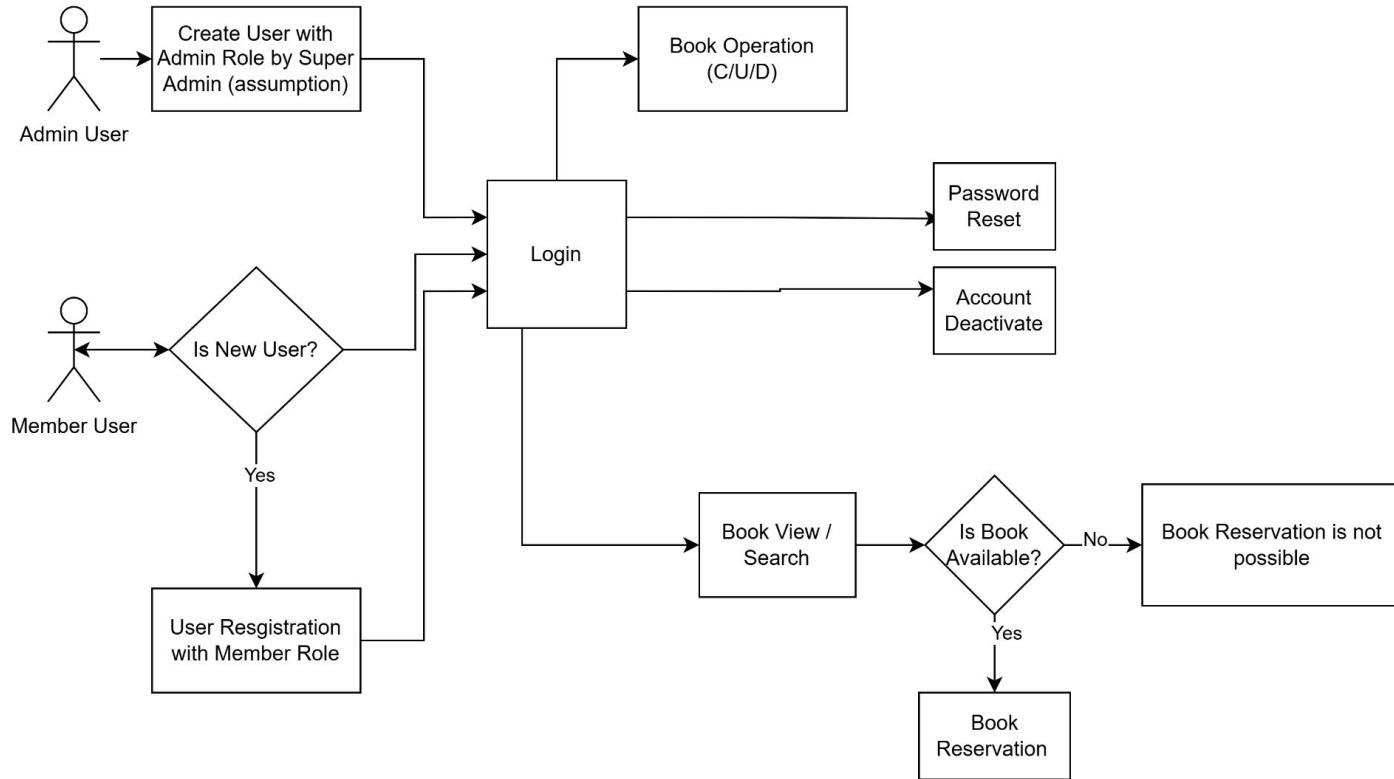


# Library Book Reservation System

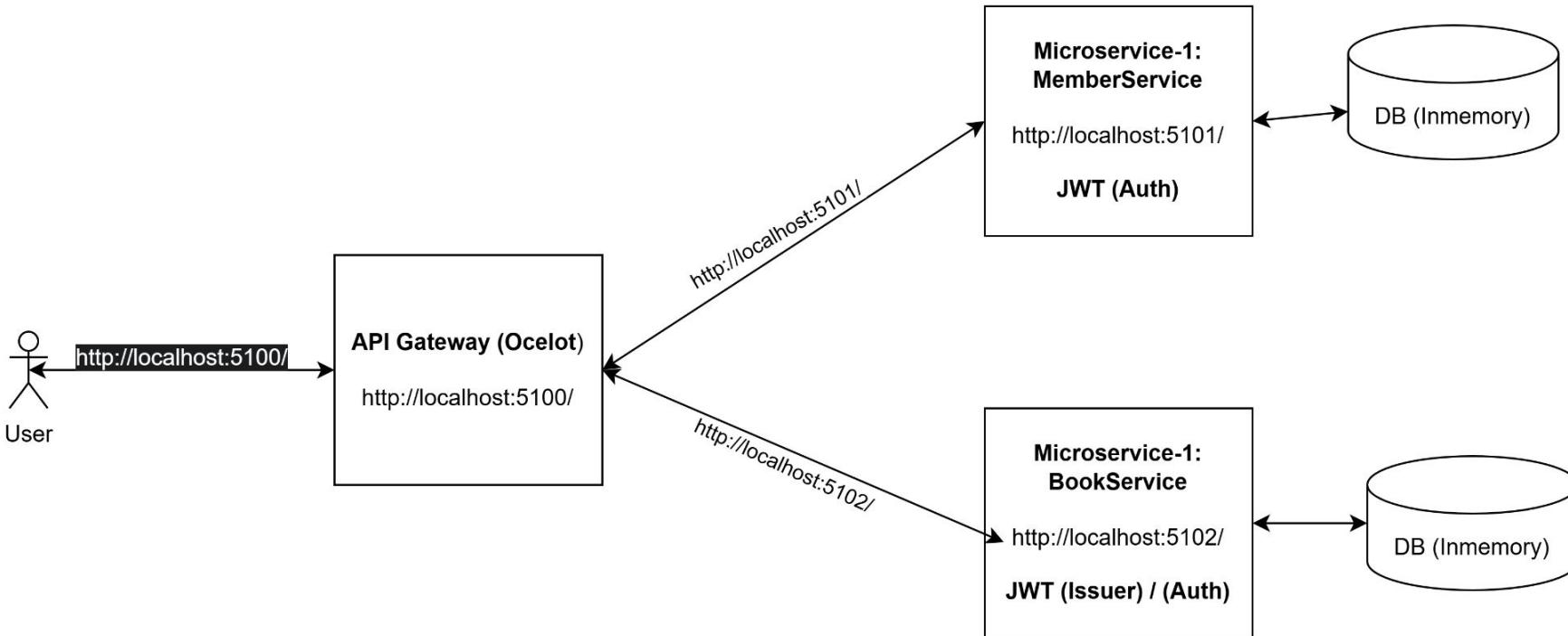
# Library Book Reservation System

## High-Level Workflow Design



# Library Book Reservation System

## Software Architecture



# List of completed components on the Test Projects

- Microservice projects with multi-tier architecture
  - Member Service project
  - Book Service project
- API Gateway project using Ocelot
- Implemented JWT based authentication & role-based authorization
- Implemented Pagination
- Implemented Serilog for daily rolling logs
- Implemented error handling with logging

# List of Role and Method names

Allow Anonymous can access the methods below:

- Member > Userlogin
- Member > UserAdd
- Member > AdminAdd

Admin Role can access the methods below:

- Book > Add Book
- Book > Update Book
- Book > Delete Book
- Member > Update
- Member > GetbyUserID
- Member > GetAll
- Member > Update

Member Role can access the methods below:

- Reservation > AddReservation
- Book > Search Books
- Book > GetReservedBooks

Both roles can access the methods below:

- Book > Get book by ID
- Book > Get all books
- Member > useraccountdeactivate
- Member > UserPasswordReset

# Postman collection list : Collection file attached for your reference

The screenshot shows the Postman application interface. On the left, the 'My Workspace' sidebar displays collections, environments, history, flows, and files. A search bar at the top right allows for searching Postman. The main area shows a collection named 'LBRS' with three main categories: 'Members', 'Books', and 'Reservations'. The 'Books' category is highlighted with a blue selection bar and contains several API endpoints. One specific endpoint, 'GET http://localhost:5100/api/books/searchbooks', is selected and highlighted with an orange border. The right panel provides detailed information for this selected endpoint, including its URL (`http://localhost:5100/api/books/searchbooks`), method (GET), and query parameters. The 'Body' tab shows the JSON response received from the API call.

