Dedicated Repository (Push) API Specification

Version 1.7

July 2016

A Digital India Initiative
National e-Governance Division.
Department of Electronics and Information Technology.
## Revision History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2</td>
<td>15/07/2015</td>
<td>New release of Digital Locker. Added CSV support for uploading URIs.</td>
</tr>
<tr>
<td>1.3</td>
<td>30/07/2015</td>
<td>Removed two attributes from Pull URI API Response - file_name, Aadhaar name.</td>
</tr>
<tr>
<td>1.4</td>
<td>08/08/2015</td>
<td>Specified the date format for CSV data.</td>
</tr>
<tr>
<td>1.5</td>
<td>01/01/2016</td>
<td>Added details of CSV Upload API. Removed Pull URI API.</td>
</tr>
<tr>
<td>1.6</td>
<td>06/05/2016</td>
<td>Added details of Push URI API.</td>
</tr>
<tr>
<td>1.7</td>
<td>20/07/2016</td>
<td>Added support to accept certificate metadata in Pull Document API.</td>
</tr>
</tbody>
</table>
Table of Contents

Revision History......................................................................................................................... 1
1. Introduction .......................................................................................................................... 3
2. Digital Locker System Overview .......................................................................................... 3
3. Key Terminology .................................................................................................................. 3
4. On-Boarding Flow ............................................................................................................... 5
5. Document Codification Scheme ......................................................................................... 5
   5.1. Unique Document URI ................................................................................................. 5
   5.2. Issuer ID (mandatory) ................................................................................................. 5
   5.3. Document Type (mandatory) ..................................................................................... 6
   5.4. Document ID (mandatory) .......................................................................................... 6
6. Document Issuance Flow ...................................................................................................... 7
7. E-Document Specifications ................................................................................................. 7
   7.1. Document URI ........................................................................................................... 7
   7.2. Document Owner ....................................................................................................... 7
   7.3. Document Format ...................................................................................................... 8
8. Issuer Interfaces ................................................................................................................... 8
   8.1. Push URI to Digital Locker ....................................................................................... 8
      8.1.1. Manual CSV Upload .......................................................................................... 9
      8.1.2. CSV Upload API .............................................................................................. 10
      8.1.3. PUSH URI API ............................................................................................... 11
   8.2. Pull Doc Request API ................................................................................................. 14
      8.2.1. Pull Doc Request Format .................................................................................. 15
      8.2.2. Pull Doc Response Format ............................................................................... 16
1. Introduction
This document provides detailed specification of Digital Locker APIs to make Aadhaar seeded documents available through Digital Locker. These APIs will be used by various issuer departments to push document meta-data to Digital Locker in bulk. For documents that are not Aadhaar seeded, please refer to Pull API Specification of Digital Locker. This document assumes that the reader is aware of Digital Locker application functionality and has read the Digital Locker Technical Specification (DLTS) available in Technical Specification section of Digital Locker Resource Center at https://digitallocker.gov.in/resource-center.php.

2. Digital Locker System Overview
The proposed architecture of the Digital Locker system is described in “Digital Locker Technical Specifications (DLTS)” document. Digital Locker system consists of e-Documents repositories and access gateways for providing an online mechanism for issuers to store and requesters to access a Digital Document in a uniform way in real-time.

3. Key Terminology
1. **Electronic Document or E-Document** – A digitally signed electronic document in XML format issued to one or more individuals (Aadhaar holders) in appropriate format compliant to DLTS specifications. Examples:
   - Degree certificate issued to a student by a university.
   - Caste certificate issued to an individual by a state government department.
• Marriage certificate issued to two individuals by a state government department.

2. **Digital Repository** – A software application complying with DLTS specifications, hosting a collection (database) of e-documents and exposing a standard API for secure real-time access.
   • While architecture does not restrict the number of repository providers, it is recommended that few highly available and resilient repositories be setup and encourage everyone to use that instead of having lots of repositories.

