**A picture containing night sky

Description automatically generated**

**A picture containing icon

Description automatically generated**

**SUBEX**BEL\_eSign\_Module\_FLow

BEL eSign Module FLow

Wednesday, October 5, 2022 version 2022\_1005

**A picture containing night sky

Description automatically generated**

Table of Contents

[1. Introduction 3](#_Toc77261409)

[2. Deployment Architecture 4](#_Toc77261413)

[3. API Sequence 6](#_Toc77261409)

[4. ASP Flow 8](#_Toc77261413)

[5. ESP Flow 11](#_Toc77261409)

[6. OTP Generation Service 13](#_Toc77261413)

[7. Project Management Approach and Methodology 14](#_Toc77261413)

[8. Error Codes 15](#_Toc77261409)

1. Introduction

This Document provides and overview of Sectrio’ s approach towards the implementation of eKYC at BEL.

1. Deployment Architecture

Graphical user interface

Description automatically generated

**Application Service Provider (ASP):** An organization or an entity using e-Sign service as part of their application to digitally sign the content. Examples include Government Departments, Banks and other public or private organizations. Currently there is no process of registration of ASP. ASP may contact the ESP (e-Sign Service Provider) directly to avail the service within its framework.

**End-User**: An Individual using the application of ASP and represents himself/herself for signing the document under the legal framework. For the purposes of KYC with UIDAI, the end-user shall also be the ‘resident’ holding the AADHAAR number. For the purposes of DSC by the CA, the end-user shall also be the ‘applicant/subscriber for digital certificate’, under the scope of IT Act.

**e-Sign Service Provider (ESP):** An organization or an entity providing e-Sign service. ESP is a “Trusted Third Party”, as per the definitions of Second Schedule of Information Technology Act. ESP must be a registered KYC User Agency (KUA) with UIDAI. ESP will facilitate subscriber’s key pair-generation, storing of key pairs on hardware security module and creation of digital signature. ESP can be a Licensed Certifying Authority (CA), by themselves, or must be having an arrangement / integration with a CA for the purpose of obtaining Signature Certificate for the generated Key-pair.

**Certifying Authority (CA):** An organization or an entity licensed under CCA for issuance of Digital Certificate and carrying out allied CA operations.

**UIDAI**: An authority established by Government of India to provide unique identity to all Indian residents. It also runs the e-KYC authentication service for the registered KYC User Agency (KUA).

**Sectrio will do the below functionalities as part of Web, Mobile, Desktop App.**

* Client application asks e-Sign user to sign the document
* Client application creates the document hash (to be signed) on the client side, has / asks-for e-KYC identification Number and asks the e-Sign user to provide consent for certificate generation and signature
* ASP forms the input data for e-Sign API
* ASP redirect to ESP application by submitting request XML
  + ESP validates the calling application and the input.
  + ESP verifies the Digital signature of ASP for e-Sign XML received
  + ESP logs the transaction
  + ESP redirects e-Sign user to e- authentication page
  + ESP performs authentication and get e-KYC information from e-KYC provider
  + ESP show the document hash along with document information to e-Sign user.
  + ESP calls the CA service and gets a Digital Signature Certificate for e-Sign user. The certificate will be a e-KYC class Digital Signature Certificate, which has e-KYC number, Name of the e-Sign user, e-KYC response code, Authentication Type, and Time Stamp embedded.
  + ESP signs the 'document hash' and provides response XML to the ASP by redirecting to ASP's response URL.
* ASP receives the document signature and the e-Sign user's Digital Signature Certificate.
* Client application attaches the signature to the document.
* e-Sign user can accept or reject the signature and DSC

1. API Sequence

A picture containing graphical user interface

Description automatically generated

e-Sign service is exposed as stateless service over HTTPS. Usage of open data format in XML and widely used protocol such as HTTP allows easy adoption and deployment of this service. To support 8 strong end to end security and avoid request tampering and man-in-the-middle attacks, it is essential that encryption of data happens at the time of capture on the capture device.

