ad1e60a6-5e23-4b52-a127-27f41c224c05</DeviceID> </MetaData> <Documents> <Document id="b81e040a-a4d8-4134-92ff-2d4bf5e9116d"> <Name>b81e040a-a4d8-4134-92ff-2d4bf5e9116d</Name>

	<DigestValue>ypkP9L2tZO2JdfNr4X4X5SRur529uJqykdc5q5HDSiLNiYcLrysO0S/H31yb8QZS
SOBYsFlVSj9/SKUqrhsUC5oEc/gr</DigestValue> </Document> </Documents> <ValidityPeriod> <ValidFrom>2019-12-07T18:25:37</ValidFrom> <ValidTo>2019-12-07T18:42:17</ValidTo> </ValidityPeriod> <Signature> <DigestMethod>SHA256</DigestMethod> <SignatureValue>MEUCID/kiJWAIqzwOp/hi+FUbJwsjdcsEoBNwlIF8sXA8XbDAiEAkGgQomyoJ2iR0ra9KGBFW/zXi6tbsn5M49YiaPNc+L8=</SignatureValue> </AuthorisationData>
hash_algorithm	Hashing algorithm used for signing SAD.
Response Parameters	
error_code	The error code
error_description	Error description message

Table 9 – Confirm a Pending Signature Request

4.10 Cancel a Pending Authorisation Request

This method cancels a pending authorisation request. That is, the user decides to decline the authorisation request sent to the mobile device.

Error! Hyperlink reference not valid.		
HTTP Verb	DELETE	
Authorization	Bearer {access_token}	
Accept	application/json	
Request Body		
Status Code	**Message**	**Response Body**
200	OK	
400	Bad request	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{request_id}	Request ID for which user cancelled the authorization.

Response Parameters	
error_code	The error code.
error_description	Error description message.

Table 10 – Cancel a Pending Signature Request

4.11 User Profile

This API is used to get user's profile information from ADSS.

https://<server>:8778/adss/service/ras/v1/users/profile		
HTTP Verb	GET	
Content-Type	application/json	
Accept	application/json	
Authorization	Bearer {access_token}	
Status Code	**Message**	**Response Body**
200	OK	{ "user_id": "Alice", "user_name": "Alice", "app_name": "samples_test_client", "user_email": "abc@ascertia.com", "user_mobile": "+9230XXXXXXXX" "last_updated_at": "2020-12-15 18:44:04" }
400	Bad Request	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
Response Parameters	
user_id	User ID as registered by the business application in ADSS Server SAM.
user_name	User name as registered by the business application in ADSS Server SAM.
user_email	User email as registered by the business application in ADSS Server SAM.
user_mobile	User mobile number as registered by the business application in ADSS Server SAM.
error_code	The error code.
error_description	A string with the description of the error_code.

Table 11 – User Profile

4.12 Get Device Registration Settings

This interface is used to get the device settings at the time of device registration, i.e. to check which key length & key type will be used to generate an authorisation key either in Device Secure Enclave or Software KeyStore/KeyChain.

http://server:8778/adss/service/ras/v1/users/profile/device/settings		
HTTP Verb	GET	
Accept	application/json	
Authorization	Bearer {access_token}	
Request Body		
Status Code	**Message**	**Response Body**
200	OK	{ "device_key_type": "ECDSA", "device_key_size": 256, "secure_element_required": true, "biometric_required": true }
400	Bad Request	
403	Forbidden	
404	Not Found	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
Response Parameters	
device_key_type	Key pair type to be generated in mobile device, e.g. RSA Possible values are RSA & ECDSA
device_key_size	Key pair size e.g. 2048
secure_element_required	If set “TRUE” then the authorisation key pair must be generated inside device secure enclave otherwise software key store or KeyChain be used for key pair generation. If this flag is set to TRUE and the device does not support a Secure Element, then an error will be returned. It can be used when Device Registration is done by mobile device.
biometric_required	If set TRUE then device must have biometric (fingerprint, TouchID, FaceID etc.) support available on it otherwise Device

	PIN/Passcode be used to protect the generated keys. If this flag is set to TRUE and the device doesn't support biometric functionality, then an error will be returned. It can be used when Device Registration is done by mobile device.
error_code	The error code
error_description	Error description message

Table 12 - Get Device Registration Settings

4.13 Generate QR Code

This API will be used by the business application to generate a QR Code using the RAS Service. The RAS Service will generate a QR Code image and send in response.