3. **Digital Locker** – A dedicated storage space assigned to each resident, to store authenticated documents. The digital locker would be accessible via web portal or mobile application.

4. **Issuer** – An entity/organization/department issuing e-documents to individuals in DLTS compliant format and making them electronically available within a repository of their choice.

5. **Requester** – An entity/organization/department requesting secure access to a particular e-document stored within a repository. Examples:
   • A university wanting to access 10th standard certificate for admissions
   • A government department wanting to access BPL certificate
   • Passport department wanting to access marriage certificate

6. **Access Gateway** – A software application complying with DLTS specifications providing an online mechanism for requesters to access an e-document in a uniform way from various repositories in real-time.
   • Gateway services can be offered by repository providers themselves.
   • While architecture does not restrict the number of repository providers, it is suggested that few resilient and highly available central gateway systems be setup and requesters can signup with any one of the gateways for accessing documents in the Digital repositories.

7. **Document URI** – A unique document URI mandatory for every document. This unique URI can be resolved to a full URL to access the actual document in appropriate repository.
   • Document URI is a persistent, location independent, repository independent, issuer independent representation of the ID of the document.
   • The existence of such a URI does not imply availability of the identified resource, but such URIs are required to remain globally unique and persistent, even when the resource ceases to exist or becomes unavailable.
   • While document URI itself is not a secret, access to the actual document is secure and authenticated.
4. On-Boarding Flow

- Get Issuer ID
- Create Document type
- Generate URI
  - Create REST based Pull Doc Request API
  - Create CSV File and upload via issuer login or API
  - Map URI with e-Document

5. Document Codification Scheme

5.1. Unique Document URI

Every document that is issued and made accessible via DigiLocker must have a unique way to resolve to the correct repository without conflict. This is critical to eliminate the need for all documents reference to be in one system. Federated repositories storing documents issued by various departments/agencies must be “reachable” via the gateway in a unique fashion.

All documents issued in compliance to DLTS should have the following URI format: **IssuerId-DocType-DocId** where

- **IssuerId** is a unique issuer entity ID across the country
- **DocType** is the document type optionally defined by the issuer
- **DocId** is a unique document ID within the issuer system

5.2. Issuer ID (mandatory)

All departments/agencies within government issuing citizen documents, termed as “Issuers” must have a unique identification to ensure all documents issued by them are accessible via DLTS gateway.

It is recommended that list of unique issuer codes be derived via their domain URL whenever available and be published as part of e-governance standard codification scheme with ability to add new issuers on need basis. When URL is not available for a department, a unique (alpha) code may be assigned.
Examples of issuer IDs are “maharashtra.gov.in” (Maharashtra State Government), “kseeb.kar.nic.in” (Karnataka School Board”, “cbse.nic.in” (CBSE School Board), “UDEL” (Delhi University), etc. These codes **MUST BE unique across India** and published as part of standard e-governance codification list.

### 5.3. Document Type (mandatory)

Issuers can freely define a list of document types for their internal classification. For example, CBSE may classify certificates into “MSTN” (10th mark sheet), “KVPY” (certificate issued to KVPY scholarship fellows), etc. There are no requirements for publishing these via any central registry.

Classifying documents into various types allows issuers to choose different repositories for different types. This is to future-proof the design without making assumption that all certificates issued by the issuer are available in same repository. This also allows migration from one repository to another in a gradual way. Issuers are free to define their document types without worrying any collaboration across other issuers. Keeping the length minimal allows manual entry of document URI without making it too long. Hence it is recommended to keep length to be only up to 5.