**LBRS**

- Members**
  - `POST http://localhost:5100/api/members/memberadd`
  - `POST http://localhost:5100/api/members/adminadd`
  - `POST http://localhost:5100/api/members/userlogin`
  - `GET http://localhost:5100/api/members/getbyuserid`
  - `GET http://localhost:5100/api/members/getall`
  - `POST http://localhost:5100/api/members/update`
  - `POST http://localhost:5100/api/members/useraccountdeactivate`
  - `POST http://localhost:5100/api/members/userpasswordreset`
  - `GET http://localhost:5100/api/members/healthcheck`
- Books**
  - `POST http://localhost:5100/api/books/add`
  - `POST http://localhost:5100/api/books/update`
  - `POST http://localhost:5100/api/books/delete`
  - `GET http://localhost:5100/api/books/searchbooks` **Selected**
  - `GET http://localhost:5100/api/books/getall`
  - `GET http://localhost:5100/api/books/get`
- Reservations**
  - `POST http://localhost:5100/api/reservations/add`

**GET** `http://localhost:5100/api/books/searchbooks`

Params

Key	Value
genre	sof
author	eri
pageNumber	1
pageSize	2

Body

```
{ } JSON > Preview > Visualize >
```

```
3   "success": true,
4   "message": "Books retrieved successfully.",
5   "data": [
6     {
7       "bookId": "0e413a81-13f0-4d70-a6c9-79c7352eb27e",
8       "title": "Domain-Driven Design",
9       "isbn": "978-0321125224",
10      "author": "Eric Evans",
11      "genre": "Software Design",
12      "publishedYear": 2003,
13      "publisher": "Addison-Wesley",
14      "description": "Edited something",
15      "totalCopies": 1,
16      "availableCopies": 1,
17      "isActive": true,
18      "isAvailable": true
19    }
]
```

Thank you