Error! Hyperlink reference not valid.		
HTTP Verb	GET	
Content-Type		
Accept	application/json	
Request Body		
Status Code	**Message**	**Response Body**
200	OK	{ "format":"png", "size":"264", "qr_code":"<base64 encoded image>" }
400	Bad Request	
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
{ClientID}	Client ID that is registered in the ADSS client manager.
{userID}	User ID for whom the QR Code will be generated
size	(Optional) Size of the QR Code image in pixels. Since QR Code is in a square shape the parameter "size" will be used for both width and height of the image. Default is 264 pixels

format	(Optional) format of the QR Code image e.g. png/jpeg/bmp etc. Default will be "png". The following formats are supported: • png • jpg • bmp • jpeg • wbmp • gif
Response Parameters	
Format	Format of the QR Code image e.g. png/bmp/jpg etc.
Size	Size of the QR Code image
qr_code	base64 encoded image of the QR Code

Table 13 – Generate QR Code

4.14 Verify QR Code

This API will be used to verify a QR Code by RAS Service if user set the authentication mechanism QR code in RAS profile. Client app can use the QR code reader to scan the QR code. If QR code is verified successfully, the RAS Service will return the access and refresh tokens in response.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Authorization	Bearer {application_access_token}	

Request Body	{
	"user_id":"jhon123",
	"qr_code":"Information extracted from QR code"
	
	}	

Status Code	**Message**	**Response Body**

200	OK	{
		"access_token":"2YotnFZFEjr1zCsicMWpAA",
		"refresh_token":"TRVFHTHcedfJGJFLGKKJ",
		"expires_in":3600
		}

401	Unauthorized	If QR code is invalid or expired
400	Bad Request	

500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
qr_code	Information extracted from QR code
Response Parameters	
access_token	It will return the client access token after the authentication of client ID and secret.
refresh_token	Refresh token will be used to get the new access token without send the user credentials.
expires_in	It's token expiry mentioned in the seconds.

Table 14 – Verify QR Code

4.15 Register Device for Push Notification

This API is used to register the mobile device for push notification by RAS Service. It takes the device token from the mobile application and stores in ADSS RAS to send the push notification while generating the authorization request.

Error! Hyperlink reference not valid.		
HTTP Verb	POST	
Content-Type	application/json	
Accept	application/json	
Authorization	Bearer {user_access_token}	
Request Body	{ "device_token":"2YotnFZFEjr1zCsicMWpAA ", "os_type":"ANDROID | IOS" }	
Status Code	**Message**	**Response Body**
200	OK	
401	Unauthorized	Invalid or expired user access token
400	Bad Request	Device token is missing
500	Internal Server Error	

Item Details	
Name	**Description**
Request Parameters	
device_token	It's the device token which needs to be sent on server to register the mobile device for push notification.
os_type	OS type can be Android or iOS.

Table 15 – Register Device for Push Notification

5 Signature Activation Data (SAD) – Body Structure

The body structure of SAD XML is explained in the table below:

Body Structure	<AuthorisationData> <OriginatorID>Virtual_CSP_Client</OriginatorID> <UserID>olcayatli@gmail.com</UserID> <CertificateID>416edc72-6c63-45aa-bb34a373102234df</CertificateID> <TransactionID>980551837300673581</TransactionID> <Salt>924552495291565632</Salt> <MetaData> <DisplayText>Data to be displayed</DisplayText> <DeviceID>ad1e60a6-5e23-4b52-a127-27f41c224c05</DeviceID> </MetaData> <Documents> <Document id="b81e040a-a4d8-4134-92ff-2d4bf5e9116d"> <Name>b81e040a-a4d8-4134-92ff-2d4bf5e9116d</Name> <DigestValue>ypkP9L2tZO2JdfNr4X4X5SRur529uJqykdc5q5HDSiLNiYcLrysO0S/H31yb8QZS
SOBYsFlVSj9/SKUqrhsUC5oEc/gr</DigestValue> </Document> </Documents> <ValidityPeriod> <ValidFrom>2019-12-07T18:25:37</ValidFrom> <ValidTo>2019-12-07T18:42:17</ValidTo> </ValidityPeriod> <Signature> <DigestMethod>SHA256</DigestMethod> <SignatureValue>MEUCID/kiJWAIqzwOp/hi+FUbJwsjdcsEoBNwlIF8sXA8XbDAiEAkGgQomyoJ2iR0ra9KGBFW/zXi6tbsn5M49YiaPNc+L8=</SignatureValue> </AuthorisationData>

6 Get Profile Information

This interface returns the information of a RAS profile e.g. all settings configured in that profile. The business application will send the profile ID and client ID in request and RAS will return the information of that profile in response.