It is recommended that issuers define document types either using pure alpha case-insensitive strings of length up to 5. These document types **MUST BE unique WITHIN the issuer system**. This classification within the issuer system also allows versioning of documents making future documents to be of different formats and in different repositories without having the need to have all documents in one repository. **If need arises in future to go beyond length 5, maximum length of doc type can easily use increased without breaking compatibility any existing systems and documents.**

### 5.4. Document ID (mandatory)

A document ID determined by the department/agency (issuer) should be assigned to every document. It **MUST BE unique either within the document types of that issuer or it can be unique across all document types of that issuer.**

Document ID is an alpha-numeric string with maximum length of 10. It is recommended that issuers define document IDs either using pure alpha case-insensitive string using a RANDOM number/string generator. Document IDs **MUST BE unique WITHIN the issuer system within a document type. If need arises in future to go beyond length 10, maximum length of doc ID can easily use increased without breaking compatibility any existing systems and documents.** Using random string eliminates the possibility of “guessing” next sequence number and accessing a list of documents in a sequential way. This is critical to ensure security of documents and ensures document can be accessed ONLY IF the requester “knows” the actual document ID (instead of guessing sequential numbers).

It is highly recommended that issuer needing to issue a total of \( n \) documents within a document type use at least \( 10n \) random space from which the strings/numbers are chosen to randomly allocate. Notice that since document types allow further classification, it is suggested to keep the length **minimal.** Since issuers can easily add a new document type without any collaboration and approvals across other issuers, if more numbers are required, a new document type may be introduced.
6. **Document Issuance Flow**

Document issuance flow is given below:

1. Create a digitally signed e-document complying to DLTS specification with a URI.
   a. Issuer entity uses the unique code for itself (obtain a new one if not already listed) that is available in common DLTS Issuer Codification e-governance standards. This is a country wide “Unique Issuer ID”.
   b. Document type codification is done by the Digital Locker system administrator. Issuers may choose an available document type or if a new type of document is being issued then request Digital Locker team to create the required document type.
2. Issuer should create a document repository for storing documents and making it available online. This could be an existing database or document management system where the issued documents are stored.
3. Issue the printed document to the individual(s) for whom the document is issued to with a human readable document URI.
   a. Issuer should also offer an option to people to push the document URI to the digital lockers of the resident for whom the document was issued.

7. **E-Document Specifications**

7.1. **Document URI**

All documents issued in compliance to DLTS should have the following URI format:

`<IssuerId>-<DocType>-<DocId>`

Where,

- **IssuerId** (mandatory) - is a unique issuer entity ID. This is a unique pure alpha case-insensitive string. To easily make it unique, department's domain URL can be used whenever available. The list of issuer IDs must be published and should have a mechanism to add new ones as required. **Unique list of Issuer IDs MUST BE unique and published via central e-governance codification scheme.**

- **DocType** (mandatory) - is the document type optionally defined by the issuer. This is highly recommended for document classification and versioning purposes. Issuers may decide their own classification mechanism. This is a 5 char pure alpha string which can be expanded in future as needed.

- **DocId** (mandatory) - is a unique document ID of length up to 10 within the issuer system. It is highly recommended that this is either purely numeric or alpha to avoid confusion with “0” with “o” etc. Also, it is highly recommended to use random strings to avoid guessing the sequence of document IDs.

7.2. **Document Owner**

For avoiding document misuse, it is critical that all documents are “attached” to one or more Aadhaar holders. For example, a caste certificate may be attached to one Aadhaar holder while a marriage certificate is attached to two Aadhaar holders. Proposed DLTS solution offers a mechanism for issuers to secure access via Aadhaar authentication of any of the owners.
### 7.3. Document Format

All e-documents must be represented in PDF or XML format complying to DLTS specifications. This ensures that a standardized XML structure is used to capture common attributes of all documents.

### 8. Issuer Interfaces

The issuer organization integrating with Digital Locker maintains the documents/certificates in its own repository (database or file system) termed in Digital Locker as "Dedicated Repository". The issuer application provides APIs to Digital Locker to access the documents in this repository. Each issuer organization will have to implement 2 interfaces to integrate with the Digital Locker system. These 2 interfaces are:

1. **Push URI to Digital Locker**: This web based interface is provided to the issuers by Digital Locker system to push the URI's of all the documents available in their repositories so that the same can be displayed to the residents.
2. **Pull Doc Request API**: This REST based pull interface has to be implemented by the issuer organization to allow a resident to fetch a document from the issuer repository by providing the URI of the document.