**Following is the URL format and the parameters for e-Sign service:**

|  |  |
| --- | --- |
| API URL | https://<host:port>/<path>/<ver>/signdoc  Where:  Host is the domain name or IP address of the ESP server.  Port is optional  <path> is optional, and may vary for each ESP  <ver> indicates version of the API. Currently it must be “1.0”  Example:  https://[www.esp.com/esign/1.0/signdoc](http://www.esp.com/esign/1.0/signdoc)  https://portal.esp.com/esign/1.0/signdoc  https://esign.esp.com/1.0/signdoc  https://esp.com/projects/esign/1.0/signdoc  https://www.esp.com:8890/esign/1.0/signdoc |
| Method | POST |
| Content-Type | “application/xml” |
| Post data | A well-formed XML, as per the specifications provided in this  document. |

**Flow of XML**

Diagram

Description automatically generated with medium confidence

1. ASP Flow

**Multi-Factor Authentication Flow:**

Shape

Description automatically generated with medium confidence

Two-factor authentication can significantly reduce risks of compromised accounts by protecting from weak passwords, online identity theft and other online fraud. This paper presents a new easy solution to implement two-factor authentication without affecting user experience by introducing minimum user interaction based on standard Wi-Fi.

**ASP Flow:**

Text

Description automatically generated

Below is the overview of the process, to be carried out by ASP to integrate eSign.

1. Application form submission by ASP.
2. Submission of supporting documents by ASP
3. Acceptance / agreement to terms of eSign service by ASP.
4. Submission of Digital Signature Certificate (public key) by ASP
5. Integration of API in ASP application in testing / preproduction environment of ESP.
6. Conducting audit and submission of Audit report by ASP
7. Grant of production access by ESP

**ASP RESPONSIBILITES:**

* Initiates eSign - calls the ESP signing request API, asks eSign user to sign the document
* Select their required ID type and then specify the value and should construct the signer-id based on ID given by user and selected ID type and ESP.
* ASP client application asks the eSign user id for certificate generation and signature.
* ASP client application creates the document hash (to be signed) on the client side. ASP shall protect the document URL (available within eSign request) from anyone or any system accessing it using URL and from virus, malware, etc.
* ASP shall display (and allow download/print) the document that is to be signed clearly for subscribers to read before signing.
* ASP forms the input data for eSign API and is expected to store this txn ID provided by ESP in their audit log
* In case response is not received by ASP or user session ends within ASP, ASP can check status of signing request using “checkStatus” API using the same txn ID of the request.
* API shall be provided to track the history of requests and response with ESP.

1. ESP FLOW

A picture containing diagram

Description automatically generated

Overview of eSign process ESP responsibilities along with integration with ASP, eKYC and CA

* Dashboard having tabular and graphical representation for performance and transactional details which includes success/unsuccessful response in total and with respect to registered ASPs for admin.
* Report generation in total and with respect to specific ASP which may include Bill generation as per the transaction made by individuals/ organization, ASP and application as per the opted signature plan.
* Management of eKYC agents for both the cases whether it is done at ASP or from ESP.
* Configuration option for eSign CA and eKYC agent.
* The communication between Application service provider and ESP should operate in accordance with eSign API specifications
* API Specifications to provide eSign- Online Electronic Signature Service