Exposed for: Business Applications

<table>
<tr><td colspan="3">https://server:8779/adss/service/ras/v1/profile/info</td></tr>
<tr><td>HTTP Verb</td><td colspan="2">POST</td></tr>
<tr><td>Accept</td><td colspan="2">application/json</td></tr>
<tr><td>Request Body</td><td colspan="2"><pre>{
"profile_id": "adss:ras:profile:001",
"client_id": "samples_test_client"
}</pre></td></tr>
<tr><td>Status Code</td><td>Message</td><td>Response Body</td></tr>
<tr><td>200</td><td>OK</td><td><pre>{

    "profile_id": "adss:ras:profile:001",
    "profile_name": "adss:ras:profile:001",
    "profile_status": "ACTIVE",
    "basic_authentication": true,
    "oauth2_authentication": true,
    "credentials_authorisation_method": "IMPLICIT",
    "authentication_with_qr_code": false,
    "no_authentication": false,
    "sam_profile": {
    "profile_id": "adss:sam:profile:001",
    "profile_name": "adss:sam:profile:001",
    "profile_status": "ACTIVE",
    "crypto_profile": "utimaco",
    "key_type": "RSA",
    "key_size": 2048,
    "kak_size": 0,
    "signature_padding_scheme": "PKCS1",
    "compute_hash_at_signing": true,
        "hash_algorithm": "SHA256",
        "bulk_signing_allowed": false,
        "number_of_hashes": 0,
        "device_key_type": "ECDSA",
        "device_key_size": 256,
        "secure_element_required": true,
        "biometric_required": true
  },
    "saml_asscertion": {
        "idp_signing_certificate": ""
        "identify_user_id": "SAML_ATTRIBUTE_NAME",
        "identify_user_attribute": "abc"
    },
    "authentication_with_otp": {
        "sms_otp": true,
        "email_otp": true</pre></td></tr>
</table>

		} }
400	Bad Request	
403	Forbidden	
404	Not Found	
500	Internal Server Error	

Item Details	
Name	Description
Request Parameters	
client_id	Client's Originator ID which is configured in ADSS Console > Client Manager. Client ID is required because only legitimate clients can get a profile's information
profile_id	Profile ID whose information is required in response.

Table 16 – Get Profile Information

7 Error Code List

Below table contains the error codes for RAS business and mobile interfaces.

Errors	
error	**error_description**
58001	An internal server error occurred while processing the request - see the RAS service debug logs for details
58002	Service is not available - Try later
58003	Failed to process request - RAS service is not enabled in license
58004	Failed to process request - RAS service license has expired
58005	Failed to process request - RAS service is not enabled in system
58006	Failed to process request - RAS service is not allowed
58007	Failed to process request - Client ID does not exist
58008	Failed to process request - User ID does not exist
58009	Failed to process request - User ID already exists
58010	Failed to process request - Key alias does not exist
58011	Failed to process request - Transaction ID does not exist
58012	Failed to process request - Client ID not found in the request
58013	Failed to process request - User ID not found in the request
58014	Failed to process request - Key alias not found in the request
58015	Failed to process request - Subject DN not found in the request
58016	Failed to process request - User password not found in the request
58017	Failed to process request - Key length not found in the request
58018	Failed to process request - Key algorithm not found in the request
58019	Failed to process request - User name not found in the request
58020	Failed to process request - User password not found in the request
58021	Failed to process request - User mobile number not found in the request
58022	Failed to process request - Key alias exceeds the allowed limit
58023	Failed to process request - User ID exceeds the allowed limit
58024	Failed to process request - User name exceeds the allowed limit
58025	Failed to process request - User password exceeds the allowed limit
58026	Failed to process request - Invalid user mobile number
58027	Failed to process request - Invalid user email
58028	Failed to process request - Invalid user status
58029	Failed to process request - RAS profile does not exist or marked inactive
58030	Failed to process request - User certificate not found in the request
58031	Failed to process request - Profile ID not found in the request
58032	Failed to process request - Invalid client ID
58033	Failed to process request - User's new password not found in the request
58034	Failed to process request - SMS OTP not found in the request
58035	Failed to process request - Email OTP not found in the request
58036	Invalid string parameter - refresh_token
58037	Failed to process request - Invalid refresh token