These 2 interfaces are defined in greater details in subsequent sections.

#### 8.1. Push URI to Digital Locker

Once an issuer department issues a document for resident, the URI of that document has to be pushed into the Digital Locker system so that the URI can be displayed to the locker user. This way the user of the locker is notified of any new documents related to his/her Aadhaar number.

Digital Locker provides two mechanisms to push an URI.

- **Comma Separated Values (CSV) file for pushing URIs in bulk fashions.** This mechanism can be used to push URIs to Digital Locker in batches. This API can also be used to push the initial load of certificate details for the first time when an issuer comes onboard.
- **API to push a single URI.** This mechanism is used to push a single URI as and when a certificate or document is created. This mechanism can be used to push an URI to Digital Locker as and when a certificate is generated by integrating this API in the issuer application.

The format of the CSV file is provided in the table below:

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Aadhaar</td>
<td>Aadhaar number of the citizen to whom the document is issued.</td>
</tr>
<tr>
<td>2.</td>
<td>Uri</td>
<td>URI of the documents as stored in issuer repository. This URI will be used to fetch a document from the issuer repository.</td>
</tr>
<tr>
<td>3.</td>
<td>doc_type</td>
<td>The 5 character code defining the type of the document issued. The master list of document type codes will be published by Digital Locker.</td>
</tr>
<tr>
<td>4.</td>
<td>doc_name</td>
<td>This is descriptive name of the document.</td>
</tr>
<tr>
<td>Sr. No.</td>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>--------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>5.</td>
<td>doc_id</td>
<td>A unique number of the document. This id will be unique within the document type issued by the issuer.</td>
</tr>
<tr>
<td>6.</td>
<td>issued_on</td>
<td>The issue date of the document in DD/MM/YYYY format.</td>
</tr>
<tr>
<td>7.</td>
<td>valid_from</td>
<td>The date from which the document is valid in DD/MM/YYYY format. This may be same as the issue date. In case not applicable please put a blank value.</td>
</tr>
<tr>
<td>8.</td>
<td>valid_to</td>
<td>The expiry date of the document in DD/MM/YYYY format. In case not applicable please put a blank value.</td>
</tr>
<tr>
<td>9.</td>
<td>time_stamp</td>
<td>The time stamp when the document was generated in DD/MM/YYYY HH:MI:SS AM format. This will be used to locate a document or a row of a record in a CSV file.</td>
</tr>
<tr>
<td>10.</td>
<td>Action</td>
<td>Action that needs to be taken for the Aadhaar number and URI combination. Possible values are –</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A – Add a new URI for the given Aadhaar number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>U – Update an already added URI for the given Aadhaar Number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>D – Delete the URI for given Aadhaar number.</td>
</tr>
</tbody>
</table>

Currently, Digital Locker only supports addition of new URIs. Update and Delete actions may be supported in later releases.

If the Aadhaar number provided in each row of CSV file has already been used to open a locker, the URI provided in the same record will be stored and displayed to the user. If the Aadhaar number has not been used to open a locker then the Digital Locker system will hold this record in unopened locker state. Once the user creates a new locker the system query all the URI in unopened locker state any URI linked to the new Aadhaar number will be displayed to the user at the time of first login.

The uploaded CSV file will be picked up by the Digital Locker job scheduler and the data will imported into the locker database.

Currently, two mechanisms are provided to send or upload this file to Digital Locker. The registered issuers can manually upload the CSV file on the Digital Locker Issuer Portal using the issuer credentials provided by Digital Locker. Digital Locker also provides an API to send the CSV file to Digital Locker. Both these mechanisms are described below.