**ESP RESPONSIBILITES:**

* ESP validates the calling application and the input - authenticates user for eKYC before eSign through eKYC provider and then pre-populate the username for the convenience.
* ESP verifies the Digital signature of ASP for eSign XML received
* ESP should not allow modification of the username in their screen
* ESP logs the transaction
* ESP acknowledges the request back to ASP by providing an ack response with same txn ID. At this time ASP can close the connection to ESP.
* A unique response code provided by ESP. This is a unique id for the transaction (eSign user authentication & eSign request) provided by ESP. It shall make the transaction traceable, and ASP is expected to store this code in their audit log.
* ESP shall provide necessary option for signer to uncheck any document hash. Such unchecked document hash shall not be signed and shall be returned with an error called “User Rejected”.
* Registration and display of registered ASPs with their details (date of activation, expiry date, no of requests served etc.)
* Bill generation as per the transaction made by individuals/ organization, ASP and application as per the opted signature plan.
* ESP displays e-authentication page (if web flow) or notifies on ESP mobile app to the eSign user.
* ESP should facilitate generation of key pairs on their Hardware Security Module. The key pairs shall be unique to the eSign user. The private key will be destroyed after one time use.
* The private key of the eSign user shall be secured by Hardware security module (HSM).
* HSM of ESP should be separate from that of CAs for DSC issuance.
* Audit Logging Procedure
* All security audit logs, both electronic and non-electronic, shall be retained and made available during compliance audits.
* Retention
* Protection
* Backup
* Collection
* Audit Reports

1. OTP Generation Service

OTP Generation service is exposed as stateless service over HTTPS. This is a proxy service to Aadhaar OTP service. If Aadhaar holder does not have a valid mobile OTP, ASP can use this service to send a mobile OTP to Aadhaar holder’s registered mobile. Note that UIDAI provides Aadhaar holders a mechanism to obtain OTP directly (via SMS, resident portal, mobile app, etc). If Aadhaar holder does not have a valid OTP in possession, ASP application should provide an option to “Request OTP” and execute that request via this API.

**Following is the URL format and the parameters for OTP Generation service**:

|  |  |
| --- | --- |
| **API URL** | https://<host:port>/<path>/<ver>/getotp Where:   * Host is the domain name or IP address of the ESP server. * Port is optional * <path> is optional and may vary to each ESP * <ver> indicates version of the API. Currently it must be “1.0” |
| **Method** | POST |
| **Content-Type** | “application/xml” |
| **Post data** | A well-formed XML, as per the specifications provided in this  document. |

1. Project Management Approach & Methodology

Scrum Framework of Agile Methodologies shall be used for the development of the E-Sign project by Subex. The Scrum Framework breaks down work from the visionary and strategic level to easily consumable, actionable tasks that all team members can work on during a cycle of timeboxed fixed duration called Sprints.

These iterations could be less than one month in length, but for the development of the system we shall be divulging a 3–4-week Sprints. Scrum emphasizes a working product at the end of the Sprint which should be completed in a sense, the working product should be fully tested, potentially shippable to the market and should be able to be integrated with other products that were released in multiple sprints. As development involves learning, innovation and surprises, the Scrum framework prioritizing a short step of development, inspecting both the resulting product and the efficacy of current practices, and then adapting the product goals and process practices allowing team to improvise and adapt in every Sprint of the project.

Subex understands that the scrum framework is deliberately developed incomplete, only defining the parts to implement Scrum Theory, Thus Various processes, methods, and techniques would be employed latter in the development phases and Sprints to adopt and accommodate according to the requirement.

The project team (if not possible, representative from both organization) should be in constant communication throughout the duration of the project either to provide requirements or to report to their organizational management. Either Project Manager or Business Analyst should be responsible for communication with both organizations concerned stakeholders on the progress and performance on each development (In each Sprint if possible).

