ABDM-Wrapper

Updated: **29-11-2024**

Table of Contents

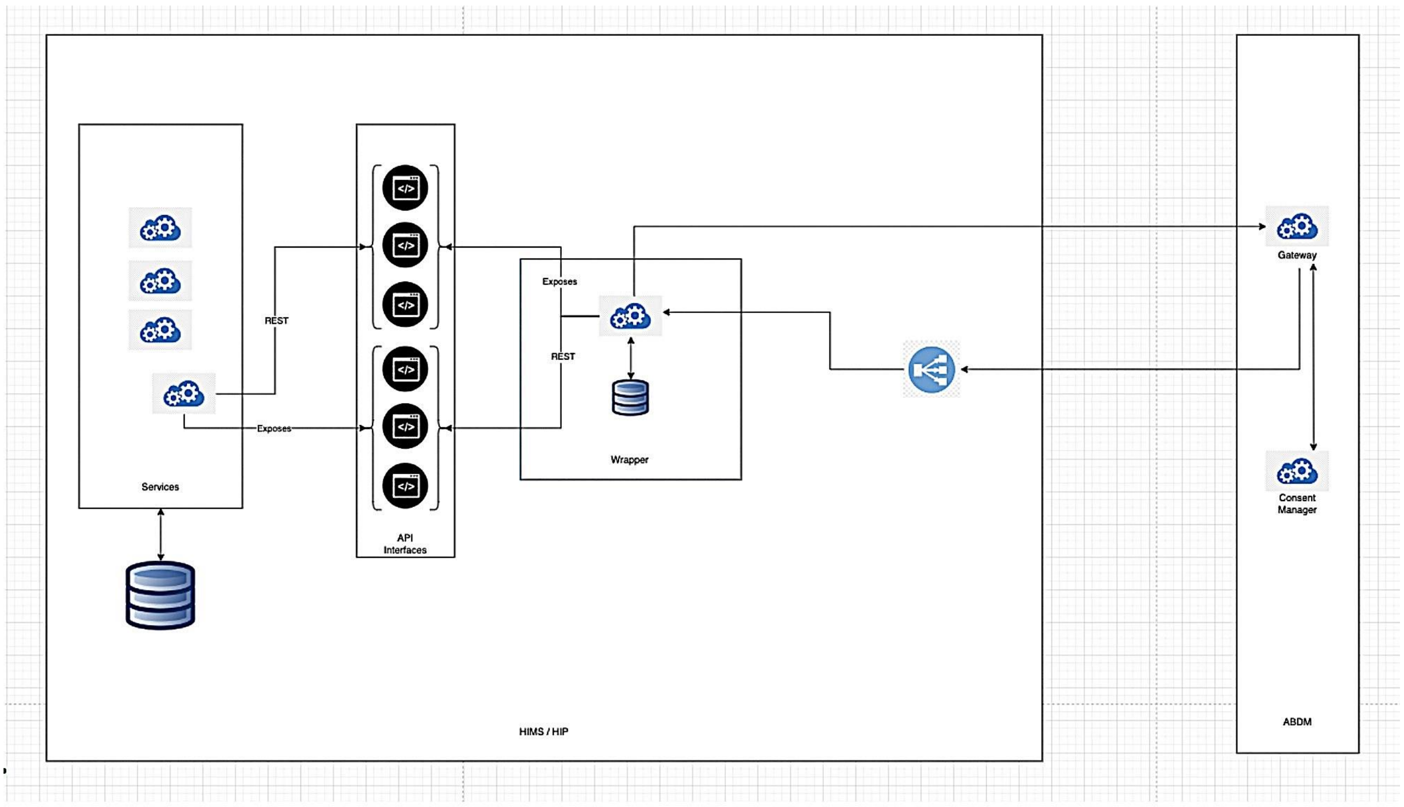
ABDM-Wrapper

The Ayushman Bharat Digital Mission (ABDM) is a government initiative that aims to develop a digital health infrastructure for India. An overview of ABDM can be found here. The ABDM aims to improve the efficiency and transparency of healthcare data transfer between patients, medical institutions, and healthcare service providers. It also allows patients to securely store their medical information and share with others as needed. The National Health Authority (NHA) is implementing Ayushman Bharat Digital Mission (ABDM) to create a digital health ecosystem for the country. ABDM intends to support different healthcare facilities like clinics, diagnostic centers, hospitals, laboratories and pharmacies in adopting the ABDM ecosystem to make available the benefits of digital health for all the citizens of India. In order to make any digital solution ABDM compliant, it has to go through 3 milestones and obtain AND certification.

- Milestone 1: ABHA Id creation, verification
- Milestone 2: Linking and exporting health data
- Milestone 3: Sending a consent request and importing data from other applications in the ecosystem

ABDM Wrapper is created to solve the challenges and issues faced by integrators to bring their systems into ABDM ecosystem. Wrapper aims to abstract complex workflows and algorithms exposing clean and simple interfaces for integrators. Wrapper abstracts implementation of HIP and HIU workflows involved in **Milestone 2** and **Milestone 3**.

1. Architecture



The wrapper follows the Facade architecture where the facility and wrapper both exposes some rest APIs and they communicate using those.

2. Introduction to ABDM

The whole ABDM architecture revolves under care contexts which is nothing but a record for a particular patient.
A care context has referenceNumber (record-id) and its display along with what type of HI-Type it is.
There are 8 Health Information Types:

- DiagnosticReport
- DischargeSummary
- HealthDocumentRecord
- ImmunizationRecord
- OPConsultation
- Prescription
- WellnessRecord
- Invoice

Every care context should be unique for an Abha Address, a single care context can have multiple HI-Types.

