Group 22

Name: Hitihami Bandara	
Student Reference Number: 10819538	

Module Code: PUSL3120	Module Name: Full-Stack Development
Coursework Title: Group Project Report For Library System	
Deadline Date:08/01/2023	Member of staff responsible for coursework: Dr Mark Dixon
Programme: Bsc (Hons) Software Engineering	

Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.

Name	ID	Task
Hitihami Bandara	10819538	Involve in both report and implementation (User login)
Kasthuri A Arachchi	10819557	Involve in both report and implementation (Category)
N.A.I.I Sandakalum	10820824	Involve in both report and implementation (Admin login)
Vithanage Bandara	10818157	Involve in both report and implementation (Transaction)
Akila E Kalupahana	10820798	Involve in both report and implementation (Book)

We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.

Signed on behalf of the group:viraj

Individual assignment: ***I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offences and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.***

Signed :

Group 22

Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.

I ~~*have used~~/not used translation software.

If used, please state name of software………………………………………………………………

Overall mark _____% **Assessors Initials ______** **Date_________**

*Please delete as appropriate Sci/ps/d:/students/cwkfrontcover/2013/14

GitHub Link –

https://github.com/DevIrosh/Library_MSystem.git

YouTube Link -

https://youtu.be/5uMo3Sewdfo

Introduction

Libraries can help people to get knowledge and inspiration. Think about a library where users able to find their preferred books easily. But managing a library is a difficult task without any technological help. The aim of this project is to develop a dynamic web application which will be act as a comprehensive library management system. Librarians and library users can use this system to do their work with a good user experience. So, we have designed a brand-new library system powered by React, Node, Express and MongoDB technologies.

Requirements

The target users:

- Librarians
  This system consists of a librarian to manage the library operations like managing books and managing library members.

- Borrowers or Library members
  This system consists of borrowers which facilitate the searching books, viewing the borrowing history, reserve books and viewing the transactions.

- Administrator or Admin
  This system consists of an Administrator to make analytical decisions, to view reports and to manage the library users.

Features:

- Library member management.
  This feature allows the users to register new borrowers. Allow to update the member information, allow to view the borrower’s profile, and remove borrowers.

- Book management.
  This feature manages the books. Able to add, update, view and delete book details like ISBN, title, author, category, and book availability status.

- Transaction management.
  This feature use to track and display the transaction history of the books. This includes burrowed and return details. Real-time transactions recorded were used. Users will be able to view the current availability of a book.

- Category management.

This feature was used to classify the books. This classification based on an category id, category and description. Allow librarian to organize the books and locate them easily by groping the books on relevant shelves. Make a clear understanding to the librarian the content and theme where a book belongs.

- Librarian management.
  This feature help to manage the library members accounts. Manage account creation and deletion. Mange books by adding updating and deleting book data.

- Administrative management.
  This feature used to manage the user accounts and librarian accounts. Involve in reviewing the reports of transactions history and book availability.

- User authentication.
  Used to create a secure login system. Some features were restricted based on the user.

Design

System Architecture.

To develop this system client-server architecture a three-tier architecture was used.

- Client-server architecture

The client-side was designed to build using HTML, CSS, and JavaScript technologies. This help to develop interactive user interfaces. This is the front-end developed using react framework which runs on the user's web browsers.

The server-side was designed to build using Node.js and Express technologies. This help to interact with the database, manage user requests and to implement the business logic. Also handle the API requests.

Database was designed to build using Mongo DB. This help to handle the data by storing and retrieving the required book, members, transactions, and other data.

- Component interaction

1. Interaction of the user

By using the browser action, the Users interact with this react application.

2. Calling API

From the client-side the browser sends HTTP requests to the Express server for the data retrieval and data actions like processing and filtering required data.

3. The server-side logic.

To handle the API requests Express routes have been used. It interact with the MongoDB to access the data and then send the required response back to the browser.

4. Fetching the data
   The browser (react components) update or change its content or the UI based on the received data from the server.

- Structure of the Data

A MongoDB collection was used to store the data as follows:

Entity	**Attributes**
User	User ID Username Password Email Role (Borrower or Librarian) Address Phone number
Book	Book name Category Status (Available state of a book)
Librarian/ Admin	Username Password Email Address Phone number
Transaction	Transaction ID User ID Name Borrow Date Return Date Status
Category	Category ID

	Name Description

- Structure of the code:

The code mainly consists of 2 parts as front-end using React and back-end using Node.js and Express.