Diagram

Description automatically generated

1. Error Codes

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.No** | **Error Code** | **Error message** | **Originator** |
| 1 | ESP-001 | Invalid e-Sign Class | ESP |
| 2 | ESP-002 | Invalid ASP ID. It cannot be Empty | ESP |
| 3 | ESP-003 | Invalid ASP ID. It may not exist or may be inactive. | ESP |
| 4 | ESP-004 | Document Hash not received | ESP |
| 5 | ESP-005 | Aadhaar cannot be Empty | ESP |
| 6 | ESP-006 | Request Time Stamp cannot be Empty | ESP |
| 7 | ESP-007 | Request Time Stamp is not valid. Please check the server time. | ESP |
| 8 | ESP-008 | Transaction ID cannot be Empty | ESP |
| 9 | ESP-009 | Duplicate Transaction ID for the given ASP. | ESP |
| 10 | ESP-010 | Input XML Signature verification failed. | ESP |
| 11 | ESP-011 | Invalid Signature on Input XML. Please use the corresponding certificate mapped with ESP. | ESP |
| 12 | ESP-012 | ESP Database Connectivity Error | ESP |
| 13 | ESP-013 | Input XML Parsing Error. | ESP |
| 14 | ESP-014 | Error Parsing CA Response XML | ESP |
| 15 | ESP-015 | KYC XML Not Parsed Properly. | ESP |
| 16 | ESP-016 | Audit Logging in DB is failed for request. | ESP |
| 17 | ESP-017 | Audit Logging in DB is failed for response. | ESP |
| 18 | ESP-018 | Audit Logging in DB is failed for Error occurred. | ESP |
| 19 | ESP-019 | KYC Request Signature Verification Failed. | ESP |
| 20 | ESP-020 | Blank Response Received from UIDAI | ESP |
| 21 | ESP-021 | KYC Response Signature Verification Failed. | ESP |
| 22 | ESP-022 | KYC Response XML Not Parsed Properly. | ESP |
| 23 | ESP-023 | Unable to connect to UIDAI server. | ESP |
| 24 | ESP-024 | Database audit logging in failed due to the duplicate transaction ID. | ESP |
| 25 | ESP-025 | OTP Request XML Not Parsed Properly. | ESP |
| 26 | ESP-026 | OTP Request Signature Verification Failed | ESP |
| 27 | ESP-027 | OTP Response XML Not Parsed Properly. | ESP |
| 28 | ESP-028 | Error during KYC Request Signature Verification. | ESP |
| 29 | ESP-029 | Error during OTP Request Signature Verification. | ESP |
| 30 | ESP-030 | Invalid ASP ID. It cannot be Empty | ESP |
| 31 | ESP-031 | Invalid ASP ID. It may not exist or may be inactive. | ESP |
| 32 | ESP-032 | Request Time Stamp cannot be Empty | ESP |
| 33 | ESP-033 | Request Time Stamp is not valid. Please check the server time. | ESP |
| 34 | ESP-034 | Transaction ID cannot be Empty | ESP |
| 35 | ESP-035 | Duplicate Transaction ID for the given ASP. | ESP |
| 36 | ESP-036 | Input XML Signature verification failed. | ESP |
| 37 | ESP-037 | Invalid Signature on Input XML. Please use the corresponding certificate mapped with ESP. | ESP |
| 38 | ESP-038 | ESP Database Connectivity Error | ESP |
| 39 | ESP-039 | Input XML Parsing Error. | ESP |
| 40 | ESP-040 | Unknown Error | ESP |
| 41 | ESP-041 | KYC XML Not Parsed Properly. | ESP |
| 42 | ESP-042 | Audit Logging in DB is failed for request. | ESP |
| 43 | ESP-043 | Audit Logging in DB is failed for response. | ESP |
| 44 | ESP-044 | Audit Logging in DB is failed for Error occurred. | ESP |
| 45 | ESP-045 | KYC Request Signature Verification Failed. | ESP |
| 46 | ESP-046 | Blank Response Received from UIDAI | ESP |
| 47 | ESP-047 | KYC Response Signature Verification Failed. | ESP |
| 48 | ESP-048 | KYC Response XML Not Parsed Properly. | ESP |
| 49 | ESP-049 | Unable to connect to UIDAI server. | ESP |
| 50 | ESP-050 | Database audit logging in failed due to the duplicate transaction ID. | ESP |
| 51 | ESP-051 | OTP Request XML Not Parsed Properly. | ESP |
| 52 | ESP-052 | OTP Request Signature Verification Failed | ESP |
| 53 | ESP-053 | OTP Response XML Not Parsed Properly. | ESP |
| 54 | ESP-054 | Error during KYC Request Signature Verification. | ESP |
| 55 | ESP-055 | Duplicate Transaction Id Error. | ESP |
| 56 | ESP-056 | Error during OTP Request Signature Verification. | ESP |