3. About Wrapper

- Wrapper is a spring boot application packaged with MongoDB database. Wrapper can be deployed on existing HMIS's / health facility's infrastructure.
- There are sets of interfaces which wrapper exposes and the existing services need to invoke them to implement ABDM workflows.
- At the same time if HMIS is an HIP, then existing services should expose a set of interfaces which wrapper needs to invoke to get information from health providers.
- The callback APIs which gateway would be making to wrapper should be behind facility's firewall.

4. Version of APIs in Wrapper

V1: v1 version represents the v0.5, v1, v2 versions of the ABDM APIs.

V3: V3 version of the APIs are the latest APIs provided by ABDM.

NOTE: V1 APIs are being deprecated so V3 version is the stable version.

5. V1 APIs and V3 APIs comparison

V1 wrapper is not supported for MULTI-HRP i.e. patient and consents all are centralized, so multiple facilities or a service provider can't utilize the wrapper properly.

So, in V3 wrapper the patient is associated with facility id which is HIP-ID, so all the consents and patient care contexts can be handled facility wise.

For detailed changes check the documentation of v1_to_v3_migration_doc

6. Pre-requisites

6.1. **Install ABHA SBX PHR App on your mobile**.

- Download APK from here

6.2. **Create ABHA Address**

Skip if ABHA Address already exists, or else abhaAddress can be created in the below following ways

- Mobile Number
- Aadhaar Number

After creating the ABHA Address, your id should look like "yourAbha@sbx"

6.3. **System Requirements and Installations**

There are two ways to get wrapper and related applications running on your system:

a. **Using docker (Preferred):** This is an easy way to get wrapper up and running,*Install docker and docker-compose*: You can install docker desktop from here to get both.
System Requirements:
   - For Mac, check here
   - For Windows, check here
   - For Linux, check here

   Using default docker-compose.yaml, you can bring up wrapper and MongoDB services.

b. If you are facing issues with installing or running docker, then you can install **individual components**:

- Install MongoDB from here
- Install jdk 17. Instructions can be found here
- Install gradle from here
- System Requirements:
- For Mongodb, you can check here to understand resource requirements.
- For Java17, you can check here for compatible system configurations.
- Gradle version >= 8.5 should be fine.
- Recommended RAM: Systems with more than 8 GB RAM

This repository provides two other services:

- Sample HIP
- Sample HIU

If you need to bring these services up, then you need to install gradle from here

7. ABDM Setup

Register bridge (hostUrl) with ABDM for callbacks.

1) **Get Access Token.**
ClientId and secret will be provided once you applied in the portal

Method/ Use case	cURl

POST
*** Session/Access Token**
	`curl --location 'https://dev.abdm.gov.in/api/hiecm/gateway/v3/sessions' \`
	`--header 'Content-Type: application/json' \`
	`--header 'REQUEST-ID: 1b091f60-d188-4cd3-b2af-839c97fe978f' \`
	`--header 'TIMESTAMP: 2024-10-21T07:00:50.104Z' \`
	`--header 'X-CM-ID: sbx' \`
	`--data '{`
	`    "clientId": "<<Enter Client Id>>",`
	`    "clientSecret": "<<Enter Client Secret>>",`
	`    "grantType": "client_credentials"`
	`}'`

2) **Register bridge url**
M2 and M3 are an async API architecture, so in order to get the callbacks you need to register your server URL, It should be a domain which is SSL certified, no IP or port numbers are allowed.

Method/ Use case	cURl

PATCH
*** The callback URL**
	`curl --location --request PATCH 'https://dev.abdm.gov.in/api/hiecm/gateway/v3/bridge/url' \`
	`--header 'REQUEST-ID: 81e520a8-17d5-4ba1-bd52-9971fca80dba' \`
	`--header 'TIMESTAMP: 2024-10-21T06:59:07.083Z' \`
	`--header 'X-CM-ID: sbx' \`
	`--header 'Content-Type: application/json' \`
	`--header 'Authorization: Bearer {{accessToken}}' \`
	`--data '{`
	`    "url": "{{URL to be updated}}"`
	`}'`

3) **Check the BRIDGE URL and Facilities**

Once the call back URL is registered you can verify them by calling this API, it responds with all the facilities registered under the ClientID

Method/ Use case	cURl

GET
*** Checking the registered facilities and callback.**
	`curl --location 'https://dev.abdm.gov.in/api/hiecm/gateway/v3/bridge-services' \`
	`--header 'REQUEST-ID: 5b114388-51df-40f8-b171-7b1a30f64253' \`
	`--header 'TIMESTAMP: 2024-10-21T07:03:33.434Z' \`
	`--header 'X-CM-ID: sbx' \`
	`--header 'Authorization: Bearer {{accessToken}}'`

To register facility and its services you need to create a facility and add the services there.
Since we are handling with sandbox, website link

Once the above mentioned are completed, we can bring up the application and start testing, The crucial step in testing if to save the patient into wrapper, refer below for adding patient.

8. Bring the application up

- There are 3 application.properties.

    - application.properties for client credentials, proxy server details and version of ABDM APIs
    - application-v1.properties for setting the facility url.
    - application-v3.properties for setting the facility url.
- Provide clientId and clientSecret in application.properties
- If you have installed docker and docker compose then you can bring the application using: **docker-compose up --build**
- If you have chosen to install separate components, then here is how you can bring the services up:
    - Start mongodb (let the port be defaulted to 27017): Instructions on how to start can be found here The links like Install on Linux do have instructions on how to start the service as well.
    - Go to root of this repository and start wrapper by running gradle bootrun

Proxy Server Settings:

If wrapper needs to send requests using proxy server then please define the following properties in application.properties:

- **useProxySettings**=true
- **proxyHost**=your proxy server ip
- **proxyPort**=your proxy server port

9. List of Functionalities in Wrapper

1. Adding of patient is mandatory.
2. DeepLinking sms.
3. Linking of careContexts
    a. HIP-InitiatedLiking
    b. User-InitiatedLinking
4. Scan and share
5. Data Transfer
6. Consent Manager

