LIBRARY MANAGEMENT SYSTEM

Higher National Diploma in Software Engineering

EAD-2 Project Documentation

24.1F

National Institute of Business Management

Kandy Regional Center

No 2, Asgiriya Road,

Kandy

LIBRARY MANAGEMENT SYSTEM

Higher National Diploma in Software Engineering

EAD-2 Project Documentation

24.1F

M.M.M AMRY	-	KAHDSE24.1F - 023
A. DHANUSHANANDAN	-	KAHDSE24.1F - 028
M.A.M AMMAR	-	KAHDSE24.1F – 026
M.Z.F.ZEENA	-	KAHDSE24.1F - 022

The project is submitted in partial fulfilment of the requirement of the Higher National Diploma of Software Engineering of National Institute of Business Management.

October 2024

DECLARATION

We declare this report was not a copy of a document done by any organization, university or any other institute and was not copied from the internet or other sources. This document is proprietary and exclusive property of the following mentioned group. No part of this document in whole or in part, may be reproduced, stored, transmitted, or used for design purposes without the prior written permission of NIBM. This report is a unique document and all the members actively participated in its accomplishment of it.

REGISTER NUMBER	Name	Signature
KAHDSE24.1F-028	A.DHANUSHANANDAN	
KAHDSE24.1F-023	M.M.M AMRY	
KAHDSE24.1F-026	M.A.M AMMAR	
KAHDSE24.1F-022	M.Z.F.ZEENA	

Certified by:

Lecturer	: Mr. Manjula kulathunga.
Date of submission	: 18/10/2024
Signature	:

Contents

DECLARATION 3
INTRODUCTION 5
  INTRODUCTIONS TO BUSINESS 5
  TOOLS & TECHNOLOGIES 5
DEPENDENCY USED IN THE SPRING BOOT 5
ATTRIBUTES AND RESOURCES 6
API END POINTS 6
REQUEST & RESPONSE FORMAT 8
  1.Book 8
  GET 8
  POST 10
  PUT 11
  DELETE 12
  2.Member 12
  GET 12
  POST 13
  PUT 14
  DELETE 15
  3.Membership 15
  GET 15
  POST 17
  PUT 17
  DELETE 18
USED DATABASE TABLES 19
  TABLES STRUCTURES 19
FRONT END INTERFACES 20
TEST CASES AND RESULTS 22
REFERENCES 30
PROJECT GIT LINK 30

INTRODUCTION

This Project aims to create a RESTful API using Java Spring Boot for Library management system. In our project we include CRUD operations for the functions Like Books, Member and Membership. Included a database to store the data permeant.

INTRODUCTIONS TO BUSINESS

We develop a RESTful API for the Library management system to manage Books, Members details, and track their Membership details. Our system helps to easily retrieve or add new Books, Members or Memberships.

TOOLS & TECHNOLOGIES

- We used the Spring Boot For create RESTful API.
- Used IDE: IntelliJ IDE 24.1.4.
- Front-End: HTML, CSS, and JavaScript.
- Database: PhpMyAdmin.

DEPENDENCY USED IN THE SPRING BOOT

1. Spring Web:
2. Spring Data JPA:
3. MySQL Driver:
4. Spring Boot DevTools:

ATTRIBUTES AND RESOURCES

1.Books

Id : String => Used as a primary key

Title: String => Book Name

Author: String => Book Written by

2.Members

Id : String => Used as a primary key

Name: String => Member Name

Email: String => Verification

Phone: String => Contact the Member

3.Membership

Id : String => Used as a primary key

Member id: String => Find the member details

Membership end date: String => Membership ending date

API END POINTS

All 3 Resources have CRUD operations.

1.Books

GET **:** http://localhost:8080/LMS/Books =Endpoint link to select all Books

http://localhost:8080/LMS/Books/002 =Select specific book using id/{ID}

End Point = /LMS/Books & /LMS/Books/{ID}

POST **:** http://localhost:8080/LMS/Books =Endpoint link to Insert Book

End Point = /LMS/Books

PUT : http://localhost:8080/LMS/Books/002 =Endpoint link to update using ID

End Point = /LMS/Books/{ID}

DELETE : http://localhost:8080/LMS/Books/002 =Endpoint link to delete using ID

End Point =/LMS/Books/{ID}

2.Member

GET : http://localhost:8080/LMS/Member =Endpoint link to select all members

http://localhost:8080/LMS/Member/002 =Select specific member using id/{ID}

End Point = /LMS/Member & /LMS/Member/{ID}

POST : http://localhost:8080/LMS/Member =Endpoint link to Insert Member

End Point = /LMS/ Member

PUT : http://localhost:8080/LMS/Member/002 =Endpoint link to update using ID

End Point = /LMS/ Member /{ID}

DELETE : http://localhost:8080/LMS/Member/002 =Endpoint link to delete using ID

End Point = /LMS/ Member /{ID}

2.Membership

GET : http://localhost:8080/LMS/Membership =Endpoint link to select all memberships

http://localhost:8080/LMS/Membership/002 =Select specific membership using id/{ID}

End Point = /LMS/Membership & /LMS/ Membership /{ID}

POST : http://localhost:8080/LMS/Membership =Endpoint link to Insert Membership

End Point = /LMS/ Membership

PUT : http://localhost:8080/LMS/Membership/002 =Endpoint link to update using ID

End Point = /LMS/ Membership /{ID}

DELETE : http://localhost:8080/LMS/Membership/002 =Endpoint link to delete using ID

End Point = /LMS/ Membership /{ID}

REQUEST & RESPONSE FORMAT

1.Book

GET

//Select all the Books

Request format

Http Method = GET

URL = http://localhost:8080/LMS/Books

FORMAT = JOSN

GET don't have a Request body only link is necessary.

Response format

```
    [
{
  "id": "002",
  "title": "Ben Doe",
  "author": "BEN"
},
  {
    "id": "003",
    "title": "Black Perl",
    "author": "CapJ"
  },
  {
    "id": "004",
    "title": "anime",
    "author": "kishimoto"
  }
]
```

//Select a Specific Book

Request format

Http Method = GET

URL = http://localhost:8080/LMS/Books/002

FORMAT = JOSN

GET don’t have a Request body only link is necessary.

Response format

```
          [
  {
    "id": "002",
    "title": "Ben Doe",
    "author": "BEN"
  },
]
```

POST

Request format

Http Method = POST

URL = http://localhost:8080/LMS/Books

FORMAT = JOSN

BODY=

```
{
        "id": "001",
        "title": "Black Perl",
        "author": "CapJ"
}
```

Response format

```
{
         "book": {
                "id": "001",
                "title": "Black Perl",
                "author": "CapJ"
                },
        "message": "Book added complete"
}
```

PUT

Request format

Http Method = PUT

URL = http://localhost:8080/LMS/Books/003

FORMAT = JOSN

BODY=

```
{
    "id": "003",
    "title": "ASD",
    "author": "ASD"
}
```

Response format

```
{
    "book": {
        "id": "003",
        "title": "ASD",
        "author": "ASD"
            },
    "message": "Book updated complete"
}
```

DELETE

Request format

Http Method = DELETE

URL = http://localhost:8080/LMS/Books/003

FORMAT = JOSN

BODY=

```
{
    "id": "003",
   "title": "ASD",
    "author": "ASD"
}
```

Response format

Book deleted.

2.Member

GET

//Select all the Member

Request format

Http Method = GET

URL = http://localhost:8080/LMS/Member

FORMAT = JOSN

GET don’t have a Request body only link is necessary.

Response format

```
            [
  {
    "id": "001",
    "name": "asd",
    "email": "asd@gmail.com",
    "phone": "0712051203"
  },
  {
    "id": "002",
    "name": "BEN_10",
    "email": "BEN@gmail.com",
    "phone": "0712051203"
  }
]
```

//Select a Specific Member

Request format

Http Method = GET

URL = http://localhost:8080/LMS/Member/002

FORMAT = JOSN

GET don't have a Request body only link is necessary.

Response format

```
            {
  "id": "002",
  "name": "BEN_10",
  "email": "BEN@gmail.com",
  "phone": "0712051203"
}
```

POST

Request format

Http Method = POST

URL = http://localhost:8080/LMS/Member

FORMAT = JOSN

BODY=

```
                                {

     "id": "003",
      "name": "CapJ",
      "email": "CapJ@gmail.com",
      "phone": "0712051203"
}
```

Response format

```
           {

   "message": "member added complete",
   "Member": {
      "id": "003",
      "name": "CapJ",
      "email": "CapJ@gmail.com",
      "phone": "0712051203"
   }
}
```

PUT

Request format

Http Method = PUT

URL = http://localhost:8080/LMS/Member/001

FORMAT = JOSN

BODY=

```
                                {

     "id": "001",
      "name": "CapJ",
      "email": "POC@gmail.com",
      "phone": "0712051203"
}
```

Response format

```
          {
  "message": "Member updated complete",
  "Member": {
    "id": "001",
    "name": "CapJ",
    "email": "POC@gmail.com",
    "phone": "0712051203"
  }
}
```

DELETE

Request format

Http Method = DELETE

URL = http://localhost:8080/LMS/Member/001

FORMAT = JOSN

BODY=

```
                         {
   "id": "001",
    "name": "CapJ",
    "email": "POC@gmail.com",
    "phone": "0712051203"
}
```

Response format

Member deleted.

3.Membership

GET

//Select all the Membership

Request format

Http Method = GET

URL = http://localhost:8080/LMS/Membership

FORMAT = JOSN

GET don't have a Request body only link is necessary.