8.1.1. Manual CSV Upload
This importing of CSV files is allowed via the Digital Locker organization portal. Digital locker system will allow all the issuers registered with the Digital Locker system to login via organization portal. A set of credentials (username/password) will be provided to the issuer organization which will allow authorized users from the issuer organization to access the issuer portal. Once the user logs into the organization portal an option of creating a new task will be provided which will allow the user to upload the CSV file.
8.1.2. CSV Upload API

The REST based Push CSV Request API is provided by Digital Locker. The registered issuers can use this API to push the CSV file into Digital Locker portal. The uploaded file can be viewed by logging in into the Issuer Portal. The registered issuers can generate the API key required for this API by going in ‘API Keys’ in the ‘Settings’ section of Issuer Portal.

The following is the Request template for the Push CSV Request API.

**REQUEST URL (API END POINT)**

Production Environment-
https://partners.digitallocker.gov.in/public/issuer/api/csvupload/1/xml

Development Environment-
https://devpartners.digitallocker.gov.in/public/issuer/api/csvupload/1/xml

**METHOD** POST

**HEADER**

Name: Content-Type

Value: application/xml

**XML REQUEST STRUCTURE**

```xml
<?xml version="1.0" encoding="utf-8"?>
<PushCSVRequest xmlns:ns2="http://tempuri.org/" ver="1.0" ts="YYYY-MM-DDThh:mm:ss+-nn:nn" txn="" orgId="" keyhash="sha256(key+ts)"
    <DocDetails>
      <DocContent> </DocContent> Base64 encoded content of CSV file
      <FileName> User Defined Field for CSV File Name
    </DocDetails>
</PushCSVRequest>
```

Various elements/attributes in the request are described below-

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>XML Element</th>
<th>Mandatory (M)/Optional (O)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>ver</td>
<td>M</td>
<td>API version.</td>
</tr>
<tr>
<td>2.</td>
<td>ts</td>
<td>M</td>
<td>A timestamp value. This will be used to decode the keyHash element described below.</td>
</tr>
<tr>
<td>3.</td>
<td>txn</td>
<td>M</td>
<td>A unique transaction id provided by issuer application. This will be used to identify this Push request uniquely.</td>
</tr>
<tr>
<td>4.</td>
<td>orgId</td>
<td>M</td>
<td>Org Id is the issuer id provided to the issuer system as part of organization registration.</td>
</tr>
<tr>
<td>5.</td>
<td>keyHash</td>
<td>M</td>
<td>Provide SHA-256 encrypted value of the API key and the timestamp values concatenated together in</td>
</tr>
</tbody>
</table>
**Dedicated Repository (Push) API Specification**

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>XML Element</th>
<th>Mandatory (M)/ Optional (O)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>ver</td>
<td>M</td>
<td>API version.</td>
</tr>
<tr>
<td>2.</td>
<td>ts</td>
<td>M</td>
<td>A timestamp value as passed in the request.</td>
</tr>
<tr>
<td>3.</td>
<td>txn</td>
<td>M</td>
<td>The transaction id as passed in the request.</td>
</tr>
<tr>
<td>4.</td>
<td>orgId</td>
<td>M</td>
<td>Org Id as sent in the request.</td>
</tr>
<tr>
<td>5.</td>
<td>ResponseStatus</td>
<td>M</td>
<td>The return status code would be either 0 or 1. “0” denotes upload failure and “1” denotes upload success.</td>
</tr>
<tr>
<td>6.</td>
<td>ResponseMessage</td>
<td>M</td>
<td>This provides a descriptive success or failure message. This will provide more details in case of failures.</td>
</tr>
</tbody>
</table>

**XML RESPONSE STRUCTURE**

Digital Locker application will send the response to the CSV Upload Request in XML format. The response to the Push CSV Request will include the response status, timestamp and message. The XML response structure is as follows:

