FRS Registration Processor

[**Edit**](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor/_edit)[**New Page**](https://github.com/mosip/mosip/wiki/_new)

Rahuul Chettri edited this page 21 hours ago · [38 revisions](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor/_history)

**Table Of Content**

* [Registration Processor](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#registration-processor)
* [1. ID Lifecycle Management](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#1-id-lifecycle-management)
  + [1.1 New ID Issuance](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#11-new-id-issuance) *(RPR\_FR\_1.1)*
  + [1.2 Update Individual’s Information](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#12-update-individuals-information) *(RPR\_FR\_1.2)*
  + [1.3 De-activate Individual’s ID](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#13-de-activate-individuals-id) *(RPR\_FR\_1.3)*
  + [1.4 Re-activate Individual’s ID](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#14-re-activate-individuals-id) *(RPR\_FR\_1.4)*
* [2. Configurable Workflow](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#2-configurable-workflow)
  + [2.1 Orchestration](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#21-orchestration) *(RPR\_FR\_2.1)*
  + [2.2 Retry Processing (In case of exceptions/failures)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#22-retry-processing-in-case-of-exceptionsfailures) *(RPR\_FR\_2.2)*
  + [2.3 Resume Workflow](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#23-resume-workflow) *(RPR\_FR\_2.3)*
  + [2.4 Integration (System capability) & Workflow Customization (Ability to plug-in/exclude stages)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#24-integration-system-capability--workflow-customization-ability-to-plug-inexclude-stages) *(RPR\_FR\_2.4)*
  + [2.5 Multiple Workflows (Specific to lifecycle – E.G.: New vs. Update, Activation vs. Deactivation, Applicant Type specific workflow)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#25-multiple-workflows-specific-to-lifecycle--eg-new-vs-update-activation-vs-deactivation-applicant-type-specific-workflow) *(RPR\_FR\_2.6)*
  + [2.6 Scalability and Throughput](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#26-scalability-and-throughput) *(RPR\_FR\_2.7)*
* [3. Types of Stages](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3-types-of-stages)
  + [3.1 Pre-processing Validations](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#31-pre-processing-validations)
    - [3.1.1 Sanity Check](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#311-sanity-check) *(RPR\_FR\_3.1)*
    - [3.1.2 Virus Scan](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#312-virus-scan) *(RPR\_FR\_3.2)*
    - [3.1.3 Source Authentication](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#313-source-authentication) *(RPR\_FR\_3.3)*
    - [3.1.4 Machine-User-Center-Device Checks](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#314-machine-user-center-mapping-check) *(RPR\_FR\_3.4)*
    - [3.1.5 GPS Capture Check](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#315-gps-capture-check) *(RPR\_FR\_3.5)*
    - [3.1.6 Operator & Supervisor Validation](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#316-operator--supervisor-validation) *(RPR\_FR\_3.6)*
  + [3.2 Processing](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#32-processing)
    - [3.2.1 Individual Data Validations](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#321-individual-data-validations)
      * [3.2.1.1 Data Quality Check: Photo, Age, Gender Data Check](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3211-data-quality-check-photo-age-gender-data-check) *(RPR\_FR\_3.7)*
      * [3.2.1.2 Biometrics Quality Check](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3212-biometrics-quality-check) *(RPR\_FR\_3.8)*
      * [3.2.1.3 Doc. Validation - OCR](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3213-doc-validation---ocr) *(RPR\_FR\_3.9)*
    - [3.2.2 Functional Validations](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#322-functional-validations)
      * [3.2.2.1 File & Document Validation](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3221-file--document-validation) *(RPR\_FR\_3.10)*
      * [3.2.2.2 Introducer Validation](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3222-introducer-validation) *(RPR\_FR\_3.11)*
      * [3.2.2.3 Deduplication – Demographic, Biometrics](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3223-deduplication--demographic-biometrics) *(RPR\_FR\_3.12)*
    - [3.2.3 External System Integration: (Elaborate with examples)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#323-external-system-integration-elaborate-with-examples)
      * [3.2.3.1 Data Verification (Pluggable by SI – Not part of MOSIP)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3231-data-verification-pluggable-by-si--not-part-of-mosip) *(RPR\_FR\_3.13)*
      * [3.2.3.2 Data Enrichment (Incl. receipt of Update Packet from ext. system and process thereafter, in terms of MOSIP’s capability)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3232-data-enrichment-incl-receipt-of-update-packet-from-ext-system-and-process-thereafter-in-terms-of-mosips-capability) *(RPR\_FR\_3.14)*
      * [3.2.3.3 Manual Verification for ext. system data update (Pluggable by SI)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3233-manual-verification-for-ext-system-data-update-pluggable-by-si)*(RPR\_FR\_3.15)*
      * [3.2.3.4 Manual Adjudication (Pluggable by SI)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3234-manual-adjudication-pluggable-by-si) *(RPR\_FR\_3.16)*
      * [3.2.3.5 ABIS Integration (Incl. ABIS Middleware)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3235-abis-integration-incl-abis-middleware) *(RPR\_FR\_3.17)*
    - [3.2.4 ID Issuance](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#324-id-issuance)
      * [3.2.4.1 Identity Generation (Refer to UIN Generation service) – Incl. UIN Generation and UIN association](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3241-identity-generation-refer-to-uin-generation-service--incl-uin-generation-and-uin-association) *(RPR\_FR\_3.18)*
      * [3.2.4.2 Store/Update ID Repository (Refer to ID-Auth)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3242-storeupdate-id-repository-refer-to-id-auth) *(RPR\_FR\_3.19)*
      * [3.2.4.3 Data Extractor for ID Authentication](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#3243-data-extractor-for-id-authentication)
    - [3.2.5 Capture Audit Trails/Analytics Data](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#325-capture-audit-trailsanalytics-data) *(RPR\_FR\_3.20)*
  + [3.3 Post-Processing](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#33-post-processing)
    - [3.3.1 Notification (Pluggable by SI)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#331-notification-pluggable-by-si) *(RPR\_FR\_3.21)*
    - [3.3.2 Print & Post (Pluggable by SI)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#332-print--post-pluggable-by-si) *(RPR\_FR\_3.22)*
    - [3.3.3 Data Seeding to External Functional ID System (Pluggable by SI)](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#333-data-seeding-to-external-functional-id-system-pluggable-by-si) *(RPR\_FR\_3.23)*