9.1 Adding of the patient in the wrapper

- The Patients in v3 wrapper are stored along with the hipId, which makes the patient an unique all over the wrapper
- AbhaAddress and patientReference and hipId must be unique in the facility.
- PatientReference is the uniqueId present in the facility for that particular patient.
- All the fields are mandatory in the request.

cURL for adding patients

Method/ Use case	cURl

PUT
*** Adding patient into the wrapper**
	```
	curl --location --request PUT 'http: //localhost:8082/v3/add-patients'\
	--header 'Content-Type: application/json' \
	--data-raw '[
	    {
	        "name": "Govada Venu Ajitesh",
	        "abhaAddress": "ajitesh26x@sbx",
	        "patientReference": "ajitesh26x",
	        "gender": "M",
	        "dateOfBirth": "YYYY-MM-DD",
	        "patientDisplay": "Venu Ajitesh",
	        "patientMobile": "9999999999",
	        "hipId": "Predator_HIP"
	    }
	]'
	```

Response

HttpStatusCode	response	Reason

202	{
	“message”: “successfully upserted 1 patient”
	}
		Abha address and hipId are unique so if you want to update the patient they will get upserted.

202	{
	“message”: “successfully upserted 0 patient”
	}
		If the patient is already present in the wrapper, that patient will be not be inserted again so count 0

400	"errors": [
	{
	"error": {
	"code": 1000,
	"message": “abhaAddress is mandatory"
	}}
	]
		If the field abhaAddress is not present in the payload, wrapper will throw an validation error

400	"errors": [
	{
	"error": {
	"code": 1000,
	"message": “dateOfBirth must be in the format yyyy-MM-dd"
	}
	}
	]
		If the dateOfBirth was incorrect or doesn’t follow the format yyyy-MM-dd,
		this error will be thrown

9.2 DeepLinking sms

- The Deep linking functionality is for the patient who doesnt have an abhaAdress, this API sends an SMS to the patient to create an abhaAddress and link the existing records present in the facility.
- SMS will be sent by the ABDM.

cURL for DeepLinking sms:

Method/ Use case	cURl

POST * DeepLinking SMS	
	```
	curl --location 'http: //localhost:8082/v3/sms/notify' \
	--header 'Content-Type: application/json' \
	--data '{
	    "notification": {
	        "phoneNo": "9999999999",
	        "hip": {
	            "name": "Max Health",
	            "id": "demo-hip-Ajitesh"
	        }
	    }
	}'
	```

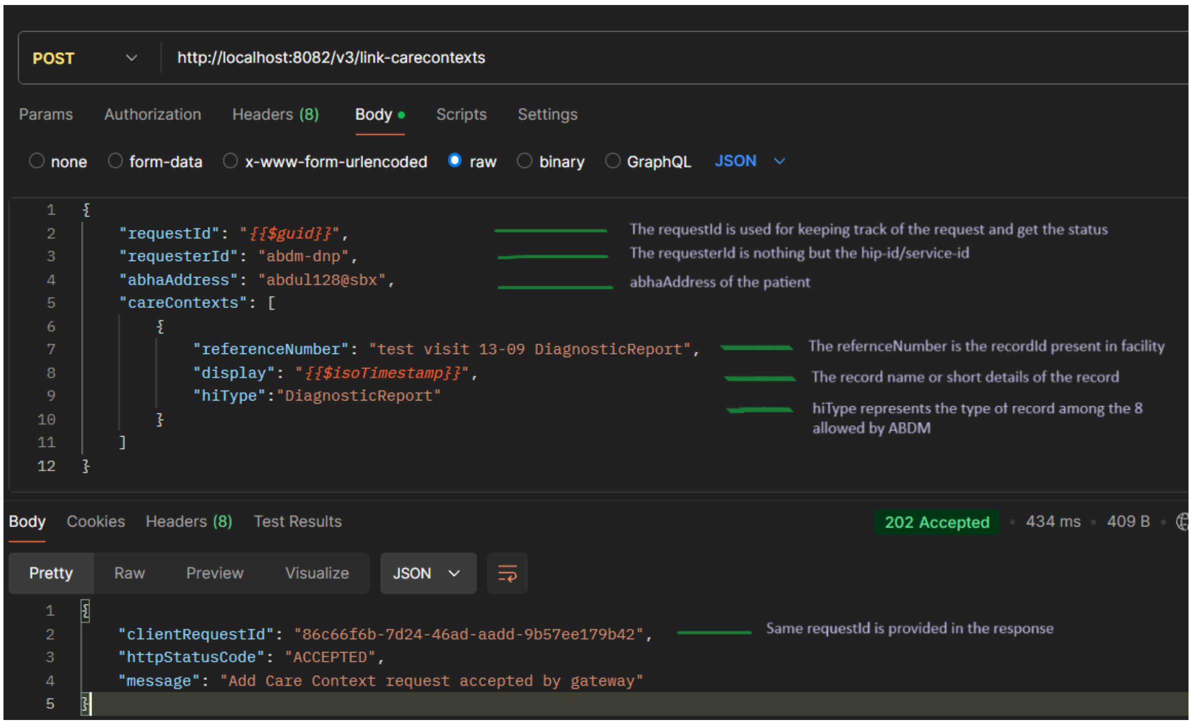
Response

HttpStatusCode	response	Reason
202	{ "clientRequestId": "86c66f6b-7d24-46ad-aadd-9b57ee179b42", "httpStatusCode": "ACCEPTED", "message": "Add Care Context request accepted by gateway"}	Successfully requested to ABDM
400	"errors": [{ "error": { "code": 1000, "message": “Error thrown by gateway during deepLinking SMS" } }]	There can be any reason why its failed, if the ABDM sends an error message it gets populated the same
400	"errors": [{ "error": { "code": 1000, "message": “DeepLinking SMS: Error while executing deepLinking SMS" } }]	Unknown exception while sending request to ABDM

9.4 HIP-Initiated linking

- Linking a record reference which is present in the facility with ABHA address.
- The care context can be any thing like a visit or a report, its upto the facility how to determine it.
- If you are updating an existing record, you can call the same API to notify other HIU for fetching the updated records.