In the development of front-end components has been used as they are UI elements with own state and strategies that can be reuse. For the handling of the navigation and rendering the content routes have been used. The services like data management with encapsulated API interactions were used.

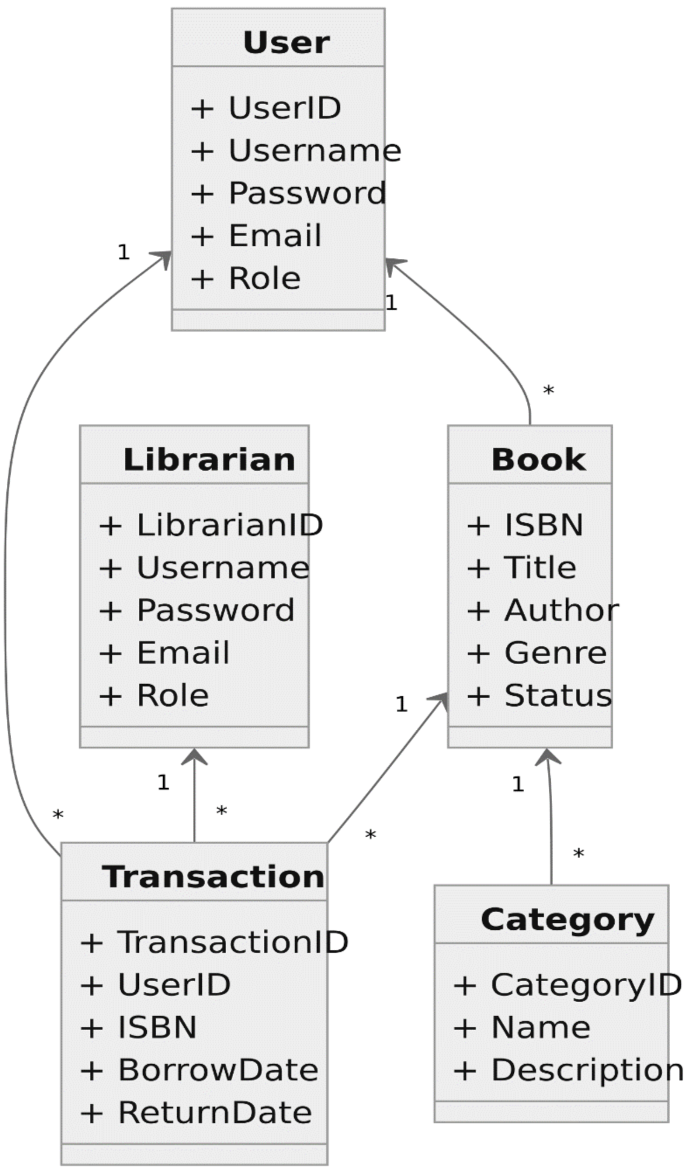
In the development of back-end routes have been used for mapping API endpoints to the controllers. It uses controllers to engage on request handling, interact with the databased and sending responses. To represent the data structure and the interaction with the MongoDB models have been used.

When consider about the appropriateness of the above structures we used those structures due to the following reasons.

Use MERN stack because it aligns with the best practices with modern web development. MongoDB was used because it is flexible and can accommodate different data types and relationships. Using of React Component-Based architecture help the reusability and maintainability of the code. To streamline the routing and API development easily Express Framework was used. To handle concurrent requests and scaling in an effective manner Node.js was used.

