Event Registry Service Documentation

1. **Introduction**

EVENT REGISTRY SERVICE API v1.0.0 OAS3

EVENT REGISTRY SERVICE CONSTANT

▶ **Error Codes** (Click here)

Servers

https://intdev-api.starhealth.in/e

Authorize

S
u
b
s
c
ri
b
e
r-
E
v
e
n
t

Subscriber-Event Mapping API's

APIs

crud operations

POST `/int/api/v1/subscribers-event/{mappingId}/update` UPDATE SUBSCRIBER EVENT MAPPING API

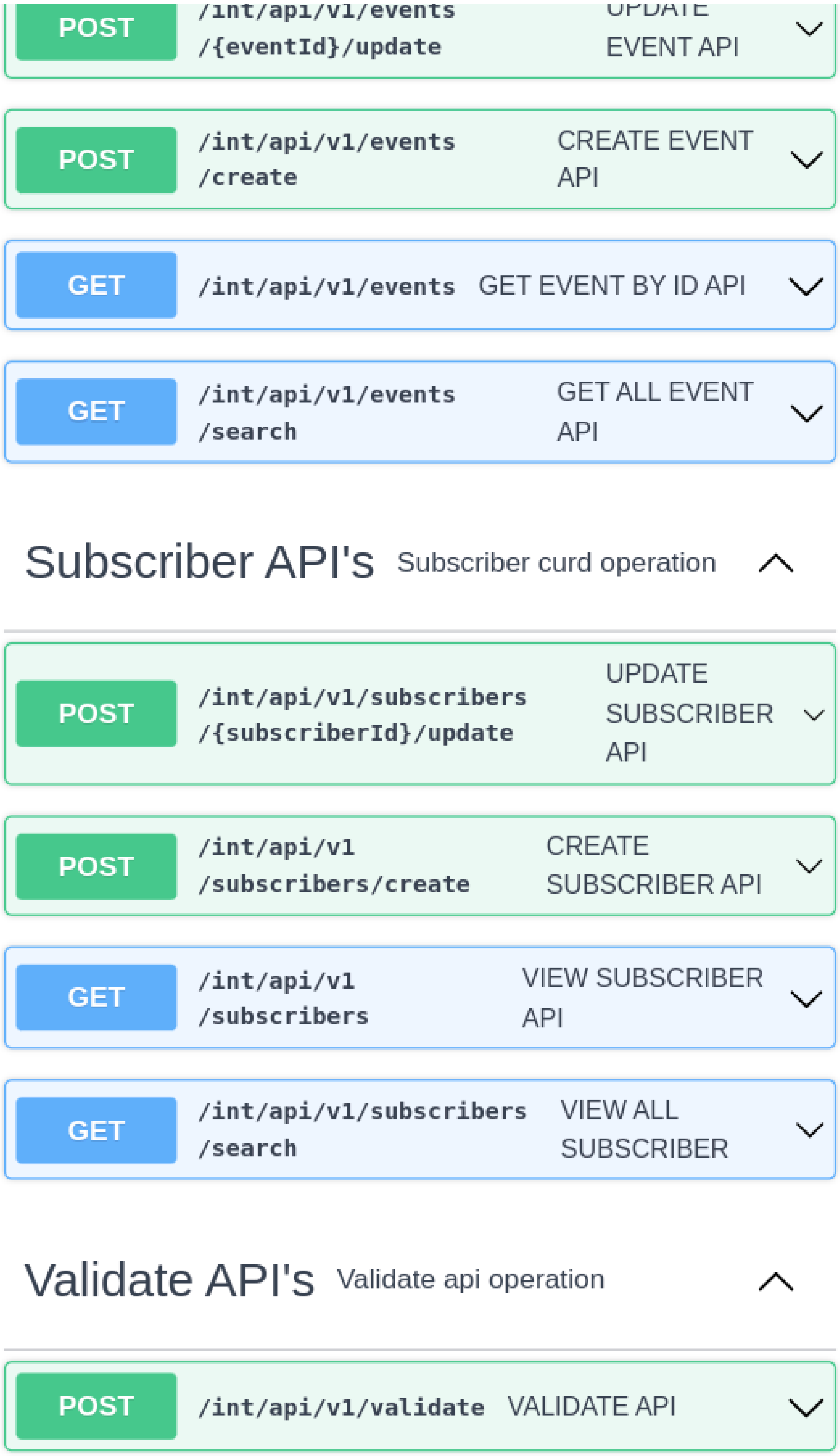
POST `/int/api/v1/subscribers-event/create` CREATE SUBSCRIBER EVENT API