DB view: The logs will be stored in **requestlogs**, and the care contexts are appended to the patient inside **patients**



http://localhost:8082/v3/link-carecontexts

```
{
    "requestId": "{{$guid}}",
    "requesterId": "abdm-dnp",
    "abhaAddress": "abdul128@sbx",
    "careContexts": [
        {
            "referenceNumber": "test visit 13-09 DiagnosticReport",
            "display": "{{$isoTimestamp}}",
            "hiType":"DiagnosticReport"
        }
    ]
}
```

The requestId is used for keeping track of the request and get the status
The requesterId is nothing but the hip-id/service-id
abhaAddress of the patient
The refernceNumber is the recordId present in facility
The record name or short details of the record
hiType represents the type of record among the 8 allowed by ABDM

```
{
    "clientRequestId": "86c66f6b-7d24-46ad-aadd-9b57ee179b42",
    "httpStatusCode": "ACCEPTED",
    "message": "Add Care Context request accepted by gateway"
}
```

Same requestId is provided in the response

cURL for linking careContexts:

Method/ Use case	cURl

POST
*** Linking careContexts**
	`curl --location 'http: //localhost:8082/v3/link-carecontexts' \`
	`--header 'Content-Type: application/json' \`
	`--data-raw '{`
	`    "requestId": "bdecf7dc-6fa9-4067-841b-da69a2f97266",`
	`    "requesterId": "abdm-dnp",`
	`    "abhaAddress": "abdul128@sbx",`
	`    "careContexts": [`
	`        {`
	`            "referenceNumber": "test visit 13-09 DiagnosticReport",`
	`            "display": "2024-09-15T12:48:59.831Z",`
	`            "hiType": "DiagnosticReport"`
	`        }`
	`    ]`
	`}'`

Internal working of the API:

Linking of the careContext needs a linkToken which is obtained by ABDM gateway.
A) check the patient and linkToken.
B) If patient not found in wrapper, wrapper will call facility for the patient details

a. If patient not found this api will respond with a bad request: patient not found error
b. If patient found, the wrapper will initiate a linkToken request and respond the facility with a success message
c. Once the linkToken if generated, careContext will be automatically gets linked.

C) if linkToken is expired, the wrapper will call the ABDM for generation of linkToken and responds back the facility.

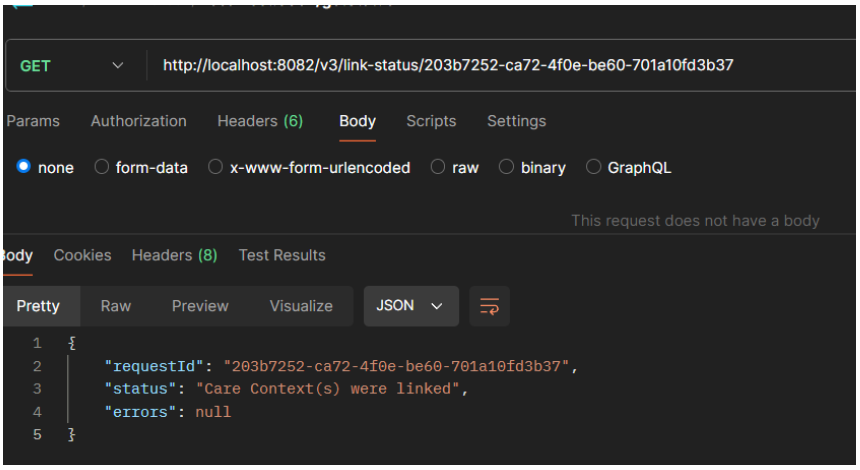
D) If both patient and linkToken are present the careContexts will be linked.

The List of responses can get from link/carecontexts API.

HttpStatusCode	response	Reason
202	{ "clientRequestId": "86c66f6b-7d24-46ad-aadd-9b57ee179b42", "httpStatusCode": "ACCEPTED", "message": "Add Care Context request accepted by gateway"}	Linking of the careContexts has been initiated.
202	{ "clientRequestId": "86c66f6b-7d24-46ad-aadd-9b57ee179b42", "httpStatusCode": "ACCEPTED", "message": " Request for link token generation ACCEPTED by gateway " }	Generation of careContexts has been done, once the linkToken is received linking of careContexts will be done.
400	"errors": [{ "error": { "code": 1000, "message": " Patient not found to generate linkToken" } }]	Patient not found in facility
400	"errors": [{ "error": { "code": 1000, "message": "Error Thrown by gateway while generating LinkToken" } }]	Unable to generate linkToken, Probable errors, demograpic details mismatch, those errors will be shown in the response.
400	"errors": [{ "error": { "code": 1000, "message": " Error thrown by Gateway for HIP Initiated link add care context " } }]	Unable to link CareContexts so check the errors below which are thrown by ABDM and check again.
400	"errors": [{ "error": { "code": 1000, "message": " Duplicate linkToken request " } }]	Kindly wait for some time and link again, frequent calls to the abdm blocks the requests.

9.5 Status of linking careContexts

- The requestId should be passed in the path paramater of the API to get the status.