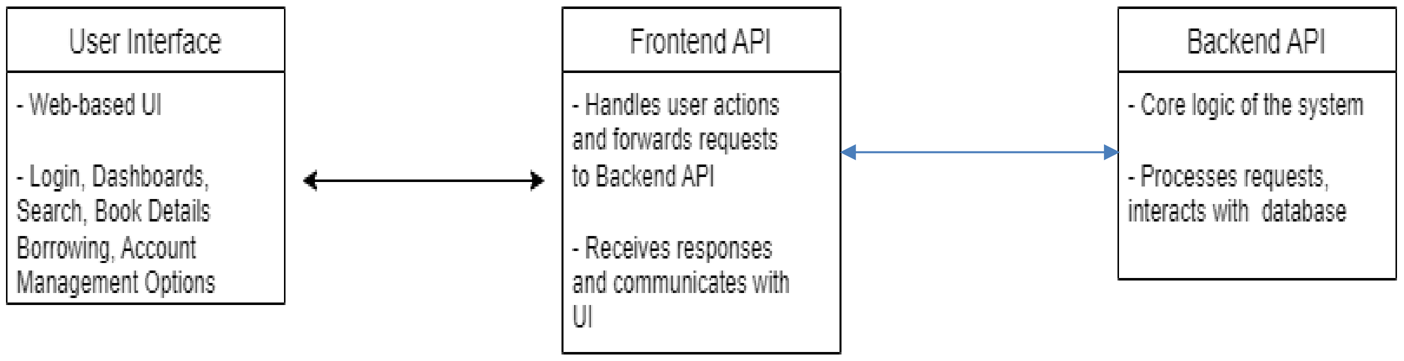
- **UML diagrams Structural Diagrams**

1. Class Diagram

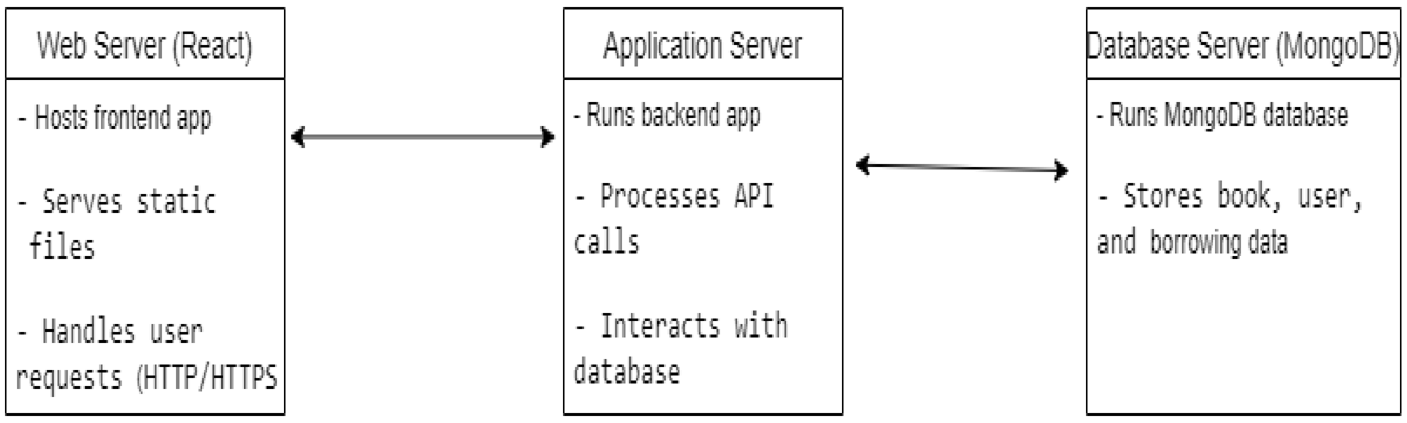


This is the class diagram of the system.