58038	Failed to process request - Invalid access token
58039	The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed.
58040	Failed to process request - Basic authentication is not enabled in RAS profile
58041	Failed to process request - SAML authentication is not enabled in RAS profile
58042	Failed to process request - Missing (or invalid type) string parameter token
58043	Failed to process request - Invalid string parameter token_type_hint
58044	Failed to process request - Invalid string parameter token
58045	Failed to process request - Client ID is not configured for certification service settings in RAS service manager
58046	Failed to process request - Certification profile is not configured for certification service settings in RAS service manager
58047	Failed to process request - Certification service address is not configured for certification service settings in RAS service manager
58048	Failed to process request - Unable to get device certificate from certification service
58049	Failed to generate access token - HMAC Key not configured in RAS service manager
58050	Failed to process request - User Email not found in the request
58051	Failed to process request - Client ID is not configured for default settings in RAS service manager
58052	Failed to process request - Device ID not found in the request
58053	Failed to process request - Push notification token not found in the request
58054	Failed to process request - OS type not found in the request
58055	Missing (or invalid type) string parameter credentialID
58056	Missing (or invalid type) integer parameter numSignatures
58057	Invalid parameter numSignatures
58058	Invalid request parameter - numSignatures doesn't match with no of hashes in hash array
58059	Invalid request parameter - no of documents in clientData doesn't match with no of hashes in hash array
58060	Missing parameter hash
58061	Failed to authorise user credentials - Request timeout for mobile authorisation
58062	Failed to authorise user credentials - User cancelled mobile authorisation
58063	Invalid parameter credentialID
58064	Missing (or invalid type) string parameter SAD
58065	Invalid parameter SAD
58066	Empty hash array
58067	Invalid Base64 hash string parameter
58068	Missing (or invalid type) string parameter signAlgo
58069	Invalid parameter signAlgo
58070	Missing (or invalid type) string parameter hashAlgo
58071	Invalid parameter hashAlgo

58072	Failed to validate SAML assertion - Invalid base64 data
58073	Failed to validate SAML assertion - Not comply with SAML 2.0 schema
58074	Failed to validate SAML assertion - Unable to parse SAML assertion
58075	Failed to validate SAML assertion - Validity period expired or not yet valid
58076	Failed to validate SAML assertion - Server certificate does not match with the certificate configured in RAS Profile
58077	Failed to validate SAML assertion - Multiple or no attributeValue found
58078	Failed to validate SAML assertion - Invalid Signature
58079	Failed to process request - User status is blocked or inactive
58080	Failed to process request - Certificate chain not found in the request
58081	Failed to process request - Invalid certificate chain
58082	Failed to process request - Client secret not found in the request
58083	Failed to authorise user credentials - An internal server error occurred during signature computation
58084	Failed to process request - Device CSR not found in the request
58085	Failed to process request - Invalid device CSR
58086	Failed to process request - Device information not found in the request
58087	Failed to process request - Device ID not found in the request
58088	Failed to process request - Device name not found in the request
58089	Failed to process request - SAD not found in the request
58090	Failed to process request - Request ID not found in the request
58091	Failed to process request - Invalid request
58092	Failed to process request - Either request ID is invalid or the transaction is expired
58093	An internal server error occurred - please contact your service provider
58094	Failed to process request - User mobile exceeds the allowed limit
58095	Failed to process request - User email exceeds the allowed limit
58096	Failed to process request - Configurations for SMS/Email OTP(s) not available
58097	Failed to process request - No OTP(s) found in request
58098	Failed to process request - QR Code authentication is not allowed for this RAS profile
86000	Failed to authenticate client - TLS client authentication certificate has expired
86001	Failed to authenticate client - TLS certificate CN does not match with Client ID
86002	Failed to authenticate client - TLS client certificate is revoked
86003	Failed to authenticate client - revocation status for TLS client certificate is unknown
86004	Failed to authenticate client - Client ID does not match with the client identified by TLS client certificate
86005	Failed to authenticate client - TLS client certificate does not match with the configured client certificate
86006	Failed to authenticate client - request signing certificate has expired
86007	Failed to authenticate client - request signing certificate is revoked

86008	Failed to authenticate client - revocation status for request signing certificate is unknown
86009	Failed to authenticate client - request signing certificate does not match with the configured client certificate
86010	Failed to authenticate client - Client ID does not match with the client identified by the request signing certificate
86011	Failed to authenticate client - Client ID does not exist
86012	An error occurred while communicating with database - see the service debug logs for details
86013	An error occurred when checking the certificate revocation status - see the service debug logs for details
86014	An internal error occurred while authenticating the client - see the service debug logs for details
86015	Failed to authenticate client - Client ID is not found in the request
86016	Failed to process request - Request signing certificate is not trusted
86017	Failed to authenticate client - client is marked inactive
86018	Failed to authorise client - service is not allowed to this client
86019	Failed to authorise client - service profile does not exist
86020	Failed to authorise client - service profile is inactive
86021	Failed to authorise client - profile is not allowed to this client
86022	Failed to authorise client - default profile not configured and neither found in request
86023	Failed to authorise client - default profile is inactive
86024	Failed to authorise client - client secret is invalid
internal_error	An internal server error occurred while processing the request
invalid_csr	CSR is invalid
invalid_otp	OTP is either invalid or expired
missing_csr	CSR is missing in the request
missing_device_id	Device ID is missing in the request
missing_device_info	Device information is missing in the request
missing_device_name	Device name is missing in the request
missing_request_id	Request ID is missing in the request
refresh_token_revoked	Refresh token is either invalid or expired

Table 11 - Error Codes

*** End of Document ***