GET http://localhost:8082/v3/link-status/203b7252-ca72-4f0e-be60-701a10fd3b37

Params Authorization Headers (6) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL

This request does not have a body

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
{
    "requestId": "203b7252-ca72-4f0e-be60-701a10fd3b37",
    "status": "Care Context(s) were linked",
    "errors": null
}
```

cURL for checking the status of careContext linking

Method/ Use case	cURl
GET *** Status of linking careContexts**	curl --location 'http://localhost:8082/v3/link-status/f162c0a7-9fee-4efd-a691-c4053a1042e9'

Response

HttpStatusCode	response	Reason

202	{
	"clientRequestId": "86c66f6b-7d24-46ad-aadd-9b57ee179b42",
	"httpStatusCode": "ACCEPTED",
	"message": "Add Care Context request accepted by gateway"
	}
		Linking of the careContexts has been initiated.

202	{
	"clientRequestId": "86c66f6b-7d24-46ad-aadd-9b57ee179b42",
	"httpStatusCode": "ACCEPTED",
	"message": " Request for link token generation ACCEPTED by gateway "
	}
		Generation of careContexts has been done, once the linkToken is received linking of careContexts will be done.

400	"errors": [
	{
	"error": {
	"code": 1000,
	"message": " Patient not found to generate linkToken"
	}
	}]
		Patient not found in facility

400	"errors": [{	Unable to generate linkToken,

	"error": {
	  "code": 1000,
	  "message": "Error Thrown by gateway while generating LinkToken"
	  }
	 }
	]
		Probable errors, demograpic details mismatch, those errors will be shown in the response.

400	"errors": [
	 {
	  "error": {
	   "code": 1000,
	   "message": " Error thrown by Gateway for HIP Initiated link add care context "
	  }}
	]
		Unable to link CareContexts so check the errors below which are thrown by ABDM and check again.

400	"errors": [
	 {
	  "error": {
	   "code": 1000,
	   "message": " Duplicate linkToken request "
	  }
	 }]
		Kindly wait for some time and link again, frequent calls to the abdm blocks the requests.

9.6 User initiated linking

The ABDM sends a discover request to the wrapper:

A) If patient not found wrapper call facility with - **{hipBaseUrl}/patient-discover**

Method/ Use case	cURl

POST
Discovery request
	{
	  "requestId": "499a5a4a-7dda-4f20-9b67-e24589627061",
	  "timestamp": "2024-09-14T13:20:01.931Z",
	  "transactionId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
	  "patient": {
	   "id": "<patient-id>@<consent-manager-id>",
	   "verifiedIdentifiers": [
	    {
	     "type": "MR",
	     "value": "+919800083232"

```
        }
      ],
      "unverifiedIdentifiers": [
        {
          "type": "MR",
          "value": "+919800083232"
        }
      ],
      "name": "chandler bing",
      "gender": "M",
      "yearOfBirth": 2000
    },
    "hipId": "Predator_HIP"
}
```

Response

HttpStatusCode	response	Reason

202	{"abhaAddress": "ajitesh6x@sbx",
	"name": "Venu Ajitesh",
	"gender": "M",
	"dateOfBirth": "2003-09-23",
	"patientReference": "APOLLO_12334",
	"patientDisplay": "Venu Ajitesh",
	"patientMobile": "9999999999",
	"hipId": "Predator_HIP",
	"careContexts": [
	{
	"referenceNumber": "ff219ebf-1959-4514-9d28-839b677d6fc6",
	"display": "Visit OP 20-06-2024",
	"hiType": "Prescription"
	}
	]}
		If patient and care context found.

400	{
	"error": {
	"code": "1000",
	"message": "Patient not found"
	}
	}
		Patient not found in facility

404		If patient not found, the facility can return 404 status code

B) If patient found wrapper calls facility with - **{hipBaseUrl}/patient-care-contexts**
If the patient is already present in the wrapper, this api will be invoked for the care contexts.

Method/ Use case	cURl

Discovery response	```{ "abhaAddress": "ajitesh6x@sbx", "hipId": "Predator_HIP" }```

Response

HttpStatusCode	response	Reason
202	```{ "abhaAddress": "ajitesh6x@sbx", "name": "Venu Ajitesh", "gender": "M", "dateOfBirth": "YYYY-MM-DD", "patientReference": "APOLLO_12334", "patientDisplay": "Venu Ajitesh", "patientMobile": "9999999999", "hipId": "Predator_HIP", "careContexts": [{ "referenceNumber": "guid", "display": "Visit OP 20-06-2024", "hiType": "Prescription" }] }```	If patient and care context found.
400	```{ "error": { "code": "1000", "message": " CareContexts not found" } }```	If there are no new care contexts, the facility can throw an error.

Since the care contexts are under hospital, they need to verified so facility needs to send an SMS to the patient.

a) **The wrapper will request for the otp: {hipBaseUrl}/request/otp**

Method/ Use case	cURl
POST Request OTP	```{ "requestId": "ff219ebf-1959-4514-9d28-839b677d6fc6", "abhaAddress": "ajitesh6x@sbx", "patientReference": "APOLLO_12334" }```

Response

HttpStatusCode	response	Reason

202	`{ "linkRefNumber": "ff219ebf-1959-4514-9d28-839b677d6fc6", "requestId": "ff219ebf-1959-4514-9d28-839b677d6fc7", "status": "SUCCESS" }`	Successfully requested to ABDM
400	`{ "error": { "code": "1000", "message": "CareContexts not found" }}`	There can be any reason why its failed, if the ABDM sends an error message it gets populated the same

b) **And to verify the otp: {hipBaseUrl}/verify/otp**

Method/ Use case	cURl
POST Verify OTP	`{ "requestId": "ff219ebf-1959-4514-9d28-839b677d6fc6", "loginHint": "Discovery otp verify", "authCode": "123456", "linkRefNumber": "ff219ebf-1959-4514-9d28-839b677d6fc7" }`

Response

HttpStatusCode	response	Reason
202	`{ "linkRefNumber": "ff219ebf-1959-4514-9d28-839b677d6fc6", "requestId": "ff219ebf-1959-4514-9d28-839b677d6fc7", "status": "SUCCESS" }`	Successfully requested to ABDM
400	`{ "error": { "code": "1000", "message": "CareContexts not found" } }`	There can be any reason why its failed, if the ABDM sends an error message it gets populated the same

9.7 Data Transfer

When someone is requesting for the data, the wrapper calls the below API to facility
{hipBaseUrl}/health-information

Method/ Use case	cURl
POST Data Transfer	`{ "hipId": "Predator_HIP", "careContextsWithPatientReferences": [ { "patientReference": "APOLLO_12334", "careContextReference": "ff219ebf-1959-4514-9d28-839b677d6fc6" }]`