GET `/int/api/v1/subscribers-event` VIEW SUBSCRIBER EVENT API

GET `/int/api/v1/subscribers-event/search` VIEW ALL SUBSCRIBER EVENT MAPPING

EVENT API

Event crud operations



POST /int/api/v1/events/{eventId}/update UPDATE EVENT API

POST /int/api/v1/events/create CREATE EVENT API

GET /int/api/v1/events GET EVENT BY ID API

GET /int/api/v1/events/search GET ALL EVENT API

Subscriber API's Subscriber curd operation

POST /int/api/v1/subscribers/{subscriberId}/update UPDATE SUBSCRIBER API

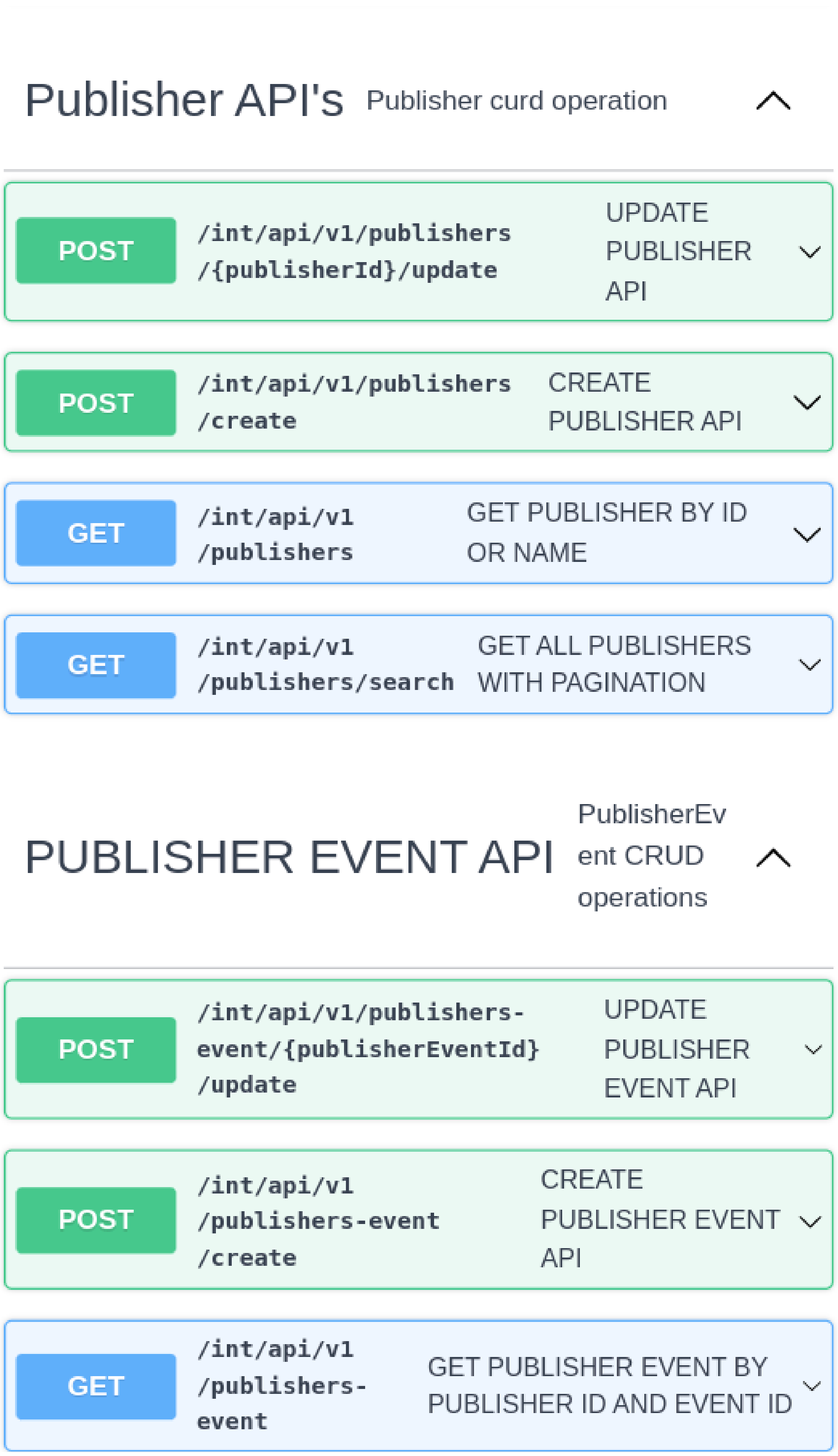
POST /int/api/v1/subscribers/create CREATE SUBSCRIBER API

