**Member Management Service – Low Level Design Document**

**1. Overview**

The Member Management Service is responsible for handling member registration, profile updates, and membership lifecycle. It exposes RESTful APIs to create, update, fetch, and deactivate members. It runs independently and is discoverable via Eureka.

**2. Functional Scope**

* Register new members
* Retrieve members by ID or email
* Update member details
* Manage membership status (ACTIVE / INACTIVE)
* Optional: search and filter

**3. Tech Stack**

|  |  |
| --- | --- |
| **Layer** | **Technology** |
| Language | Java 17+ |
| Framework | Spring Boot 3.x |
| API Docs | Springdoc OpenAPI |
| Persistence | Spring Data JPA |
| DB | MySQL |
| Service Discovery | Eureka |
| Gateway Routing | Spring Cloud Gateway |

**4. API Design**

|  |  |  |
| --- | --- | --- |
| **Method** | **Endpoint** | **Description** |
| POST | /api/members | Register new member |
| PUT | /api/members/{id} | Update member info |
| GET | /api/members/{id} | Get member by ID |
| GET | /api/members?email=email | Get member by email |

**5. Data Model**

**Entity: Member**

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| memberId | BIGINT | Primary key, auto generated |
| name | VARCHAR | Member's full name |
| email | VARCHAR | Unique |
| phone | VARCHAR |  |
| address | VARCHAR |  |
| membershipStatus | ENUM | ACTIVE / INACTIVE |
| registrationDate | DATE | Date of member onboarding |

**6.** **Service Layer**

**Responsibilities:**

* Validate incoming data
* Handle update and exception logic
* Communicate with JPA repository

**7. Controller Layer**

**Responsibilities:**

* Expose public REST endpoints
* Validate and transform input
* Delegate logic to the service layer
* Return appropriate HTTP codes