Response format

```
[
  {
    "id": "001",
    "m_id": "002",
    "membership_end": "2024/10/15"
  },
  {
    "id": "002",
    "m_id": "001",
    "membership_end": "2024-10-19"
  },
  {
    "id": "003",
    "m_id": "001",
    "membership_end": "2024/10/15"
  },
  {
    "id": "004",
    "m_id": "001",
    "membership_end": "2024-10-08"
  }
]
```

//Select a Specific Membership

Request format

Http Method = GET

URL = http://localhost:8080/LMS/Membership/002

FORMAT = JOSN

GET don't have a Request body only link is necessary.

Response format

```
{
  "id": "002",
  "m_id": "001",
  "membership_end": "2024-10-19"
}
```

POST

Request format

Http Method = POST

URL = http://localhost:8080/LMS/Membership

FORMAT = JOSN

BODY=

```
{
    "id": "005",
    "m_id": "001",
    "membership_end": "2024-10-19"
}
```

Response format

```
{
  "Membership": {
    "id": "005",
    "m_id": "001",
    "membership_end": "2024-10-19"
  },
  "message": "Membership added complete"
}
```

PUT

Request format

Http Method = PUT

URL = http://localhost:8080/LMS/Membership/005

FORMAT = JOSN

BODY=

```
{
    "id": "005",
    "m_id": "003",
    "membership_end": "2024-10-19"
}
```

Response format

```
{
  "Membership": {
    "id": "005",
    "m_id": "003",
    "membership_end": "2024-10-19"
  },
  "message": "Membership updated complete"
}
```

DELETE

Request format

Http Method = DELETE

URL = http://localhost:8080/LMS/Member/005

FORMAT = JOSN

BODY=

```
        {

    "id": "005",
    "m_id": "003",
    "membership_end": "2024-10-19"
}
```

Response format

Membership deleted

USED DATABASE TABLES

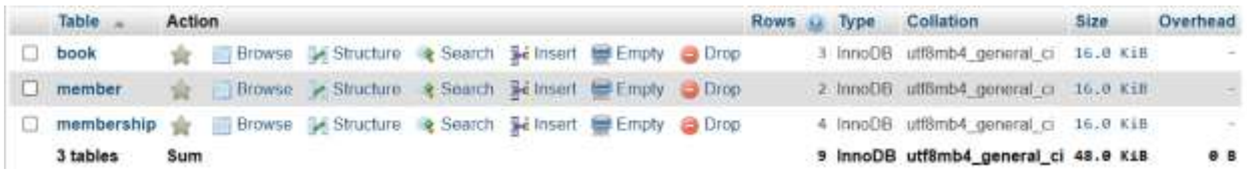
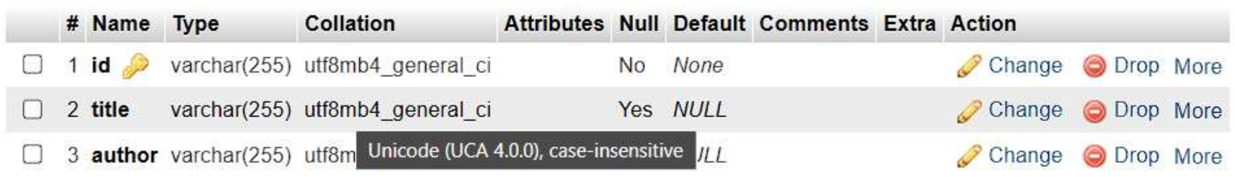


Table	Action							Rows	Type	Collation	Size	Overhead
book	★	Browse	Structure	Search	Insert	Empty	Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
member	★	Browse	Structure	Search	Insert	Empty	Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
membership	★	Browse	Structure	Search	Insert	Empty	Drop	4	InnoDB	utf8mb4_general_ci	16.0 KiB	-
3 tables	Sum							9	InnoDB	utf8mb4_general_ci	48.0 KiB	0 B

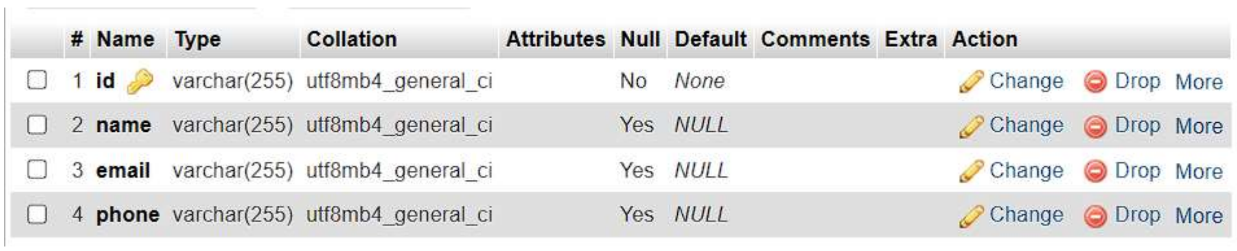
TABLES STRUCTURES

1.Book



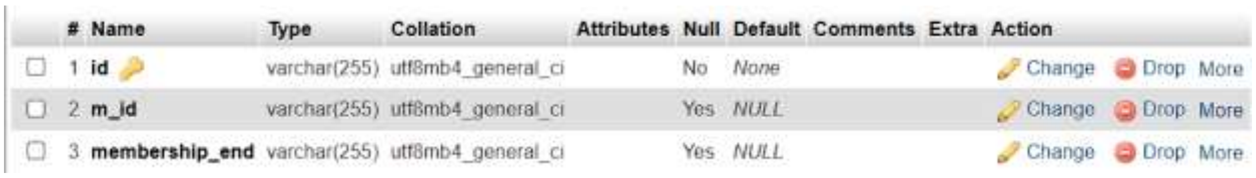
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
1	**id**	varchar(255)	utf8mb4_general_ci		No	*None*			Change	Drop	More
2	**title**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change	Drop	More
3	**author**	varchar(255)	utf8m Unicode (UCA 4.0.0), case-insensitive			*JLL*			Change	Drop	More

2.Member



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
1	**id**	varchar(255)	utf8mb4_general_ci		No	*None*			Change	Drop	More
2	**name**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change	Drop	More
3	**email**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change	Drop	More
4	**phone**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change	Drop	More