**Registration Processor**

**1. ID Lifecycle Management**

When an individual goes to the registration center, the registration officer or supervisor captures the Demographic and Biometric details of the Individual. Then Registration Client packages the information in a secure way (encrypted packets) and sends it to the Registration Processor. Registration Processor now processes the data of the Individual for quality and uniqueness and then issues new ID or updates the individual’s details. The packages received from the Registration Client must pass the sanity checks and validations to carry out the following processes:

* New ID Issuance
* Update individual’s information
* De-activate individual’s ID
* Re-activate individual’s ID

**1.1 New ID Issuance**

After the packets for new ID issuance are received from the Registration Client and has passed the sanity checks and validations, the system performs the demographic deduplication (using name, date of birth, and gender) and biometric deduplication (using [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface)) and then issues a new ID to the individual. After issuance of the ID, the system notifies the individual via the configured mode of notification (e-mail or SMS) and sends the ID card to the printing & postal service provider.

**1.2 Update Individual’s Information**

After the packets for update are received from the Registration Client or Residential Portal and have passed the sanity checks and validations, the system performs the demographic deduplication (using name, date of birth, and gender) and biometric deduplication (using [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface)) and then updates the individual’s information via two different ways:

1. Packets that are received through Registration Client, updates the individual’s Biometric and demographic details.
2. Packets that are received through Residential Portal, updates the individual’s address and contact information.

After the individual’s information is updated, the system notifies the individual via the configured mode of notification (e-mail or SMS) and sends the ID card to the printing & postal service provider.