GET /int/api/v1/subscribers VIEW SUBSCRIBER API

GET /int/api/v1/subscribers/search VIEW ALL SUBSCRIBER

Validate API's Validate api operation

POST /int/api/v1/validate VALIDATE API



Publisher API's Publisher curd operation

POST /int/api/v1/publishers/{publisherId}/update UPDATE PUBLISHER API

POST /int/api/v1/publishers/create CREATE PUBLISHER API

GET /int/api/v1/publishers GET PUBLISHER BY ID OR NAME

GET /int/api/v1/publishers/search GET ALL PUBLISHERS WITH PAGINATION

PUBLISHER EVENT API PublisherEvent CRUD operations

POST /int/api/v1/publishers-event/{publisherEventId}/update UPDATE PUBLISHER EVENT API

POST /int/api/v1/publishers-event/create CREATE PUBLISHER EVENT API

GET /int/api/v1/publishers-event GET PUBLISHER EVENT BY PUBLISHER ID AND EVENT ID

GET `/int/api/v1/publishers-event/all` GET ALL PUBLISHER EVENTS

Schemas

ValidateEventRequestDto ›

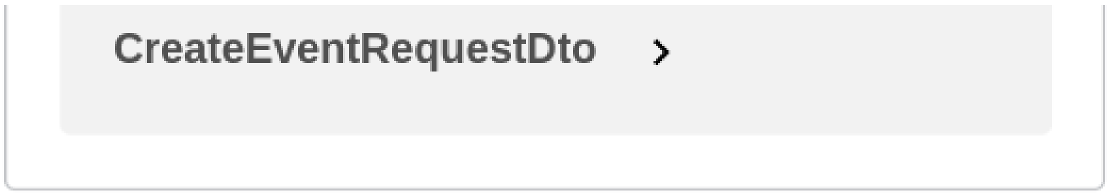
CreateSubscriberRequestDto ›

CreateSubscriberEventMappingRequestDto ›

PublisherRequestDto ›

UpdatePublisherEventRequestDto ›

PublisherEventRequestDto ›



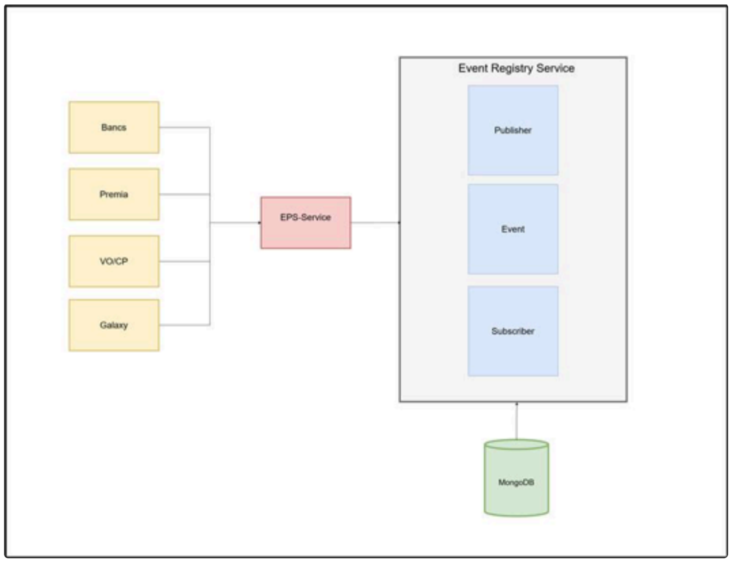
CreateEventRequestDto >

This document explains how the Event Registry Service works. It manages relationships between publishers (who send out events), subscribers (who receive events), and the events themselves.