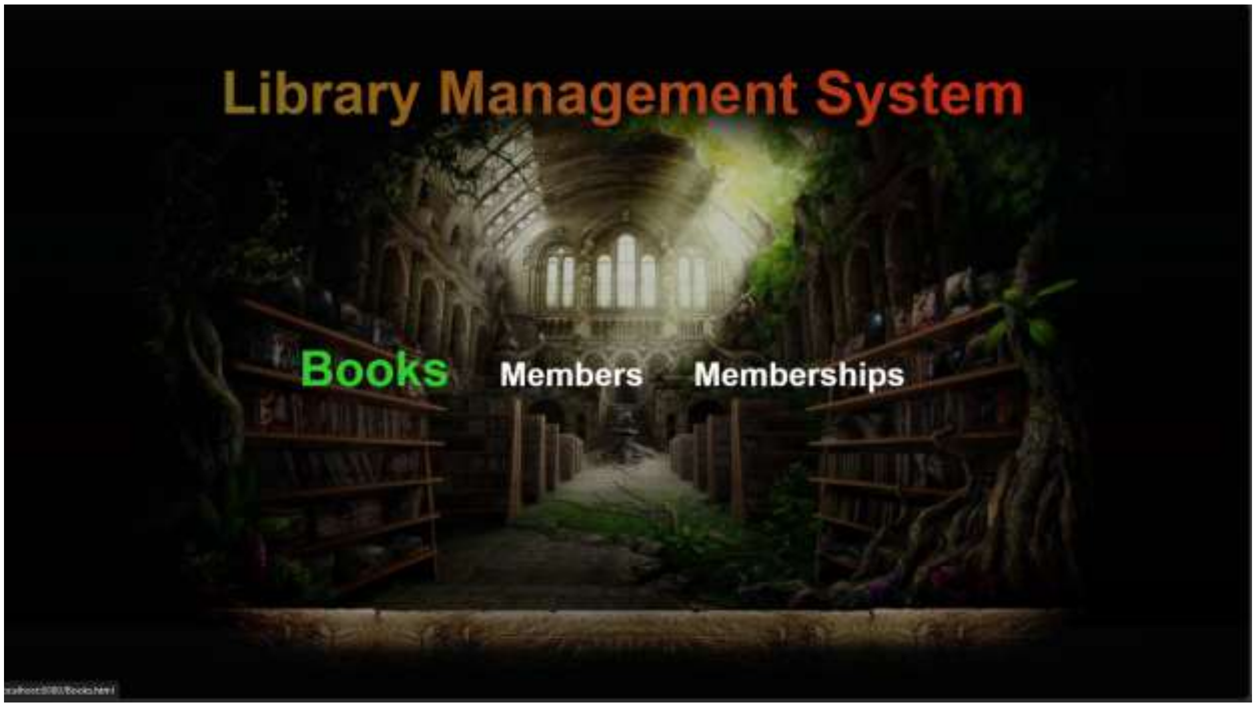
3.Membership



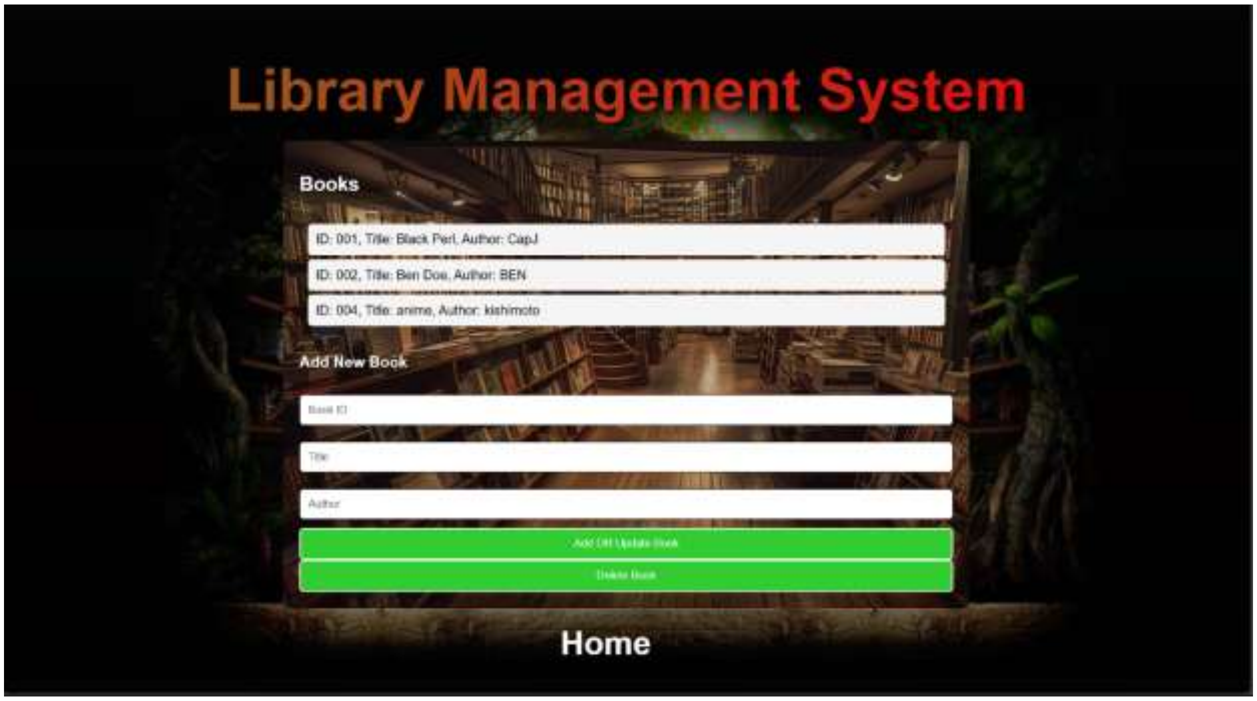
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action		
1	**id**	varchar(255)	utf8mb4_general_ci		No	*None*			Change	Drop	More
2	**m_id**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change	Drop	More
3	**membership_end**	varchar(255)	utf8mb4_general_ci		Yes	*NULL*			Change	Drop	More

FRONT END INTERFACES

1.Landing page (Named as : Index.html)

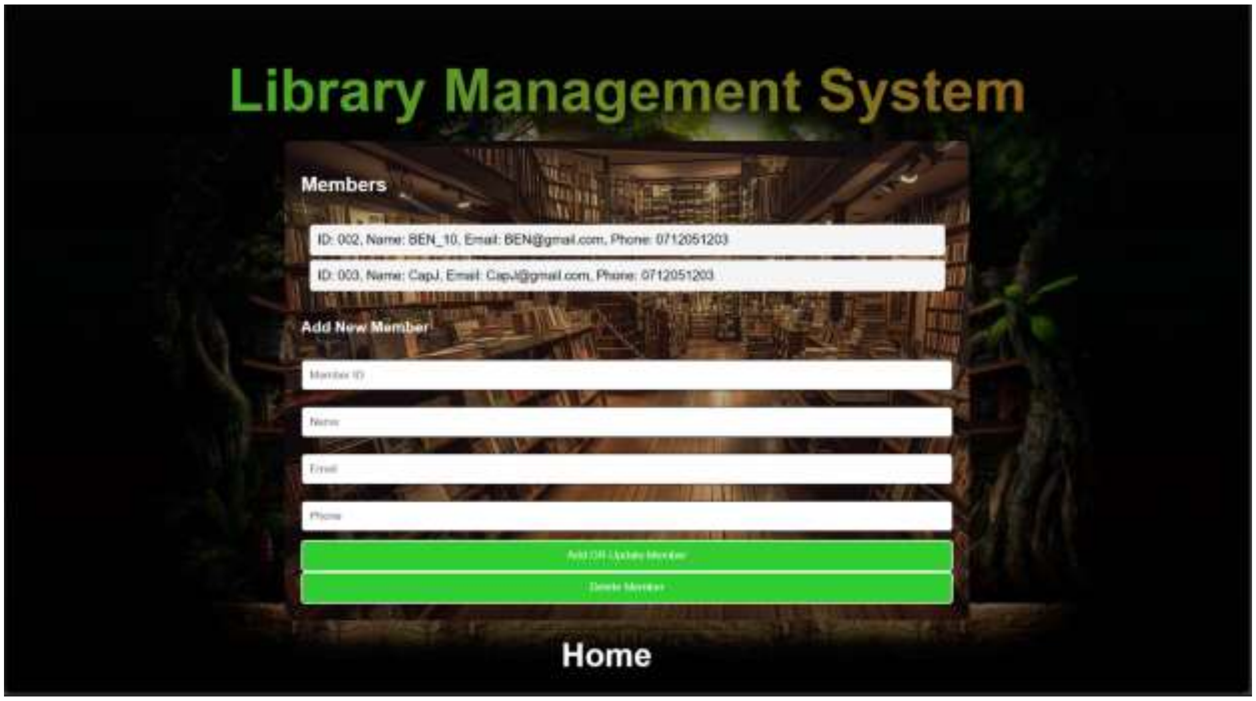


2.Book Page (Named as : Book.html)



ID: 001, Title: Black Perl, Author: CapJ
ID: 002, Title: Ben Doe, Author: BEN
ID: 004, Title: anime, Author: kishimoto

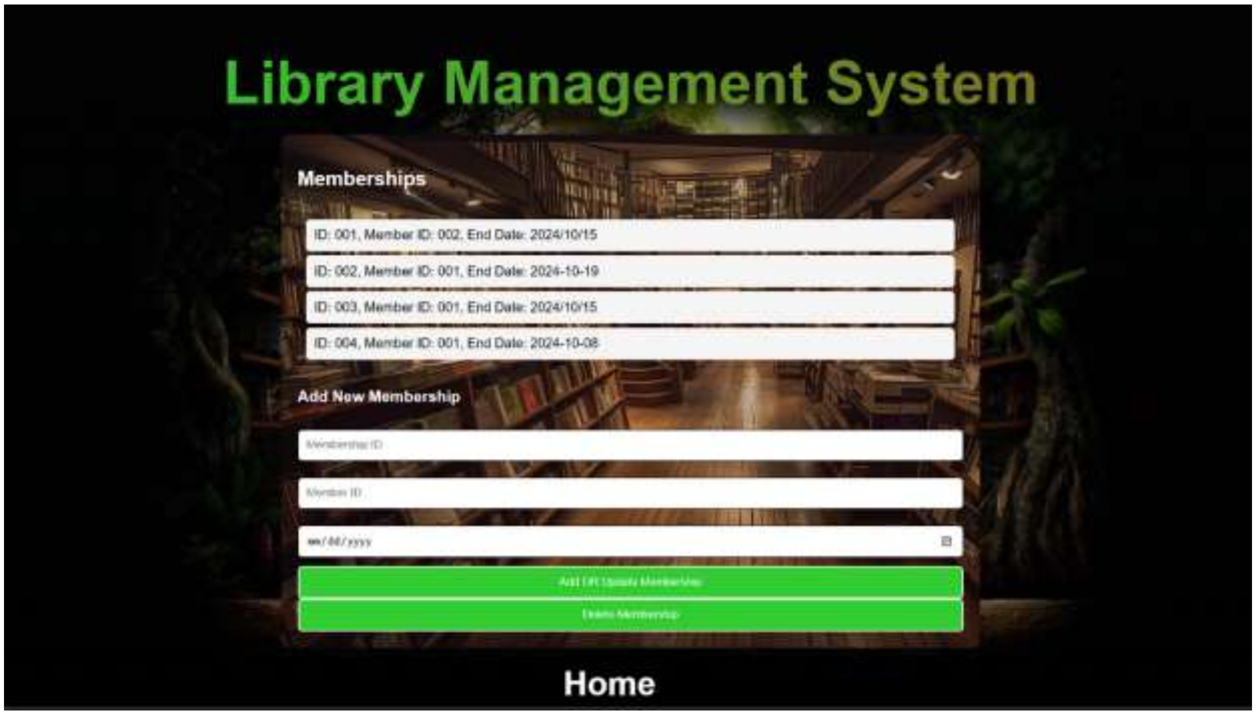
3.Member Page (Named as :Member.html)



ID: 002, Name: BEN_10, Email: BEN@gmail.com, Phone: 0712051203

ID: 003, Name: CapJ, Email: CapJ@gmail.com, Phone: 0712051203

4.Membership Page (Named as :Membership.html)