**1.3 De-activate individual’s ID**

When the country chooses to de-activate individual’s ID due to any specific reason, the packets for UIN de-activate will go through the sanity checks and validations. Then the system checks if the status of the UIN is in activated state or not. If in activated state, the system de-activates the individual’s ID.

**1.4 Re-activate individual’s ID**

When the country chooses to re-activate individual’s ID due to any specific reason, the packets for UIN re-activate will go through the sanity checks and validations. Then the system checks if the status of the UIN is in de-activated state or not. If in de-activated state, the system re-activates the individual’s ID.

**2. Configurable Workflow**

**2.1 Orchestration**

MOSIP provides the flexibility to sequence micro services to achieve certain functionality. This feature enables System Integrator to plug in micro services as per a country requirement.

Link to [**design**](https://github.com/mosip/mosip/blob/master/docs/design/registration-processor/Approach_for_external_system_integration.md)

Link to [**design**](https://github.com/mosip/mosip/blob/master/docs/design/registration-processor/Approach_for_http_integration.md)

**2.2 Retry Processing (In case of exceptions/failures)**

If the system is unable to process a request due to infrastructure failure or stuck at a particular stage, then the system will retry processing the packet for a certain number of times (the threshold limit is configurable).

**2.3 Resume Workflow**

If the packets processing is incomplete and is stuck at a particular stage due to some system exception, such as technical, components, and infrastructure failure (database, internal service, queue, etc.) then the system identifies the blocked packets, which are in processing stage and re-sends the packets to the particular stage where it has stopped.

**2.4 Integration (System capability) & Workflow Customization (Ability to plug-in/exclude stages)**

System Integrator can integrate their system with MOSIP.

Link to [**design**](https://github.com/mosip/mosip/blob/master/docs/design/registration-processor/Approach_for_external_system_integration.md)

Link to [**design**](https://github.com/mosip/mosip/blob/master/docs/design/registration-processor/Approach_for_http_integration.md)

**2.5 Multiple Workflows (Specific to lifecycle – E.G.: New vs. Update, Activation vs. Deactivation, Applicant Type specific workflow)**

Multiple workflows increase reusability, readability and maintainability of different micro services.

**2.6 Scalability and Throughput**

MOSIP is scalable so that it can handle any kind of processing load or request in the future without disturbing the base architecture. MOSIP infrastructure can handle additional processing request based on the requirement of a country. The architecture is design in such a way that it is flexible to support scalability and holds the request until the end goal is achieved.

Link to [**design**](https://github.com/mosip/mosip/blob/0.9.0_MOS-15017/docs/design/registration-processor/Approach_for_Back_Pressure_Handling.md)

**3. Types of Stages**

**3.1 Pre-processing Validations**

**3.1.1 Sanity Check**

After the packets received from the Registration Client, the system performs the sanity check as follows:

1. **Authentication** - Authenticates the packet whether it is received from the verified source.
2. **Virus Scan** - Performs a virus scan of that received packet and moves the packet to the DMZ file system. Refer below for the process:
   * Sends the byte array of the encrypted packet to the virus scanner.
   * If the virus scanner finds a virus, then the system rejects the packet.
   * Sends the byte array of the encrypted packet received to decrypt the Packet.
   * If decryption fails, then the system rejects the packet.
   * Sends the byte array of the decrypted Packet received to the virus scanner.
   * If the virus scanner finds a virus, then the system rejects the packet.
   * If the virus scanner does not find a virus, then the system moves the packet to DMZ file system.
3. **Packet Integrity Check** - Calculates a hash sequence of the packet and compares with that of the hash sequence received from the registration client, to verify that the packet was not tempered during a transit. Refer below for the process:
   * Fetches the hash sequence for the registration id from registration-sync list table.
   * If registration id is not available then responds with an error code.
   * Calculates the hash sequence using the check sum utility.
   * Compares the hash sequences and if the hash sequences do not match, then responds with an error code. If the hash sequence matches then the system proceeds further to another step.
4. **Packet Size Check** - Calculates the size of the packet received and compares with that of the packet size received from the registration client, to verify that the packet was not tempered during a transit.
   * Checks if the packet size is more than the configured size set in the configuration server. If the packet size is more than the configured size, then sends an error response to Registration Client.
5. **Packet Format Check** - Validates if the packet format is in the configured format i.e. “.zip” (configured by the admin).
   * Checks if the packet format is not as the configured format i.e. “.zip” set in the configuration server. If the packet format is not like the configured format, then sends an error response to Registration Client.
6. **Duplicate Check** – Validates if the packet is already available in the system
   * Checks if the packet is a duplicate request by validating the RID that is available in the Registration Table and the packet is not in RESEND status. If the Packet Request is a Duplicate Request, then send an error response to Registration Client.