The service allows users to:

- Create, update, and view events, publishers, and subscribers.
- Map events to publishers and subscribers to establish connections.

ER Diagram:



Event Registry HLD

2. **Data Model Overview**

The system uses several collections (like tables) to organize data:

Events

Stores details about each event.

Field Name	Data Type	Description
eventId	Integer	Unique ID for the event
eventName	String	Unique Name of the event
description	String	Brief description
createdDate	LocalDataTime	When the event was created
updatedDate	LocalDateTime	Last update time
status	Boolean	Whether the event is active (True/False)
schemaId	String	Unique ID for Schema
schemaVersion	String	Version of the data format
priority	String	Event importance (low, medium, high)
schemaValidationRequired	Boolean	If data validation is needed (True/False)
isSequence	Boolean	If sequence validation needed (True/False)

Publishers

Stores information about entities that publish events.

Field Name	Data Type	Description
publisher_id	String	Unique ID
publisherName	String	Publisher's name

description	String	Details about the publisher
status	Boolean	Active status (True/False)
createdDate	Date	Creation date
updatedDate	Date	Last update date
webhookDetails	Object	Contains Callback URL for the publisher

Subscribers

Stores details of entities that subscribe to events.

Field Name	**Data Type**	**Description**
subscriberId	Integer	Unique ID
subscriberName	String	Subscriber's name
description	String	Details about the subscriber
status	Boolean	Active status (True/False)
createdDate	Date	Creation date
updatedDate	Date	Last update date
topicName	String	Kafka topic of the subscriber

Mappings

Link publishers and events:

- **Publisher_Event Collection:** Connects publishers to events.

Field Name	Data Type	Description
publisherEventId	String	unique ID for the publisher event mapping
publisherId	Integer	ID of the publisher
eventId	Integer	Event mapped to the publisher
status	Boolean	Active status (True/False)
createdDate	Date	Creation date
updatedDate	Date	Last updated date
webhookEnabled	Boolean	webhook enable status (True/False)

- **Subscriber_Event Collection:** Connects subscribers to events.

Field Name	Data Type	Description
subscriberId	Integer	ID of the Subscriber
eventId	Integer	ID of the event mapped to the Subscriber
isActive	Boolean	Active status (True/False)
createdDate	Date	Creation date
updatedDate	Date	Last updated date
mappingId	String	unique ID for the subscriber event map

3. **API Endpoints with Request Bodies** [Will replace this swagger]

Event APIs

Create Event

- **Endpoint:** `POST int/api/v1/events/create`
- **Request Body Example:**

```
{
    "eventName": "DOCUMENT_INTITATED",
    "description": "document intitated",
    "status": true,
    "schemaId": "762348708732",
    "schemaVersion": "1.0",
    "priority": "MEDIUM",
    "schemaValidationRequired": true
  }
```

- **Response Body Example:**

```
{
    "responseCode": "200",
    "responseMessage": "SUCCESS",
    "displayMessage": "Event Detail Created Successfully",
    "data": {
        "eventID": "1374702085072216064"
    },
    "requestId": "682db1153238c476232cea69601caac2",
    "timestamp": 1747824917812
}
```

Update Event

- **Endpoint:** `POST int/api/v1/events/update`
- **Request Body Example:**

-
  ```
  {
      "eventName": "CREATE_INTIMATION",
      "description": "Create intimation event",
      "status": true,
      "schemaId": "00101010",
      "priority": "HIGH",
      "schemaValidationRequired": true
  }
  ```

- **Response Body Example:**

```
{
    "responseCode": "200",
    "responseMessage": "SUCCESS",
    "displayMessage": "Event Table Updated successfully",
    "data": {
        "eventId": "1374040013304619008",
        "eventName": "CREATE_INTIMATION",
```

```json
        "description": "Create intimation event",
        "createdDate": "2025-05-19T20:34:27.604",
        "updatedDate": "2025-05-20T11:26:54.258614400",
        "status": true,
        "schemaId": "00101010",
        "priority": "HIGH",
        "schemaValidationRequired": true
    },
    "requestId": "682c19a538cec56667ee302f158fe95b",
    "timestamp": 1747720614325
}
```