ID: 001, Member ID: 002, End Date: 2024/10/15

ID: 002, Member ID: 001, End Date: 2024-10-19

ID: 003, Member ID: 001, End Date: 2024/10/15

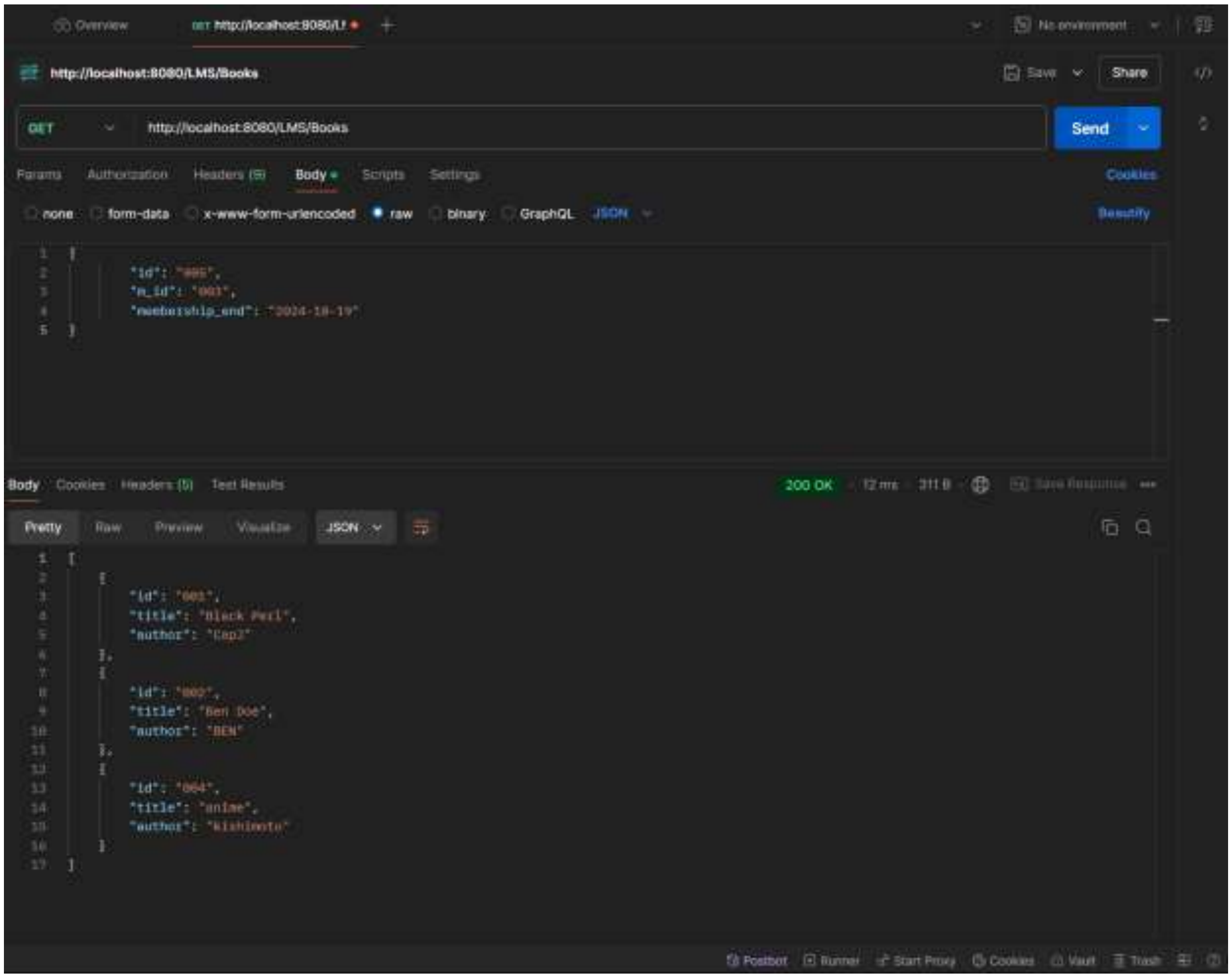
ID: 004, Member ID: 001, End Date: 2024-10-08

TEST CASES AND RESULTS

Testing manually using POST-Man Application.

TEST CASE : Using GET method retrieve all the books.

TEST RESULT :



http://localhost:8080/LMS/Books

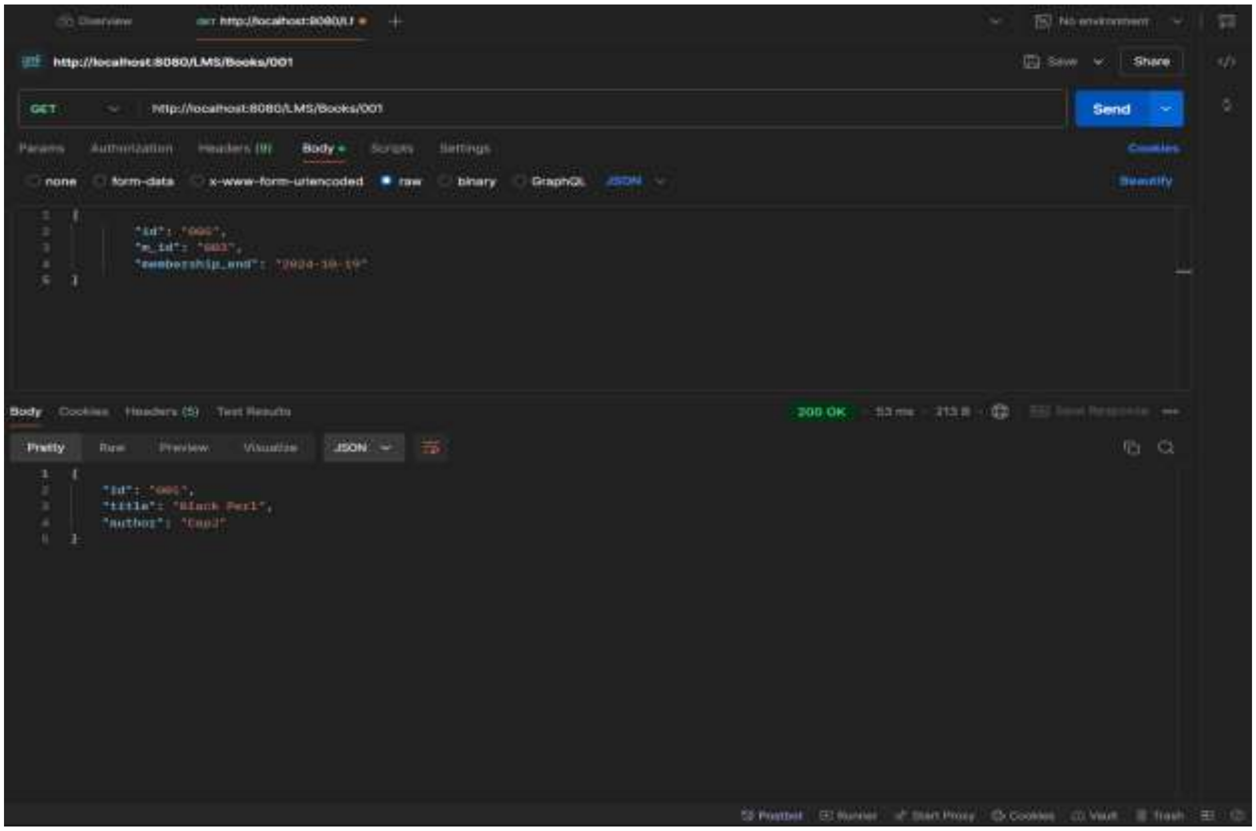
GET http://localhost:8080/LMS/Books

```
{
    "id": "005",
    "m_id": "003",
    "membership_end": "2024-10-19"
}
```

```
[
    {
        "id": "001",
        "title": "Black Perl",
        "author": "CapJ"
    },
    {
        "id": "002",
        "title": "Ben Doe",
        "author": "BEN"
    },
    {
        "id": "004",
        "title": "anime",
        "author": "kishimoto"
    }
]
```

TEST CASE : Using GET method retrieve 001 ID Book

TEST RESULT :



GET http://localhost:8080/LMS/Books/001

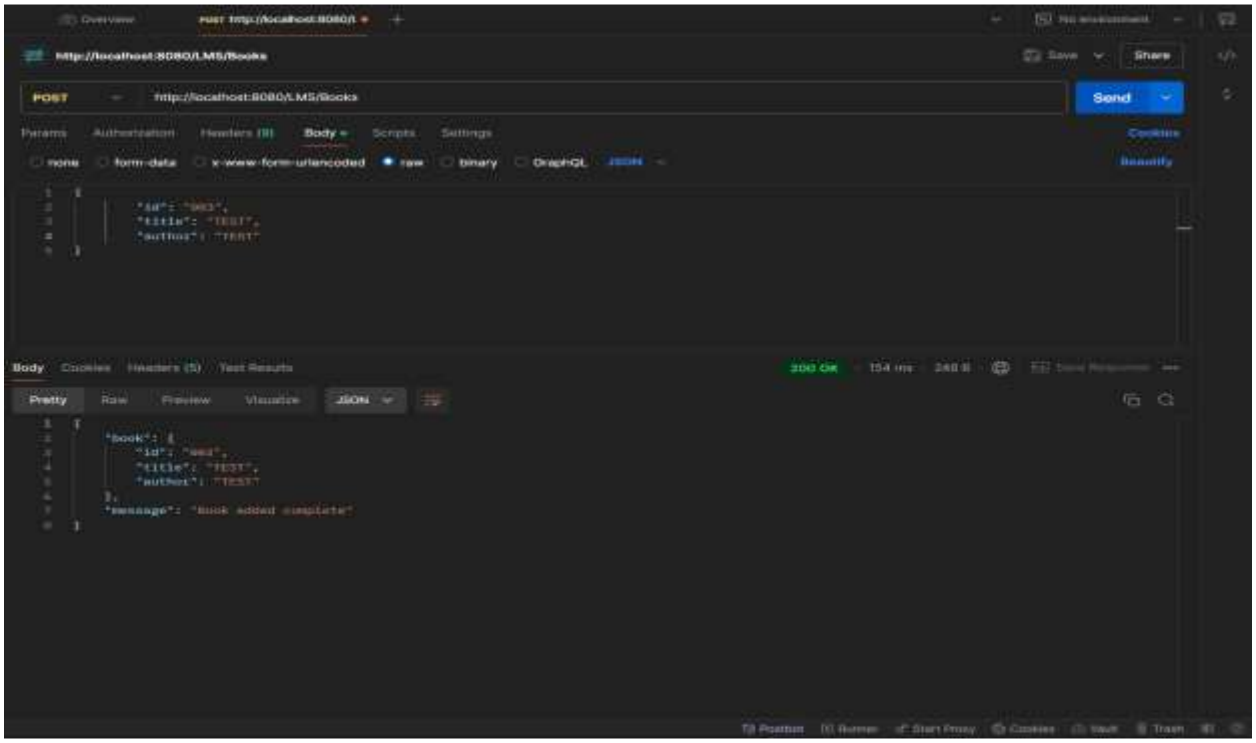
```
{
    "id": "006",
    "m_id": "003",
    "membership_end": "2024-10-19"
}
```