**3.1.2 Virus Scan**

The system performs the virus scan in two different stages, which are listed below:

1. When the packet is received from a Registration Client, the system performs virus scan of that packet and moves to the DMZ file system.
   * Refer to [**virus scan**](https://github.com/mosip/mosip/wiki/FRS-Registration-Processor#311-sanity-check) of Sanity Check.
2. When the packet is picked from the DMZ file system, the system performs virus scan of that packet and store the packet in packet store. Refer below for the process:
   * Fetches the packet from DMZ file system.
   * Sends the byte array of the encrypted packet to the virus scanner.
   * If the virus scanner finds a virus, then the system rejects the request.
   * Sends the byte array of the encrypted packet received to decrypt the packet.
   * If Decryption fails, then the system rejects the packet.
   * Sends the byte array of the decrypted Packet received to the virus scanner.
   * If the virus scanner finds a virus, then the system rejects the packet.
   * If the virus scanner does not find a virus, then the system updates the transaction and status.

**3.1.3 Source Authentication**

**3.1.4 Machine-User-Center Mapping Check**

System validates the registration machine, registration officer, and registration center details to ensure that the packet is received from the verified source. Also, validates those devices that are used for packet creation, to ensure that the devices are verified. Refer below for the process:

1. Fetches the registration machine ID, officer ID, center ID and GPS from the database. Then validates if machine ID, center ID, GPS, and officer ID are sync in the master databases as follows:
   * Checks if the center ID was active in center master table when a packet was created.
   * Checks if the machine ID was active in machine master table when a packet was created.
   * Checks if at least one from a supervisor/officer ID is present in packet meta info. Checks if officers/supervisor was active in user master table when a packet was created.
   * Checks if the mapping of center-machine-officer was present in center-machine-user mapping table.
   * Checks if GPS was captured or not.
2. Validates if the devices were mapped with the center where it was created.
3. Validate if the devices were active when a packet was created.
4. For all the failure scenarios, marks registration status with the corresponding error message.

**3.1.5 GPS Capture Check**

**3.1.6 Operator & Supervisor Validation**

When the packet is received from the Registration Client, the system first checks the Operator & Supervisor ID using configuration manager. Then validates an operator & a supervisor using biometric authentication such as face image, iris image, and finger print image, OTP based authentication, and password-based authentication to make sure that the Operator & Supervisor is authenticated. Refer below for the process:

**A. Authenticate and Validates using Officer and Supervisor ID**

System fetches an officer details and/or a supervisor details from the master list using officer and/or supervisor ID and checks if the officer and/or supervisor details are available. Then the system performs the following steps:

* If the officer and/or supervisor details are not available, then sends packet for manual adjudication.
* If the officer and/or supervisor details are available, then checks if an officer and/or supervisor is active or not.
* If an officer and/or a supervisor is not active, then sends packet for manual adjudication.
* If an officer and/or a supervisor is active, checks the type of authentication required for the officer and/or the supervisor.

**B. Authenticate using Biometric**

The system sends the UIN, face image, iris image, and finger print image of the operator and supervisor to ID Authentication and receives a response as "TRUE" or "FALSE" to authenticate the operator and supervisor. If the biometric is valid, then responds is received as "TRUE", which indicates that officer or supervisor is authenticated.