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<PushCSVResponse xmlns:ns2="http://tempuri.org/" ver="1.0" ts="YYYY-MM-DDThh:mm:ss+-/nn:nn" txn="" orgId="">
  <ResponseStatus> //1-Success //0-Failure</ResponseStatus>
  <ResponseMessage> // 1</ResponseMessage>
</PushCSVResponse>
```

Various elements/attributes in the response are described below:

- **Ver**: API version.
- **Ts**: A timestamp value as passed in the request.
- **Tn**: The transaction id as passed in the request.
- **OrgId**: Org Id as sent in the request.
- **ResponseStatus**: The return status code would be either 0 or 1. “0” denotes upload failure and “1” denotes upload success.
- **ResponseMessage**: This provides a descriptive success or failure message. This will provide more details in case of failures.

**8.1.3. PUSH URI API**

The REST based Push URI API can use to push a single URI into Digital Locker. This API can be used to push the certificate details to Digital Locker as and when a certificate is generated in the issuer system. The registered issuers can generate the API key required for this API by going in ‘API Keys’ in the ‘Settings’ section of Issuer Portal.
The following is the Request template for the Push URI API.

**REQUEST URL (API END POINT)**

Production Environment-  
https://partners.digitallocker.gov.in/public/issuer/api/issuedoc/1/xml

Development Environment-  
https://devpartners.digitallocker.gov.in/public/issuer/api/issuedoc/1/xml

**METHOD** POST

**HEADER**

- **Name**: Content-Type  
  - **Value**: application/xml

**XML REQUEST STRUCTURE**

```xml
<?xml version="1.0" encoding="utf-8"?>
<PushUriRequest xmlns:ns2="http://tempuri.org/" ver="1.0" ts="08-01-2016 10:22:54" txn="12345" orgId ="in.gov.issuer" keyhash="437b66e879183ab6a6d96a14bcf">
  <UriDetails>
    <Aadhaar>123412341234</Aadhaar>
    <Uri>in.gov.issuer-INCER-387467</Uri>
    <DocType>INCER</DocType>
    <DocName>Income Certificate</DocName>
    <DocId>387467</DocId>
    <IssuedOn>31/12/2015</IssuedOn>
    <ValidFrom>31/12/2015</ValidFrom>
    <ValidTo>31/06/2016</ValidTo>
    <Timestamp>31/12/2015 12:30:20 PM</Timestamp>
    <Action>A</Action>
  </UriDetails>
</PushUriRequest>
```

Various elements/attributes in the request are described below-

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>XML Element</th>
<th>Mandatory (M)/ Optional (O)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>ver</td>
<td>M</td>
<td>API version.</td>
</tr>
<tr>
<td>2.</td>
<td>ts</td>
<td>M</td>
<td>A timestamp value. This will be used to decode the keyHash element described below.</td>
</tr>
<tr>
<td>3.</td>
<td>txn</td>
<td>M</td>
<td>A unique transaction id provided by issuer application. This will be used to identify this Push request uniquely.</td>
</tr>
<tr>
<td>4.</td>
<td>orgId</td>
<td>M</td>
<td>Org Id is the issuer id provided to the issuer system as part of</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>keyHash</td>
<td>M</td>
<td>Provide SHA-256 encrypted value of the API key and the timestamp values concatenated together in this sequence. You must use the same timestamp value that you have specified in the ts element described above. The API key can be generated from Digital Locker Issuer portal.</td>
</tr>
<tr>
<td>6.</td>
<td>Aadhaar</td>
<td>M</td>
<td>Aadhaar number of the citizen to whom the document is issued.</td>
</tr>
<tr>
<td>7.</td>
<td>Uri</td>
<td>M</td>
<td>URI of the documents as stored in issuer repository. This URI will be used to fetch a document from the issuer repository.</td>
</tr>
<tr>
<td>8.</td>
<td>DocType</td>
<td>M</td>
<td>The 5 character code defining the type of the document issued. The master list of document type codes will be published by Digital Locker.</td>
</tr>
<tr>
<td>9.</td>
<td>DocName</td>
<td>M</td>
<td>This is descriptive name of the document.</td>
</tr>
<tr>
<td>10.</td>
<td>DocId</td>
<td>M</td>
<td>A unique number of the document. This id will be unique within the document type issued by the issuer.</td>
</tr>
<tr>
<td>11.</td>
<td>IssuedOn</td>
<td>M</td>
<td>The issue date of the document in DD/MM/YYYY format.</td>
</tr>
<tr>
<td>12.</td>
<td>ValidFrom</td>
<td>M</td>
<td>The date from which the document is valid in DD/MM/YYYY format. This may be same as the issue date. In case not applicable please put a blank value.</td>
</tr>
<tr>
<td>13.</td>
<td>ValidTo</td>
<td>M</td>
<td>The expiry date of the document in DD/MM/YYYY format. In case not applicable please put a blank value.</td>
</tr>
<tr>
<td>14.</td>
<td>Timestamp</td>
<td>M</td>
<td>The timestamp when the document was generated in DD/MM/YYYY HH:MI:SS AM format. This will be used to locate a document or a row of a record in a CSV file.</td>
</tr>
</tbody>
</table>
| 15. | Action | M | Action that needs to be taken for the Aadhaar number and URI combination. Possible values are – A – Add a new URI for the given
Aadhaar number.

U – Update an already added URI for the given Aadhaar Number.
D – Delete the URI for given Aadhaar number.

Currently, Digital Locker only supports addition of new URIs. Update and Delete actions may be supported in later releases.

XML RESPONSE STRUCTURE

The response to the Push URI Request will include the response status, timestamp and message. The XML response structure is as follows:

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<PushUriResponse xmlns:ns2="http://tempuri.org/" ver="1.0" ts="YYYY-MM-DDThh:mm:ss+/nn:nn" txn="12345" orgId ="in.gov.issuer">
  <ResponseStatus> 1</ResponseStatus> //1-Success //0-Failure
  <ResponseMessage> </ResponseMessage>
</PushUriResponse>
```