200 OK · 53 ms · 213 B

```
{
  "id": "001",
  "title": "Black Perl",
  "author": "CapJ"
}
```

TEST CASE : Using POST method Inserting a Book

TEST RESULT :



POST http://localhost:8080/LMS/Books

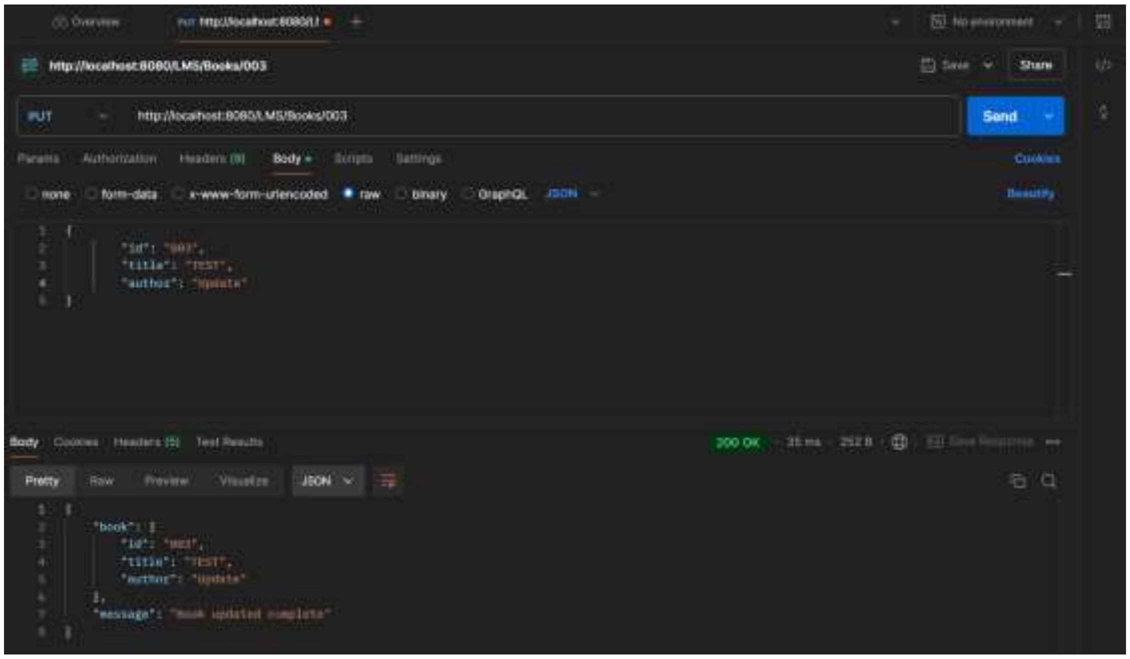
```
{
    "id": "003",
    "title": "TEST",
    "author": "TEST"
}
```

200 OK · 154 ms · 248 B

```
{
  "book": {
    "id": "003",
    "title": "TEST",
    "author": "TEST"
  },
  "message": "Book added complete"
}
```

TEST CASE : Using PUT method Updating a Book

TEST RESULT :



PUT http://localhost:8080/LMS/Books/003

Body: raw, JSON

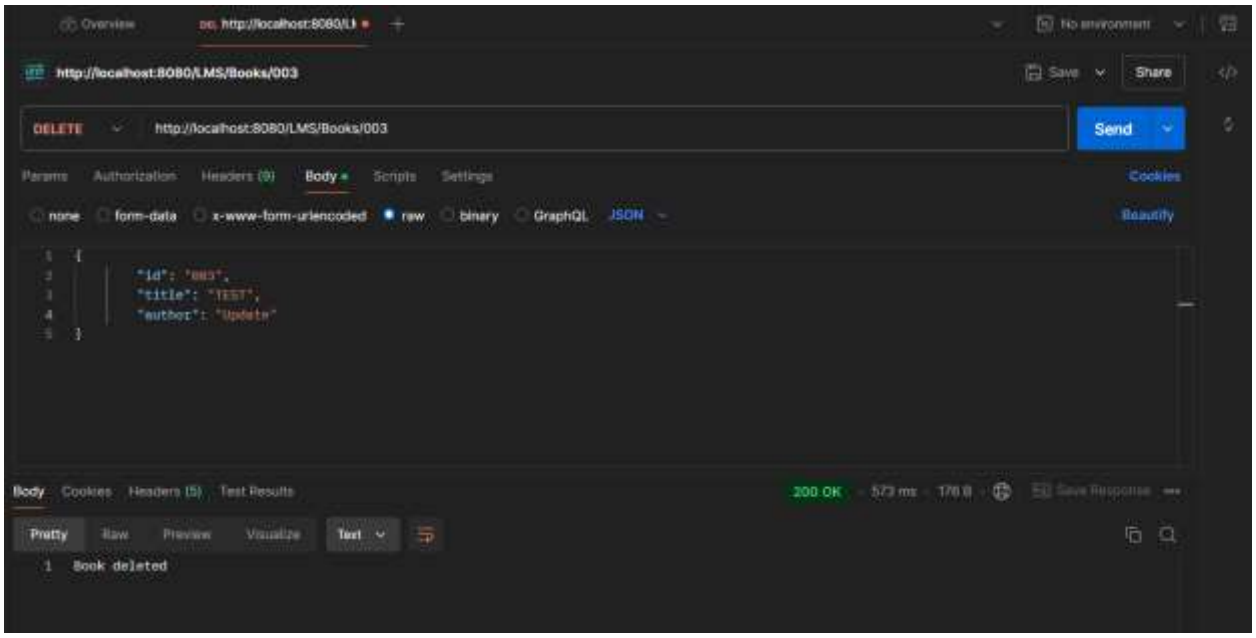
```
{
    "id": "003",
    "title": "TEST",
    "author": "Update"
}
```

200 OK · 35 ms · 252 B

```
{
  "book": {
    "id": "003",
    "title": "TEST",
    "author": "Update"
  },
  "message": "Book updated complete"
}
```

TEST CASE : Using DELETE method Deleting a Book

TEST RESULT :



DELETE http://localhost:8080/LMS/Books/003

Body: raw, JSON

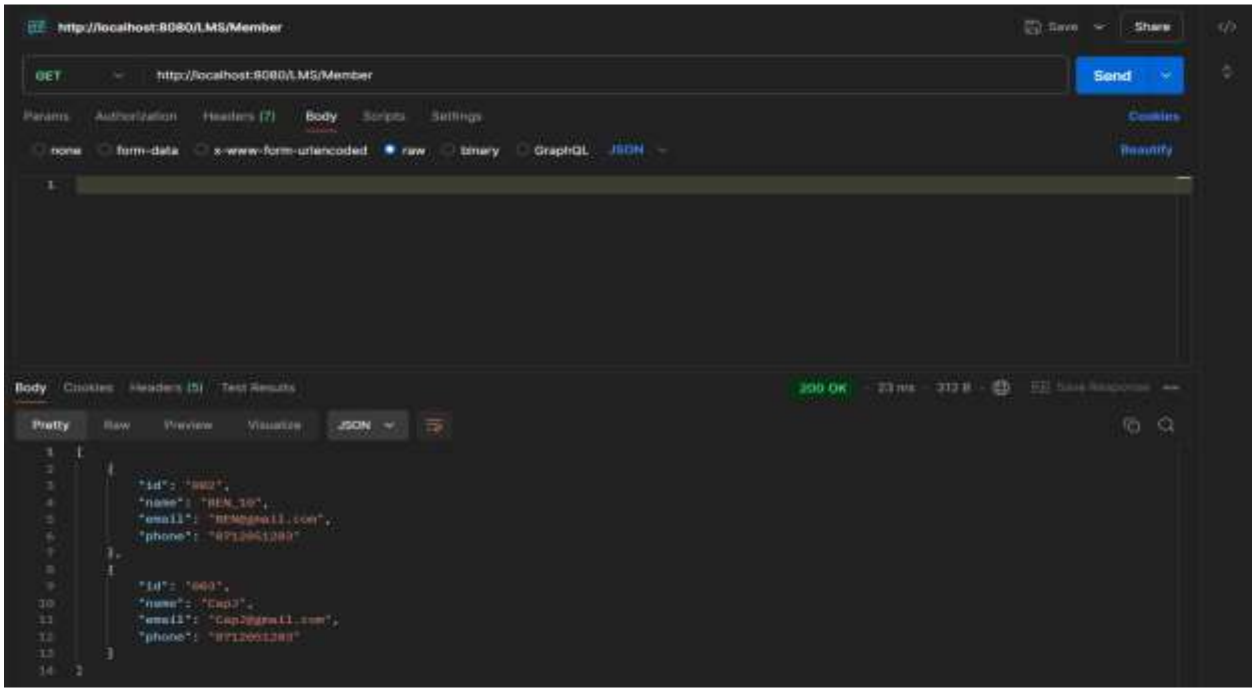
```
{
    "id": "003",
    "title": "TEST",
    "author": "Update"
}
```

200 OK · 573 ms · 176 B

```
Book deleted
```

TEST CASE : Using GET method retrieve all the Member.