**C. Authenticate using OTP and Password Validation**

The system validates an officer with a password/OTP authentication if the officer or supervisor ID is available but all biometrics (iris, fingerprint, face, pin) of the officer or the supervisor are null. In that case, the system checks if any of the officer or supervisor bio-metrics are available and officer or supervisor password/OTP is valid then responds is received as "TRUE", which indicates that officer or supervisor is authenticated.

**3.2 Processing**

**3.2.1 Individual Data Validations**

**3.2.1.1 Data Quality Check: Photo, Age, Gender Data Check**

The system checks if the photo, age and gender captured by the registration officer while registering an individual using registration client are in sync.

**3.2.1.2 Biometrics Quality Check**

The system checks the quality of biometrics (Face Photo, Finger Print Image, and Iris Image) captured by the registration officer while registering an individual using registration client.

The system also checks the liveliness of the Finger Print Image captured of an individual and validates the photo captured as per ICAO (International Civil Aviation Organization) standards.

**3.2.1.3 Doc. Validation - OCR**

The system checks if some data captured while registering an individual using registration client is available in the document, which is uploaded using OCR (Optical Character Recognition).

**3.2.2 Functional Validations**

**3.2.2.1 File & Document Validation**

When the files that are received from the Registration Client, the system first check the file’s availability in the registration packet. If available, then verifies the documents required for an individual based on the type of registration. Refer below for the process:

**A. Validates the File**

The system performs file validation by checking files availability in a packet with the file names available in the MetaInfo file as follows:

1. Extracts the file names from the meta info file in the packet.
2. Compares the file names with files extracted in DFS by searching for the files
3. In case of successful comparison, updates the Registration table with "File Validation Successful".
4. In case of unsuccessful comparison, updates the Registration table with "File Validation Failed" and marks for retry.

**B. Verify the Documents**

1. The system identifies a document based on application type (new registration) and applicant type (child or adult) from Packet Meta Info
2. If the applicant is a child and application is new registration, the system checks a Proof of Residence (POR) document availability.
3. If the applicant is an adult and application is new registration, the system checks a Proof of Identity (POI) and Proof of Address (POA) documents availability.
4. If Proof of Birth (POB) is verified for both the applicant (adult or child) and application is new registration, then system checks a POB documents availability.

**3.2.2.2 Introducer Validation**

**3.2.2.3 Deduplication – Demographic, Biometrics**

When the packet from the Registration Client has gone through the sanity checks, the system performs the demographic deduplication and then biometric deduplication. Deduplication is the process to find a duplicate by comparing the individual’s details (biometric and demographic data) with the data stored in the system for which UIN has been generated or yet to be generated.

**A. Demographic deduplication**

1. Compares the demographic details (name, gender, and date of birth) of the individual with the data available in the system to find a potential match.
2. If a potential match is found, then sends the packet to [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface) to perform 1:1 biometric match (it is an additional check to confirm that it is a duplicate).
3. If a potential match is not found in [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface) or Demographic deduplication, then sends the packet to perform biometric deduplication.

**B. Biometric deduplication**

1. Performs Biometric deduplication (1:N, where N indicates the whole set of biometric available in the system) by sending the data to [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface).
2. [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface) compares the biometric data received with the whole set of the data to find a potential matches based on a configured threshold.
3. If a potential match is found in [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface), then the system sends the packet for manual verification.
4. If a manual verifier (experts who know more about biometrics) finds a duplicate, then rejects the packet.
5. If the manual verifier or [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface) does not find the duplicate, then moves the packet for UIN generation.

**3.2.3 External System Integration: (Elaborate with examples)**

**3.2.3.1 Data Verification (Pluggable by SI – Not part of MOSIP)**

The System Integrator can plug-in a stage in the workflow where the stage can communicate with any other external system and receives some data. Therefore, the system can verify the data captured during a registration with the data received from the external system.