View Event

- **Endpoint:** `GET int/api/v1/events/{eventId}{eventName}`
- **Request Param :** `eventId` or `event`
- **Response Body Example:**

```json
{
    "responseCode": "200",
    "responseMessage": "SUCCESS",
    "displayMessage": "Event Details Fetched Successfully",
    "data": {
        "eventId": "1373973976601190400",
        "eventName": "INTIMATION_CREATED",
        "description": "nothig to say",
        "createdDate": "2025-05-19T16:12:03.223",
        "updatedDate": null,
        "status": true,
        "schemaId": "7330481174778421248",
        "priority": "HIGH",
        "schemaValidationRequired": true
    },
    "requestId": "682db1c2f0e43d863508ddcf95c189e2",
    "timestamp": 1747825090561
}

```

View All Events

- **Endpoint:** `GET int/api/v1/events`
- **Response Body Example:**

```json
{
    "responseCode": "200",
    "responseMessage": "SUCCESS",
    "displayMessage": "Event Details Fetched Successfully",
    "data": [
        {
            "eventId": "1373904311078215680",
            "eventName": "evet1",
            "description": "nothig to say",
            "createdDate": "2025-05-19T11:35:13.673",
            "updatedDate": "2025-05-19T11:35:13.673",
            "status": true,
            "schemaId": "fd4",
            "priority": "MEDIUM",
            "schemaValidationRequired": true
        },
        {
            "eventId": "1373904518272638976",
            "eventName": "event03",
            "description": "nothig to say",
            "createdDate": "2025-05-19T11:36:03.067",
            "updatedDate": "2025-05-19T11:45:23.920",
            "status": true,
            "schemaId": "fd4",
            "priority": "MEDIUM",
            "schemaValidationRequired": false
        },
        {
            "eventId": "1373919092224393216",
            "eventName": "intimation.create",
            "description": "Create intimation event",
            "createdDate": "2025-05-19T12:33:57.768",
            "updatedDate": "2025-05-19T13:34:17.121",
            "status": true,
            "schemaId": "00101010",
            "priority": "HIGH",
            "schemaValidationRequired": true
        },
        {
            "eventId": null,
            "eventName": "reversefeed.create",
            "description": "Create intimation event",
            "createdDate": null,
            "updatedDate": "2025-05-19T14:06:56.419",
            "status": true,
            "schemaId": "00101010",
            "priority": "HIGH",
            "schemaValidationRequired": true
        }
    ],
    "requestId": "682b382f5257d070b31cd4cd725fcc7f",
    "timestamp": 1747662895737
}
```

Publisher APIs

Create Publisher

- **Endpoint:** `POST int/api/v1/publisher/create`
- **Request Body Example:** `{`

  `"name": "OrderService",`

`"description": "Publishes order events",`
`"isActive": true}`

- **ResponseBody:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Publisher Added Successfully",

"data": null

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

Update Publisher

- **Endpoint:** `POST int/api/v1/publisher/update`
- **Request Body Example:** `{`
  `"name": "OrderServiceUpdated",`
  `"description": "Updated publisher description",`
  `"isActive": false`
  `}`
- **ResponseBody:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Publisher Updated Successfully",

"data": null

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

View Publisher

- **Endpoint:** `GET int/api/v1/publisher/view{publisherId}`
- **Request Param:** `publisherId`
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Publisher Fetched Successfully",

"data": [{

“publisherId”:” `0987654321` ”

`"name": "abc",`

“isActive”: True,

}]

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

View All Publisher

- **Endpoint:** `GET int/api/v1/publisher/view`
- **Response Body Example:{**

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Publisher Fetched Successfully",

"data": [{

“publisherId”:” `0987654321` ”

`"name": "abc",`

“isActive”: True,

}]

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

Subscriber APIs

Create Subscriber

- **Endpoint:** `POST int/api/v1/subscriber/create`
- **Request Body Example:** `{`
  `"name": "OrderTrackingService",`
  `"description": "Receives order events for tracking",`
  `"isActive": true`
  `}`
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Subscriber Created Successfully",

"data": [{

“subscriberId”:” `0987654321` ”

`"name": "abc",`

“isActive”: True,

}]