TEST RESULT :



GET http://localhost:8080/LMS/Member

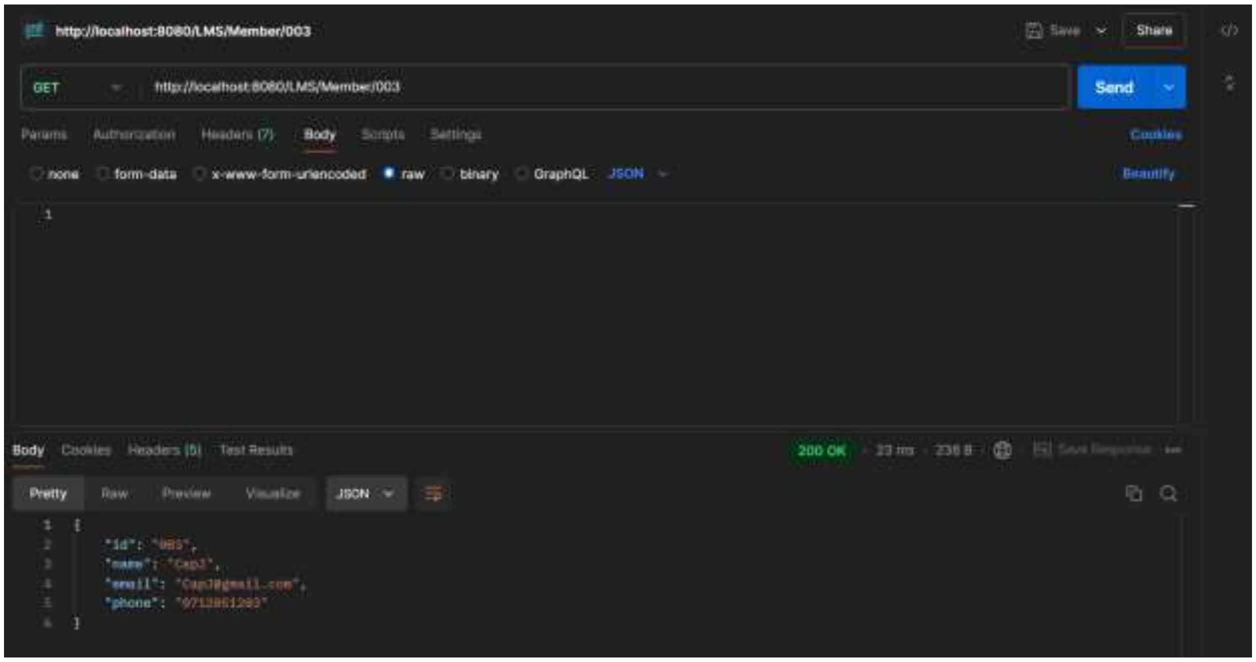
Body: raw, JSON

200 OK · 23 ms · 312 B

```
[
    {
        "id": "002",
        "name": "BEN_10",
        "email": "BEN@gmail.com",
        "phone": "0712061283"
    },
    {
        "id": "003",
        "name": "Cap3",
        "email": "Cap3@gmail.com",
        "phone": "0712061283"
    }
]
```

TEST CASE : Using GET method retrieve a single Member.

TEST RESULT :



GET http://localhost:8080/LMS/Member/003

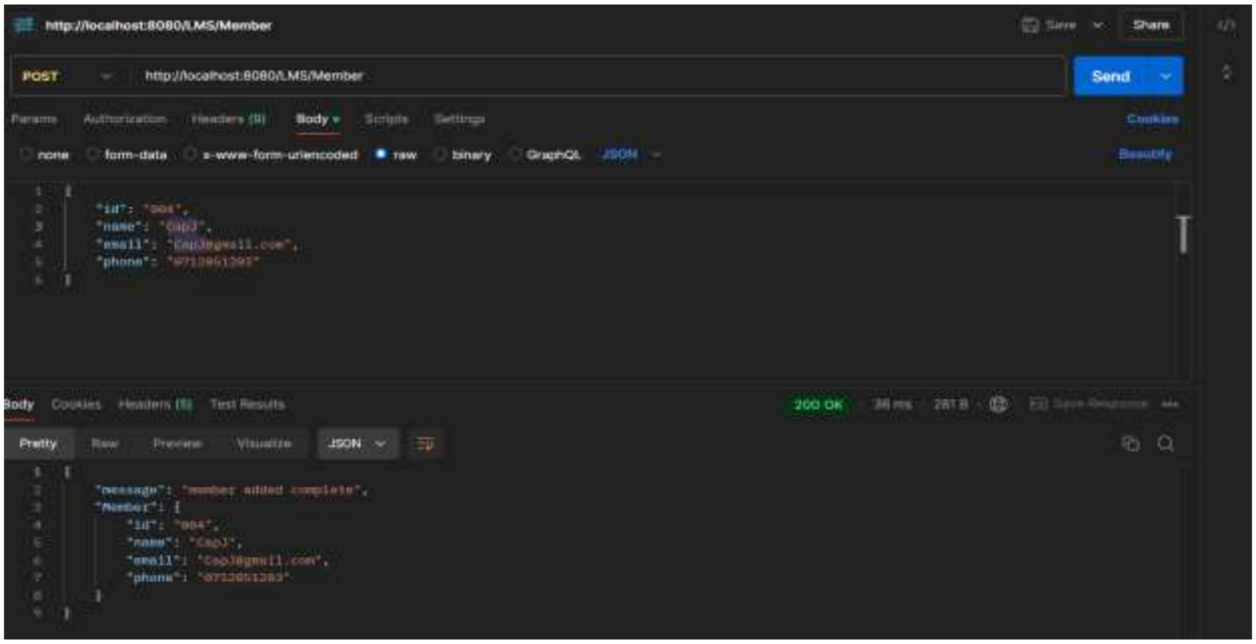
Body: raw, JSON

200 OK · 23 ms · 238 B

```
{
    "id": "003",
    "name": "Cap3",
    "email": "Cap3@gmail.com",
    "phone": "0712061283"
}
```

TEST CASE : Using POST method insert single Member.

TEST RESULT :



http://localhost:8080/LMS/Member

POST http://localhost:8080/LMS/Member

Body: raw JSON

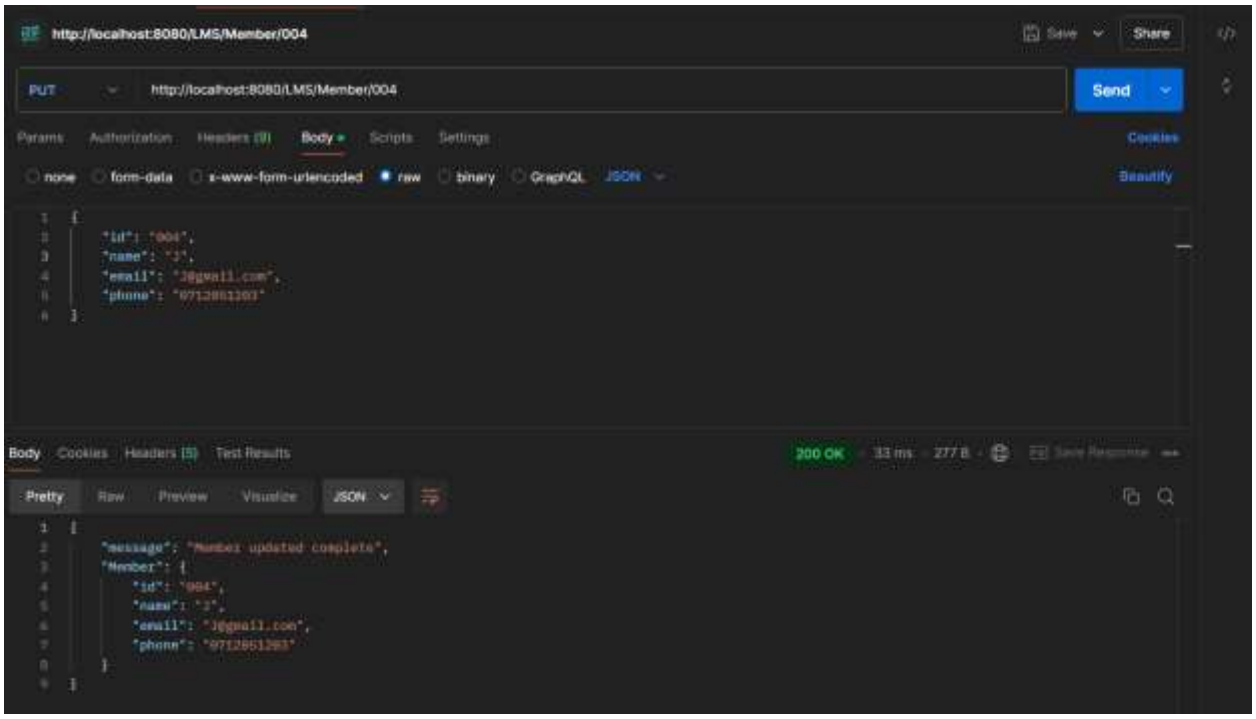
```
{
    "id": "004",
    "name": "CapJ",
    "email": "CapJ@gmail.com",
    "phone": "0712851203"
}
```

200 OK · 36 ms · 281 B

```
{
    "message": "member added complete",
    "Member": {
        "id": "004",
        "name": "CapJ",
        "email": "CapJ@gmail.com",
        "phone": "0712851203"
    }
}
```

TEST CASE : Using PUT method updating single Member.