	``` }```

Note: Kindly follow FHIR-Mapper documentation for generation of FHIR bundle.
The facility should respond like for data transfer request:

Response

HttpStatusCode	response	Reason

202	
	```
	{
	    "healthInformationBundle": [
	      {
	         "careContextReference": "OP visit 20-06-2024",
	         "bundleContent": "stringified FHIR bundle"
	      }
	    ]}
	```
		Successfully requested to ABDM

400	
	```
	{
	    "error": {
	       "code": "1000",
	       "message": "Unable to generate FHIR bundle"
	    }
	}
	```
		There can be any reason why its failed, if the ABDM sends an error message it gets populated the same

9.8 Consent Manager

For raising consent and get some records **/consent-init**

Method/ Use case	cURl

POST
Consent Init
	```
	curl --location 'http: //localhost:8082/v3/consent-init' \
	--header 'Content-Type: application/json' \
	--data-raw '{
	    "requestId": "6eb843a6-4904-4126-86df-b262ad31260a",
	    "timestamp": "2024-09-15T12:54:38.628Z",
	    "consent": {
	```

```
    "purpose": {
      "text": "Care Management",
      "code": "CAREMGT",
      "refUri": "wrapper"
    },
    "patient": {
      "id": "ajitesh3@sbx"
    },
    "hiu": {
      "id": "abdm-dnp"
    },
    "requester": {
      "name": "Dr. Venu AJitesh",
      "identifier": {
        "type": "REGNO",
        "value": "MH1001",
        "system": "https://www.mciindia.org"
      }
    },
    "hiTypes": [
      "DiagnosticReport",
      "DischargeSummary",
      "HealthDocumentRecord",
      "ImmunizationRecord",
      "OPConsultation",
      "Prescription",
      "WellnessRecord"
    ],
    "permission": {
      "accessMode": "VIEW",
      "dateRange": {
        "from": "2023-02-16T12:45:18.548Z",
        "to": "2024-09-07T11:43:18.548Z"
      },
      "dataEraseAt": "2025-07-28T07:41:30.171Z",
      "frequency": {
        "unit": "HOUR",
        "value": 1,
        "repeats": 0
      }
    }
  }}'
```

Response

HttpStatusCode	response	Reason

202	{
	    "clientRequestId": "86c66f6b-7d24-46ad-aadd-9b57ee179b42",
	    "httpStatusCode": "ACCEPTED",
	    "message": Consent init request accepted by gateway"}
		Consent request is accepted

400	"errors": [
	    {
	      "error": {
	        "code": 1000,
	        "message": "invalid refUri"
	      }
	    }
		refUri in purpose is mandatory, it ca be anything.

	]	

400	"errors": [
	{
	"error": {
	"code": 1000,
	"message": "the expiry date should not be greater than present date"
	}
	}
	]
		The permission expiry date should be more than the current timestamp.

400	"errors": [
	{
	"error": {
	"code": 1000,
	"message": " Duplicate consent request "
	}
	}
	]
		Kindly wait for some time and request again, frequent calls to the abdm blocks the requests.

9.9 Getting the status of consent

- /consent-status/{requestId}

Method/ Use case	cURl
POST **Verify OTP**	curl --location 'http://localhost:8082/v3/consent-status/71582e36-b039-43e8-80e6-0edf26833e5d'

Response

HttpStatusCode	response	Reason

200	{
	"status": "CONSENT_STATUS_ACCEPTED",
	"errors": null,
	"httpStatusCode": "OK",
	"initConsentRequest": null,
	"consentDetails": null
	}
		Consent request is accepted, but not granted.

200	{
	"status": "CONSENT_ON_NOTIFY_RESPONSE_RECEIVED",
	"errors": null,
	"httpStatusCode": "OK",
	"initConsentRequest": {
		This response shows 2 details the requested and the granted details here, the consent has been denied.

	"requestId": "92719971-02fd-4e7b-89b5-88f18dd984f6",
	"timestamp": "2024-09-16T14:22:06.198Z",
	"consent": {
	"purpose": {
	"text": "Care Management",
	"code": "CAREMGT",
	"refUri": "wrapper"
	},
	"patient": {
	"id": "ajitesh3@sbx"
	},
	"hip": null,
	"careContexts": null,
	"hiu": {
	"id": "abdm-dnp"
	},
	"requester": {
	"name": "Dr. Venu AJitesh",
	"identifier": {
	"type": "REGNO",
	"value": "MH1001",
	"system": "https://www.mciindia.org"
	}
	},
	"hiTypes": [
	"DiagnosticReport",
	"DischargeSummary",
	"HealthDocumentRecord",
	"ImmunizationRecord",
	"OPConsultation",
	"Prescription",
	"WellnessRecord"
	],
	"permission": {
	"accessMode": "VIEW",
	"dateRange": {
	"from": "2023-02-16T12:45:18.548Z",
	"to": "2024-09-07T11:43:18.548Z"
	},
	"dataEraseAt": "2025-07-28T07:41:30.171Z",
	"frequency": {
	"unit": "HOUR",
	"value": 1,
	"repeats": 0
	}
	}
	}
	},
	"consentDetails": {
	"consent": [
	{
	"status": "DENIED",
	"consentArtefacts": null
	}
	]
	}
	}	