Various elements/attributes in the response are described below-

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>XML Element</th>
<th>Mandatory (M)/Optional (O)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>ver</td>
<td>M</td>
<td>API version.</td>
</tr>
<tr>
<td>2.</td>
<td>ts</td>
<td>M</td>
<td>A timestamp value as passed in the request.</td>
</tr>
<tr>
<td>3.</td>
<td>txn</td>
<td>M</td>
<td>The transaction id as passed in the request.</td>
</tr>
<tr>
<td>4.</td>
<td>orgId</td>
<td>M</td>
<td>Org Id as sent in the request.</td>
</tr>
<tr>
<td>5.</td>
<td>ResponseStatus</td>
<td>M</td>
<td>The return status code would be either 0 or 1. &quot;0&quot; denotes upload failure and “1” denotes upload success.</td>
</tr>
<tr>
<td>6.</td>
<td>ResponseMessage</td>
<td>M</td>
<td>This provides a descriptive success or failure message. This will provide more details in case of failures.</td>
</tr>
</tbody>
</table>

8.2. Pull Doc Request API

The REST based Pull Doc Request API has to be implemented by the issuers and will be consumed by Digital Locker system. This API will be invoked when the resident clicks on the URI displayed in the Govt. Issued documents section of the Digital locker portal. The issuer system will respond to this API by sending the certificate data. The certificate data should be sent in one of the two formats depending on the request send by Digital Locker:

a. PDF document format
b. XML format for machine readable metadata

8.2.1. Pull Doc Request Format

The following is the XML request template for the PULL Doc Request API.