TEST RESULT :



http://localhost:8080/LMS/Member/004

PUT http://localhost:8080/LMS/Member/004

Body: raw JSON

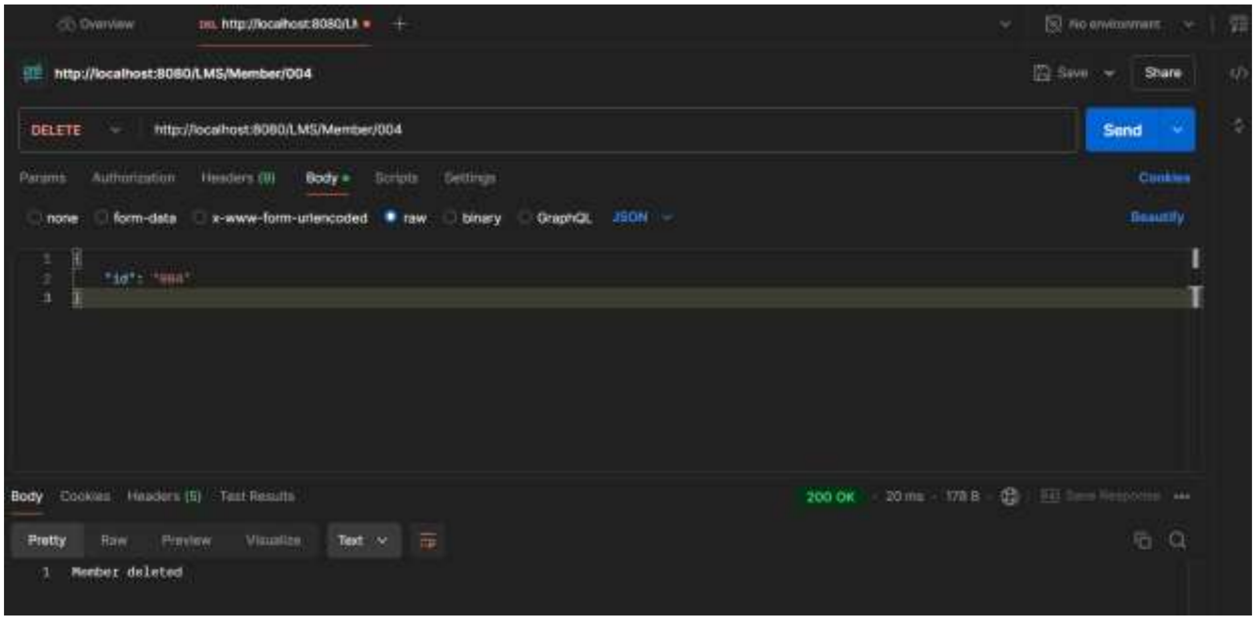
```
{
    "id": "004",
    "name": "J",
    "email": "J@gmail.com",
    "phone": "0712851203"
}
```

200 OK · 33 ms · 277 B

```
{
    "message": "Member updated complete",
    "Member": {
        "id": "004",
        "name": "J",
        "email": "J@gmail.com",
        "phone": "0712851203"
    }
}
```

TEST CASE : Using DELETE method deleting a single Member.

TEST RESULT :



DELETE http://localhost:8080/LMS/Member/004

Body: raw, JSON

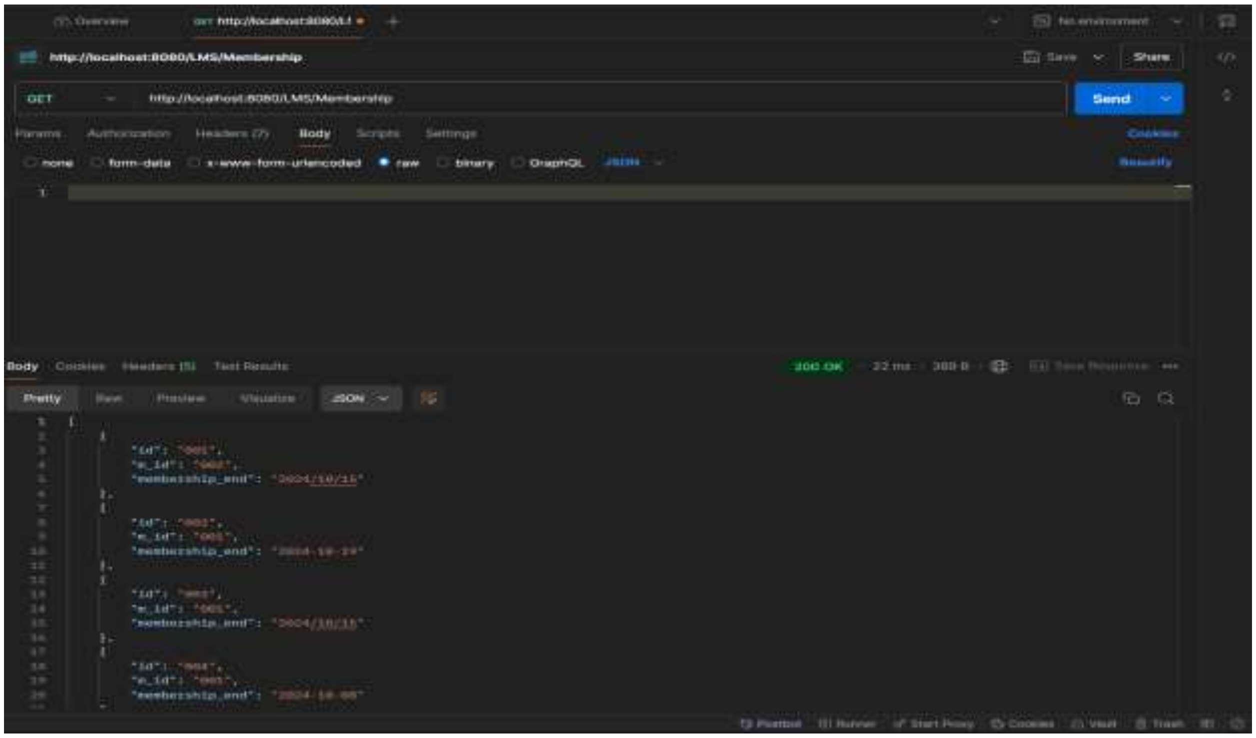
```
{
    "id": "004"
}
```

200 OK · 20 ms · 178 B

```
Member deleted
```

TEST CASE : Using GET method select all Membership.

TEST RESULT :



GET http://localhost:8080/LMS/Membership

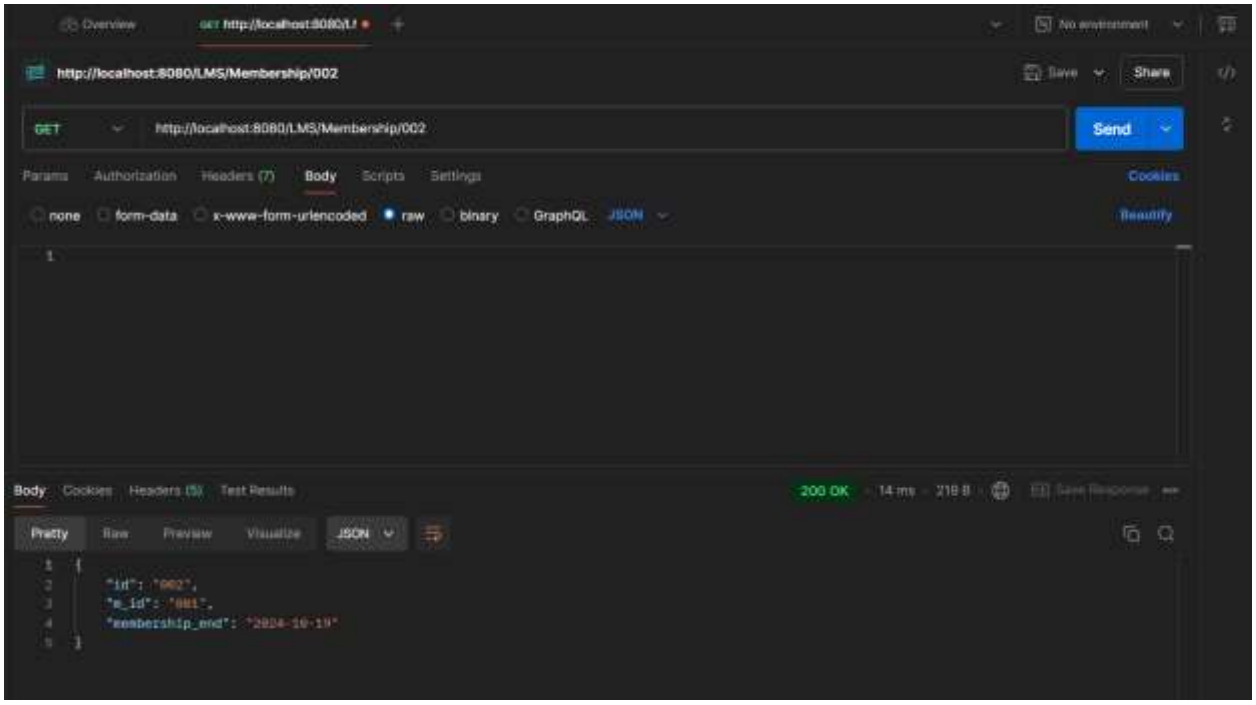
Body: raw, JSON

200 OK · 22 ms · 388 B

```
[
    {
        "id": "001",
        "m_id": "002",
        "membership_end": "2024/10/15"
    },
    {
        "id": "002",
        "m_id": "001",
        "membership_end": "2024-10-29"
    },
    {
        "id": "003",
        "m_id": "001",
        "membership_end": "2024/10/15"
    },
    {
        "id": "004",
        "m_id": "001",
        "membership_end": "2024-10-08"
    -
```

TEST CASE : Using GET method select a single Membership.