200	{ "status": "CONSENT_FETCH_ACCEPTED", "errors": null, "httpStatusCode": "OK",	Consent has been granted and the details of the consent are provided below.

```
  "initConsentRequest": {
    "requestId": "8e3e2242-cea7-4806-bf93-
1b720e7798c4",
    "timestamp": "2024-09-16T15:05:50.351Z",
    "consent": {
      "purpose": {
        "text": "Care Management",
        "code": "CAREMGT",
        "refUri": "wrapper"
      },
      "patient": {
        "id": "ajitesh3@sbx"
      },
      "hip": null,
      "careContexts": null,
      "hiu": {
        "id": "abdm-dnp"
      },
      "requester": {
        "name": "Dr. Venu AJitesh",
        "identifier": {
          "type": "REGNO",
          "value": "MH1001",
          "system": "https://www.mciindia.org"
        }
      },
      "hiTypes": [
        "DiagnosticReport",
        "DischargeSummary",
        "HealthDocumentRecord",
        "ImmunizationRecord",
        "OPConsultation",
        "Prescription",
        "WellnessRecord"
      ],
      "permission": {
        "accessMode": "VIEW",
        "dateRange": {
          "from": "2023-02-16T12:45:18.548Z",
          "to": "2024-09-07T11:43:19.548Z"
        },
        "dataEraseAt": "2025-07-28T07:44:20.171Z",
        "frequency": {
          "unit": "HOUR",
          "value": 1,
          "repeats": 0
        }
      }
    }
  },
  "consentDetails": {
    "dateRange": {
      "from": "2023-02-16T12:45:18.548Z",
      "to": "2024-09-07T11:43:19.548Z"
    },
    "dataEraseAt": "2025-07-28T07:44:20.171Z",
    "hiTypes": [
      "DiagnosticReport",
      "DischargeSummary",
      "HealthDocumentRecord",
      "ImmunizationRecord",
      "OPConsultation",
      "Prescription",
```

	"WellnessRecord"], "consent": [{ "status": "GRANTED", "consentArtefacts": [{ "id": "cbeed37b-52db-4521-b895-92e5572aac2a", "lastUpdated": null, "hipId": "Demo_Ajitesh_HIP", "careContextReference": ["test visit-f9af123c-a723-49c2-b50b-617f4b1a5be0", "test visit-f7af123c-a723-45c2-b50b-617f4b1a5be0", " test visit-f9af123c-a723-45c2-b50b-617f4b1a5be5", " test visit-f9af123c-a723-45c2-b50b-617f4b1a5ie0", " test visit-f9af123c-a723-46c2-b50b-617f4b1a5be5", "test visit-f7ag123c-a723-45c2-b50b-617f4b1a5be0", "test visit-f9af123c-a823-49c2-b50b-617f4b1a5be0", " test visit-f7af71e0-7356-4f47-a5c9-28c620203b4e", " test visit-8d706393-41cb-4b26-b853-c6415bbcb3c2", " test visit-09aa5b05-6d95-49c9-a303-81b95190b04a", " test visit-3fbf7dfe-138e-46a2-a47a-dcdf235154bb", " test visit-d83df69e-7774-4268-8f4a-956069e53c8c"] }]}]}}	

400	"errors": [
	{
	"error": {
	"code": 1000,
	"message": "Patient not found"
	}
	}
	]
		The permission expiry date should be more than the current timestamp.

400	{
	"httpStatusCode": "BAD_REQUEST",
	"message": "Invalid arguments provided.",
	"errors": [
	{
	"error": {
	"code": "Wrapper-1001",
	"message": "Client request not found in database: 8e3e2242-cea7-4806-bf93-1b720e7798c"
	}
	}
	]
	}
		Check the requestId again,
		It should be same as the consent/init requestId

9.10 Request for fetching records

- Once the consent is granted you can request a fetch record to the HIP
- **/health-information/fetch-records**

Method/ Use case	cURl
POST **Verify OTP**	`curl --location 'http: //localhost:8082/v3/health-information/fetch-records' \ --header 'Content-Type: application/json' \ --data '{ "requestId": "60d4dd16-28b8-4ab6-9d8a-60999cabecb7", "consentId": "d8b0669f-6296-4a13-a016-18babdf39fb9", "requesterId": "Predator_HIP" }'`

Response

HttpStatusCode	response	Reason
202	{ "clientRequestId": "86c66f6b-7d24-46ad-aadd-9b57ee179b42", "httpStatusCode": "ACCEPTED", "message": health information request accepted by gateway"}	Successfully requested the data to the ABDM.
400	"errors": [{ "error": { "code": 1000, "message": “Consent expired" } }]	Consent has been expired so kindly raise again and fetch the records
400	"errors": [{ "error": { "code": 1000, "message": " Duplicate health information request " } }]	Kindly wait for some time and request again, frequent calls to the abdm blocks the requests.

9.11 Status for fetching records

- Checking the status if the HIP sent the FHIR bundles or not
- **/health-information/status/{requestId}**

Method/ Use case	cURl
POST **Verify OTP**	`curl --location 'http: //localhost:8082/v3/health-information/status/f3bf780e-b088-43da-9b63-a1c2d296973c'`

Response

HttpStatusCode	response	Reason

200	{
	    "status": " HEALTH_INFORMATION_REQUEST_SUCCESS",
	    "httpStatusCode": "200"
	}
		Request of data is success but waiting for the records from hip

200	{
	    "status": " Encrypted Health Information received by HIU from HIP",
	    "httpStatusCode": "202",
	    "decryptedHealthInformationEntries": [
	        {
	            "careContextReference": "ea4c5168-6419-4188-b076-f0e274bedfe3",
	            "bundleContent": "stringified FHIR bundle"
	        }
	    ]
	}
		After successful decryption of FHIR bundle they will be displayed here.