**3.2.3.2 Data Enrichment (Incl. receipt of Update Packet from ext. system and process thereafter, in terms of MOSIP’s capability)**

MOSIP receives some data from the external system in a form of Packet (as per MOSIP Standards). The MOSIP System has the capability to receive this updated packet and process it with the packet received from Registration Client.

**3.2.3.3 Manual Verification for ext. system data update (Pluggable by SI)**

When the system verifies the data received with the data captured and finds an issue while comparing the data or a country wants to update the data after manually verifying the data. Then the System Integrator builds a Manual Verification Module for External System data mismatch.

**3.2.3.4 Manual Adjudication (Pluggable by SI)**

When Biometric Deduplicates are found in [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface), the System Integrator can plug-in the Manual Adjudication Stage, which would send the biometric and demographic data of the duplicates to a Manual Adjudicator. The Manual Adjudicator now can perform various validations on the duplicate data and inform the MOSIP system if the two records are duplicates or not.

**3.2.3.5 ABIS Integration (Incl. ABIS Middleware)**

The MOSIP System, in-order to perform Biometric Deduplication (validate if there are no biometric duplicates in system), integrates with one or multiple ABISs ([**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface)).

ABIS Middleware, which is designed by MOSIP and MOSIP Middleware, designed by [**Automated Biometric Identification System**](https://github.com/mosip/mosip/wiki/Automated-Biometric-Identification-System-(ABIS)-Interface) is used to communicate between MOSIP system and ABIS.

**3.2.4 ID Issuance**

**3.2.4.1 Identity Generation (Refer to UIN Generation service) – Incl. UIN Generation and UIN association**

After all the business validation are completed, the system gets a Unique Identification Number (UIN) from the kernel [**UIN Generation**](https://github.com/mosip/mosip/wiki/UIN-Generation) and allocates the UIN by sending the new UIN number and the packet data to [**ID repository**](https://github.com/mosip/mosip/wiki/ID-Repository-API).

**3.2.4.2 Store/Update ID Repository (Refer to ID-Auth)**

The Registration Processor stores or updates ID Repository during registration process and [**ID Authentication**](https://github.com/mosip/mosip/wiki/FRS-Authentication-Services) retrieves identity of an individual for their authentication. For more details about ID Repository, click the [**Wiki**](https://github.com/mosip/mosip/wiki/ID-Repository-API).

**3.2.4.3 Data Extractor for ID Authentication**

A stage that extracts the latest copy of an individual’s data after the Individual has registered in MOSIP or has updated their data in MOSIP and sends it to ID Authentication. Now, ID Authentication can use the latest copy of the Individual’s data for Authentication.

**3.2.5 Capture Audit Trails/Analytics Data**

When any transaction is performed in MOSIP system or the packet fails any validations, or any system level exception happens, then the same is captured as part of MOSIP Audit Trails.

**3.3 Post-Processing**

**3.3.1 Notification (Pluggable by SI)**

Notification (SMS/Email as configured) is the final step of all the life cycle processes, which is received for an individual. System sends a notification to the individual for various life cycle scenarios such as, packet failure, UIN issuance, update of UIN data, activate or deactivate UIN, finding a lost UIN, etc. using kernel [**Template Merger**](https://github.com/mosip/mosip/wiki/FRS-Common-Services#45-template-merger-) and [**Notification Manager**](https://github.com/mosip/mosip/wiki/FRS-Common-Services#4-notification-).

**3.3.2 Print & Post (Pluggable by SI)**

After a UIN is generated or UIN data is updated, the system creates a UIN card using kernel [**Template Merger**](https://github.com/mosip/mosip/wiki/FRS-Common-Services#45-template-merger-) and sends it to Printing and Postal Service Provider.

**3.3.3 Data Seeding to External Functional ID System (Pluggable by SI)**

If the country wants to Integrate with any External Functional ID Systems, MOSIP provides the capability to integrate with any External System.