"requestId": "681d9a63a00d88cc3eda09c971649b63", "timestamp": 1746770533063

}

Update Subscriber

- **Endpoint:** `POST int/api/v1/subscriber/update`
- **Request Body Example:** `{`
  `"subscriberId": "1122334455",`
  `"name": "OrderTrackingServiceV2",`
  `"description": "Updated subscriber info",`
  `"isActive": true`
  `}`
- **ResponseBody:**

```
{
  "responseCode": "200",
  "responseMessage": "SUCCESS",
  "displayMessage": "Subscriber Updated Successfully",
   "data": {
   "name": "OrderTrackingService",
   "description": "Receives order events for tracking",
   "isActive": true
   }
  "requestId": "681d9a63a00d88cc3eda09c971649b63",
  "timestamp": 1746770533063
}
```

View Subscriber

- **Endpoint:** `GET int/api/v1/subscriber/view{subscriberId}`

- **Request Param:** `subscriberId`
- **Response Body Example:**

`{`

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Subscriber Fetched Successfully",

"data": [{

“subscriberId”:” `0987654321` ”

`"name": "abc",`

“isActive”: True,

}]

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

`}`

View All Subscriber

- **Endpoint:** `GET int/api/v1/subscriber/view`
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Subscriber Fetched Successfully",

"data": [{

“subscriberId”:” `0987654321` ”

`"name": "abc",`

“isActive”: True,

}]

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

Publisher-Event Mappings

Create Publisher-Event Mapping

- **Endpoint:** `POST int/api/v1/publisher_event/create`
- **Request Body Example:** `{`
  `"publisherId": "0987654321",`
  `"eventId": "1234567890",`
  `"isActive": true`
  `}`
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Publisher Mapped to the Event Successfully",

"data": null

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

Update Publisher-Event Mapping

- **Endpoint:** `POST int/api/v1/publisher_event/update`

- **Request Body Example:** `{`
  `"publisherId": "0987654321",`
  `"eventId": "1234567890",`
  `"isActive": false`
  `}`
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Publisher Event Mapping Updated Successfully",

"data": null

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

View Publisher-Event Mappings

- **Endpoint:** `GET int/api/v1/publisher_event/view` {publisherId}{eventId}
- **Request Param:** {publisherId}{eventId}
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Publisher Event Mapping fetched Successfully",

"data": [{

- `"publisherId": "0987654321",`
  `"eventId": "1234567890",`
  `"isActive": false`

}]

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

View All Publisher-Event Mappings

- **Endpoint:** `GET int/api/v1/publisher_event/view`
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Publisher Event Mapping fetched Successfully",

"data": [{

- `"publisherId": "0987654321",`
  `"eventId": "1234567890",`
  `"isActive": false`

}]

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

Subscriber-Event Mappings

Create Subscriber-Event Mapping

- **Endpoint:** `POST int/api/v1/subscriber_event/create`
- **Request Body Example:** `{`
  `"subscriberId": "1122334455",`
  `"eventId": "1234567890",`

`"isActive": true`
`}`

- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Subscriber Event Mapping created Successfully",

"data": null

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

Update Subscriber-Event Mapping

- **Endpoint:** `POST int/api/v1/subscriber_event/update`
- **Request Body Example:** `{`
  `"subscriberId": "1122334455",`
  `"eventId": "1234567890",`
  `"isActive": false`
  `}`
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Subscriber Event Mapping Updated Successfully",

"data": null

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

View Subscriber-Event Mappings

- **Endpoint:** `GET int/api/v1/subscriber_event/view`{subscriberId}{eventID}
- **Request Param:** {subscriberId}{eventID}
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Subscriber Event Mapping fetched Successfully",

"data": [{

- `"subscriberId": "0987654321",`
  `"eventId": "1234567890",`
  `"isActive": false`

}]

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

View All Subscriber-Event Mappings

- **Endpoint:** `GET int/api/v1/subscriber_event/view`
- **Response Body Example:**

{

"responseCode": "200",

"responseMessage": "SUCCESS",

"displayMessage": "Subscriber Event Mapping fetched Successfully", "data": [{

- `"publisherId": "0987654321",`
  `"eventId": "1234567890",`
  `"isActive": false`

}]

"requestId": "681d9a63a00d88cc3eda09c971649b63",

"timestamp": 1746770533063

}

4. **Database Details**

The data is stored in **MongoDB**, a NoSQL database, with each collection representing a component of the data model.

5. **Unique ID Generation**

To generate unique identifiers for publishers, subscribers, and events, the system uses the **Snowflake algorithm**. This ensures each ID is unique, scalable, and efficiently generated.