400	"errors": [
	    {
	      "error": {
	        "code": 1000,
	        "message": “Unable to decrypt the FHIR bundle"
	      }
	    }
	  ]
		The data which is sent by the hip is corrupted and unable to decrypt.

400	"errors": [
	    {
	      "error": {
	        "code": 1000,
	        "message": "Consent expired"
	      } }]
		The consent has been expired so you cant access the data.

10. Generic responses received from wrapper:

HttpStatusCode	response	Reason

403	`"code": 1000,`
	`"httpStatusCode": "FORBIDDEN",`
	`"message": "Kindly use v3 APIs",`
	`"error": {`
	`  "code": "1000",`
	`  "message": "v1 APIs are not allowed under v3 profile"`
	`   }`
		The profile was set to v3 in wrapper and if you call the wrapper with v1 APIs, you can’t access the wrapper. Kindly change it to v3 in application.properties file

400	`"error":{`
	`        "code": "1000",`
	`        "message": "v1 APIs are not allowed under v3 profile"`
	`}`
		In v1 the error is an object, just an code and message.

400	`"errors"`
	`[`
	`  {`
	`    "error": {`
	`     "code": "1000",`
		In v3 the errors are an array of error. multiple error objects in an array.

	`"message": "Unable to decrypt the FHIR bundle"`
	`}}]`	

400	`{`
	`"httpStatusCode": "BAD_REQUEST",`
	`"errors": [{`
	`"error": {`
	`"code": "Wrapper-1001",`
	`"message": "The required parameter 'hipId' is missing"`
	`}}]}`
		Some of the APIs require some parameters like hipId, so it is missing wrapper will throw this error.

500	`"errors": [{`
	`"error": {`
	`"code": 1000,`
	`"message": "Unknown exception"`
	`} }]`
		This exception is an last resort of error, all of the cases were handled but in any case raise an issue in GitHub if you encounter 500 status code.

503	`{`
	`"httpStatusCode": "SERVICE_UNAVAILABLE",`
	`"message": "Patient details not found",`
	`"errors": [`
	`{`
	`"error": {`
	`"code": "Wrapper-1001",`
	`"message": "HIP is currently unreachable for sharing patient details. Please try again later."`
	`}`
	`}]}`
		When facility service is unavailable the api will respond with status 503 error.

11. MongoDB of Wrapper

There are **7 tables** inside **adbm_wrapper** MongoDB

- Patients
- request-logs
- link-tokens
- consent-patient
- consent-requests
- consent-careContexts
- consent-cipher-key-mappings

patients: Patient table has all the details related to patient, their consent, their careContexts

request-logs: All the transaction logs stored with current timestamp in IST format

link-tokens: This table consists the linking token with abhaAddress and its expiry

consent-patient: This table has the abhaAddress and its related consentId's

consent-requests: This table has the gatewayRequestId along with its clientRequestId

consent-careContexts: This table has the list of careContexts related to consent

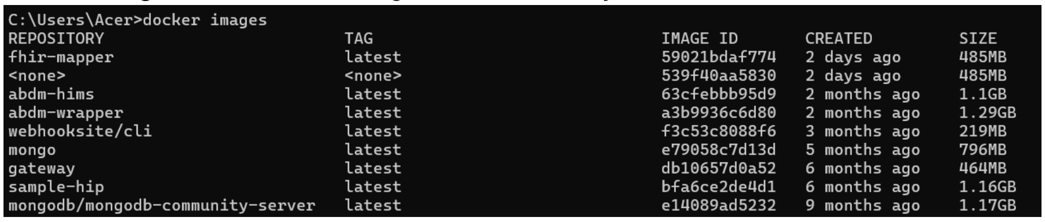
consent-cipher-key-mappings: This table has the keys related to encryption and decryption for both HIU and HIP.

12. Basic Docker commands

docker compose up –build: command to run the wrapper service. Add –build only new if new changes are in wrapper

docker ps: shows all the running images

docker images: shows all the images in our case they look like



```
C:\Users\Acer>docker images
REPOSITORY                          TAG                                 IMAGE ID       CREATED         SIZE
fhir-mapper                         latest                              59021bdaf774   2 days ago      485MB
<none>                              <none>                              539f40aa5830   2 days ago      485MB
abdm-hims                           latest                              63cfebbb95d9   2 months ago    1.1GB
abdm-wrapper                        latest                              a3b9936c6d80   2 months ago    1.29GB
webhooksite/cli                     latest                              f3c53c8088f6   3 months ago    219MB
mongo                               latest                              e79058c7d13d   5 months ago    796MB
gateway                             latest                              db10657d0a52   6 months ago    464MB
sample-hip                          latest                              bfa6ce2de4d1   6 months ago    1.16GB
mongodb/mongodb-community-server    latest                              e14089ad5232   9 months ago    1.17GB
```

docker stop {imageId}: Stops the service which is running

docker images prune: Deletes the unused and temporary images

13. Basic MongoDB commands

docker exec -it {imageId} mongosh: This will connect to the mongoDb

db.patients.find(): returns all the patients.
sample queries of MongoDB
Best way of handling the MongoDB is to connect to a MongoDB compass GUI and work with the queries as per your requirement or for the audit logs. Link to download compass

14. Sandbox to production migration

After completion of all the required flow implementation, there are few tests which needs to be followed.

1) Functional testing (FT)

- Functional testing can be done with the agencies provided on our website or it can be done at NHA.
- During the testing, all the functionalities of ABDM will be tested with your application END to END

2. Security audit

- WASA certification is needed for the exit process and it can be done with the empaneled agencies listed on our website.

3. HTC demo.

- Complete testing and audit of the application takes place.

For more detailed explanation about exit process kindly refer the documentation

Change the baseURL and enviormnet in the application.properties file.
and good to go

सत्यमेव जयते national health authority