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<PullDocRequest xmlns:ns2="http://tempuri.org/" ver="1.0" ts="YYYY-MM-DDThh:mm:ss+-/nn:nn" txn="" orgId="" keyhash="sha256(key+ts)"
metadata="">
  <DocDetails>
    <URI>testt.in.gov.kerala.edistrict-A00116301471-420</URI>
  </DocDetails>
</PullDocRequest>
```

Various elements/attributes in the request are described below-

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>XML Element</th>
<th>Mandatory (M)/Optional (O)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>ver</td>
<td>M</td>
<td>API version.</td>
</tr>
<tr>
<td>2.</td>
<td>ts</td>
<td>M</td>
<td>A timestamp value. This will be used to decode the keyHash element described below.</td>
</tr>
<tr>
<td>3.</td>
<td>txn</td>
<td>M</td>
<td>A unique transaction id provided by DigiLocker application. Issuers can use this id to track this Pull Document Request.</td>
</tr>
<tr>
<td>4.</td>
<td>orgId</td>
<td>M</td>
<td>Org Id is the user id provided to the Digital Locker application by the issuer application for accessing the API.</td>
</tr>
<tr>
<td>5.</td>
<td>keyHash</td>
<td>M</td>
<td>Provide SHA-256 encrypted value of the API key and the timestamp values concatenated together in this sequence. The issuer application may provide an API key to Digital Locker application.</td>
</tr>
<tr>
<td>6.</td>
<td>metadata</td>
<td>O</td>
<td>Possible values of this attribute are &quot;Y&quot; or &quot;N&quot;. If the value of this attribute is &quot;Y&quot;, then the API must return the certificate metadata in XML format in the response. If the value of this attribute is &quot;N&quot; or if the metadata attribute is not present in the request, then the API must return Base64 encoded PDF data in the response. Please see the response section below for more details.</td>
</tr>
<tr>
<td>7.</td>
<td>URI</td>
<td>M</td>
<td>URI identifies the document</td>
</tr>
</tbody>
</table>
8.2.2. Pull Doc Response Format

The response to the PULL Doc request will include the Doc content of any documents linked to the given URI in the request. The issuer will provide the response back to the Digital Locker system synchronously. The PDF data should be sent in DocContent element and XML metadata should be sent in MetadataContent element. The response should contain the certificate data in only one of these formats based on the metadata attribute in the request.

The following is the XML response template for the PULL Doc Response API.

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<PullDocResponse xmlns:ns2="http://tempuri.org/"
  <ResponseStatus Status="1" ts="YYYY-MM-DDThh:mm:ss+-nn:nn"
    txn=""> //1-Success //0-Failure
  </ResponseStatus>
  <DocDetails>
    //Send one of DocContent or MetadataContent element based on the metadata attribute in the request.
    <DocContent>
      //Bytes encoded with Base64 in string format
    </DocContent>
    <MetadataContent>
      //Certificate metadata in XML format
    </MetadataContent>
  </DocDetails>
</PullDocResponse>
```

Various elements/attributes in the response are described below-

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>XML Element</th>
<th>Mandatory (M)/Optional (O)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>ts</td>
<td>M</td>
<td>A timestamp value as sent in the request.</td>
</tr>
<tr>
<td>2.</td>
<td>txn</td>
<td>M</td>
<td>Transaction id as passed in the request.</td>
</tr>
<tr>
<td>3.</td>
<td>Status</td>
<td>M</td>
<td>1 for success, 0 for error.</td>
</tr>
<tr>
<td>4.</td>
<td>DocDetails</td>
<td>M</td>
<td>Issuer can add meta content specific to document here.</td>
</tr>
<tr>
<td>5.</td>
<td>DocContent</td>
<td>O</td>
<td>Enclose the Base64 byte encoded contents of PDF file in this element. The DocContent element should be sent only if the metadata attribute in the original request is sent as “N” or is absent.</td>
</tr>
<tr>
<td>6.</td>
<td>MetadataContent</td>
<td>O</td>
<td>Enclose the certificate metadata</td>
</tr>
</tbody>
</table>
in XML format. The MetadataContent element should be sent only if the original request contains `metadata` attribute as “Y”.

Please note that the DocContent and MetadataContent elements are mutually exclusive. The certificate data should be sent in only one of these elements based on the `metadata` attribute in the request.