TEST RESULT :



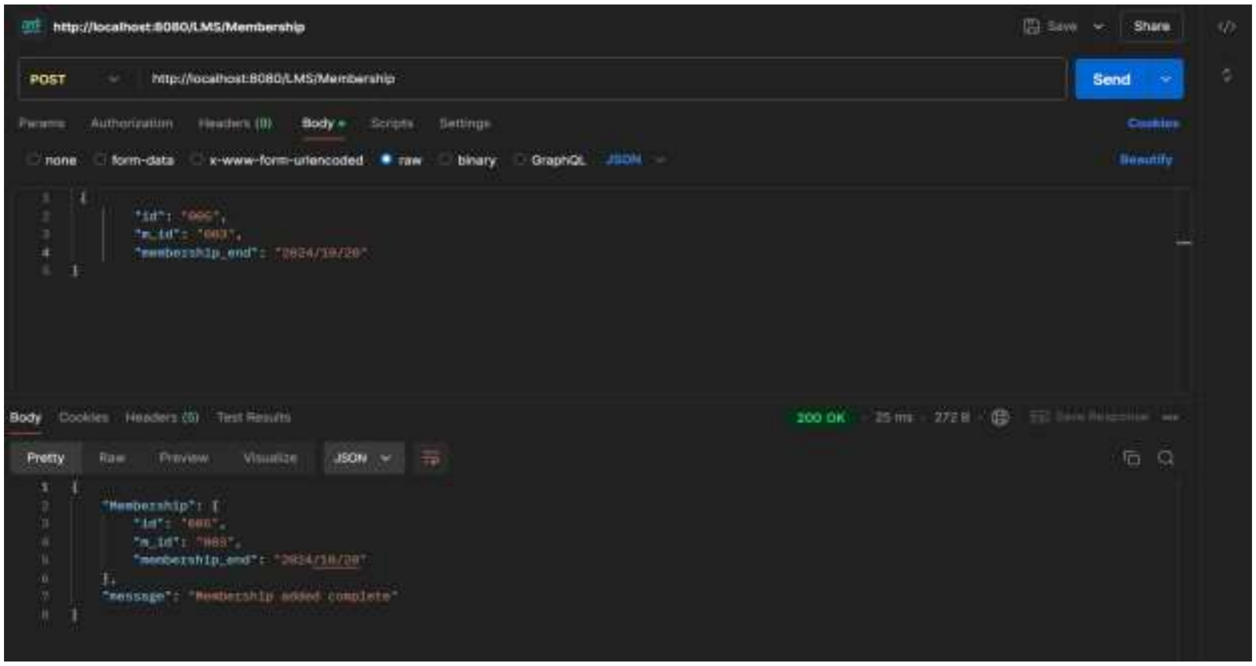
GET http://localhost:8080/LMS/Membership/002

200 OK · 14 ms · 218 B

```
{
    "id": "002",
    "m_id": "001",
    "membership_end": "2024-10-19"
}
```

TEST CASE : Using POST method insert a Membership.

TEST RESULT :



POST http://localhost:8080/LMS/Membership

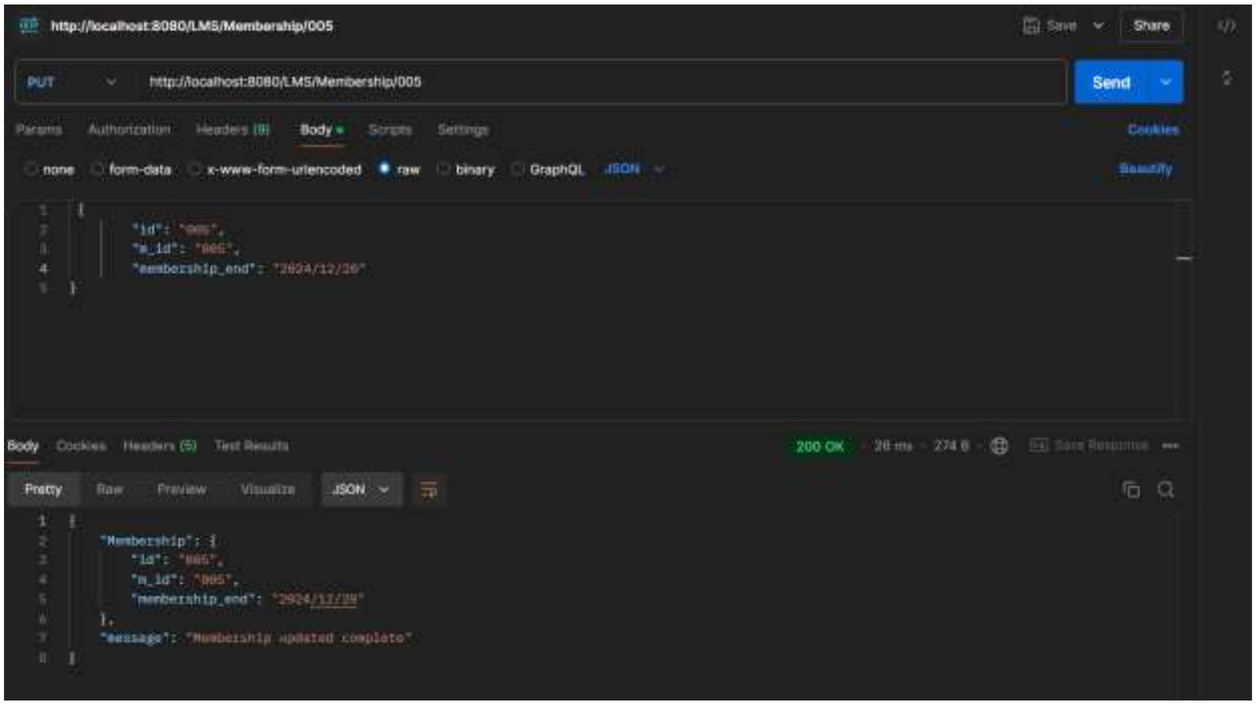
```
{
    "id": "006",
    "m_id": "003",
    "membership_end": "2024/10/20"
}
```

200 OK · 25 ms · 272 B

```
{
    "Membership": {
        "id": "006",
        "m_id": "003",
        "membership_end": "2024/10/20"
    },
    "message": "Membership added complete"
}
```

TEST CASE : Using PUT method updating a Membership.

TEST RESULT :


PUT http://localhost:8080/LMS/Membership/005

Body: raw JSON

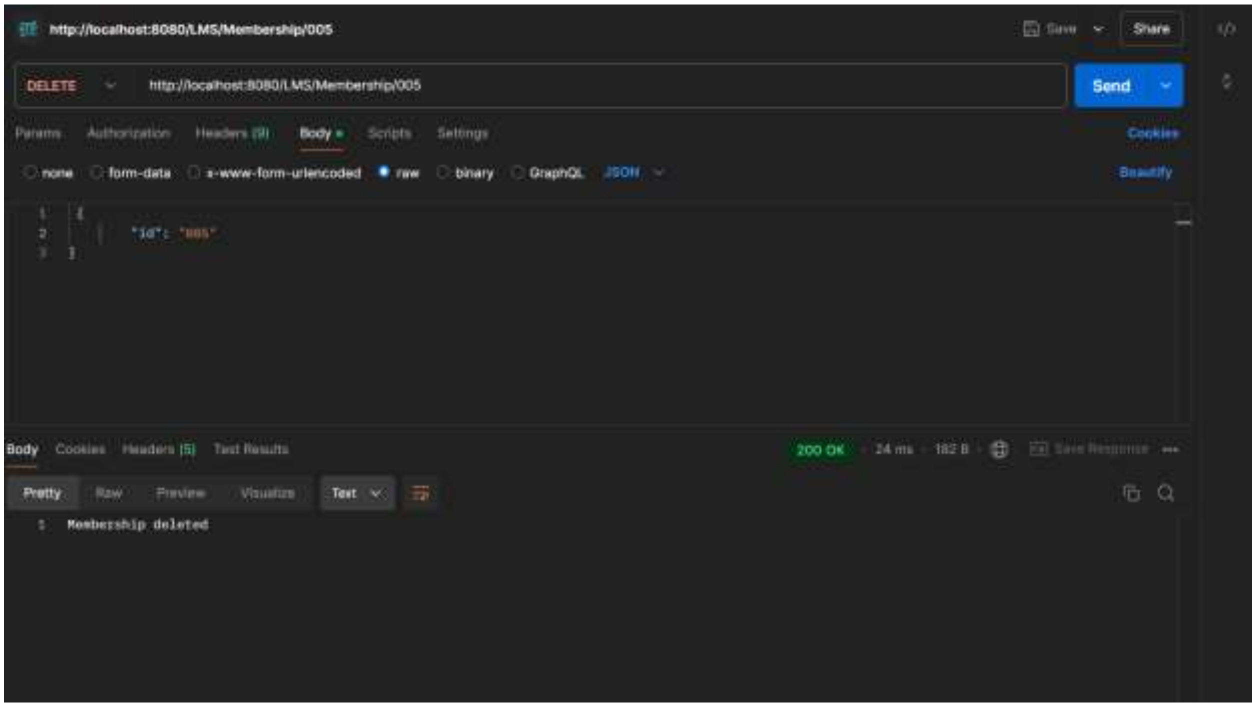
```
{
    "id": "005",
    "m_id": "005",
    "membership_end": "2024/12/26"
}
```

200 OK · 26 ms · 274 B

```
{
  "Membership": {
      "id": "005",
      "m_id": "005",
      "membership_end": "2024/12/28"
  },
  "message": "Membership updated complete"
}
```

TEST CASE : Using DELETE method Deleting a Membership.

TEST RESULT :


DELETE http://localhost:8080/LMS/Membership/005

Body: raw JSON

```
{
    "id": "005"
}
```

200 OK · 24 ms · 182 B

```
Membership deleted
```

REFERENCES

(n.d.). Retrieved from GeeksforGeeks: https://www.geeksforgeeks.org/exception-handling-in-spring-boot/

(n.d.). Retrieved from W3School: https://www.w3schools.com/java/java_try_catch.asp

(n.d.). Retrieved from StackOverflow: https://stackoverflow.com/questions/66762006/spring-boot-exception-handling-best-practice

baeldung. (n.d.). Retrieved from https://www.baeldung.com/spring-boot-bean-validation

spring. (n.d.). Retrieved from https://spring.io/guides/gs/rest-service

Tutorialpoints. (n.d.). Retrieved from https://www.tutorialspoint.com/spring_boot/spring_boot_exception_handling.htm

PROJECT GIT LINK

https://github.com/Dhanushanandan/LMS_EAD2_CW.git

README.md file add to the Git