2. Component Diagram

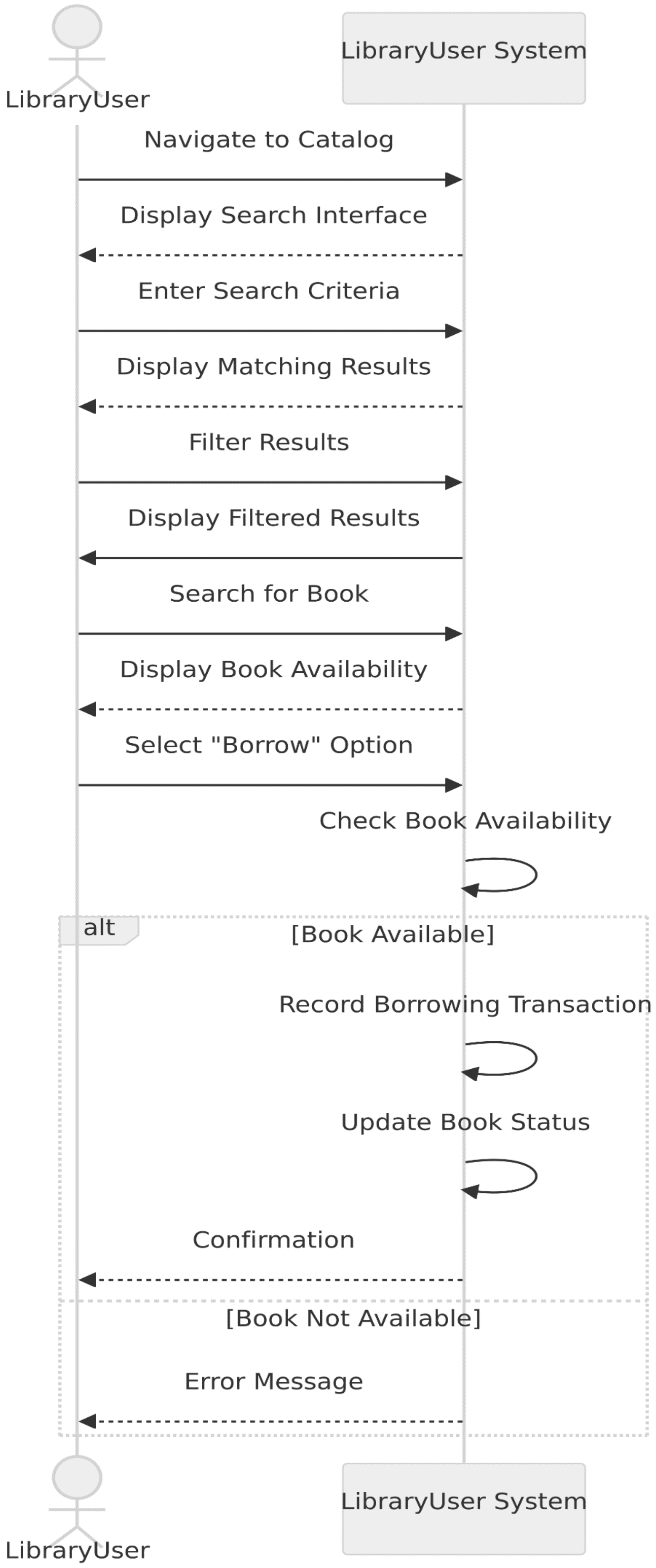

User Interface
- Web-based UI
- Login, Dashboards, Search, Book Details Borrowing, Account Management Options
Frontend API
- Handles user actions and forwards requests to Backend API
- Receives responses and communicates with UI
Backend API
- Core logic of the system
- Processes requests, interacts with database

3. Deployment Diagram



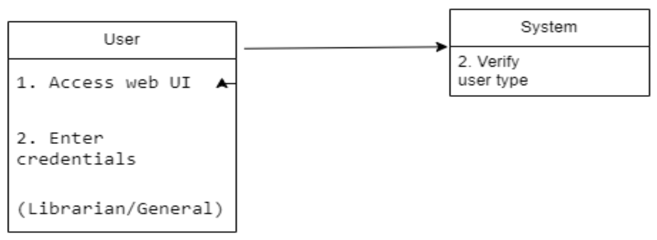
- **UML diagrams Behavioural Diagrams**

4. Use case Diagram.

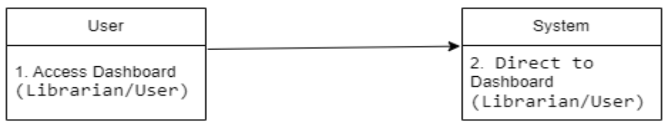


5. Activity Diagram

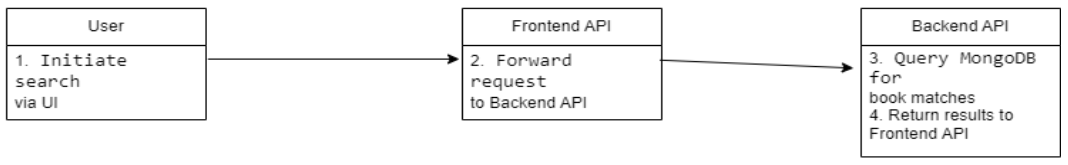
1. User Login:



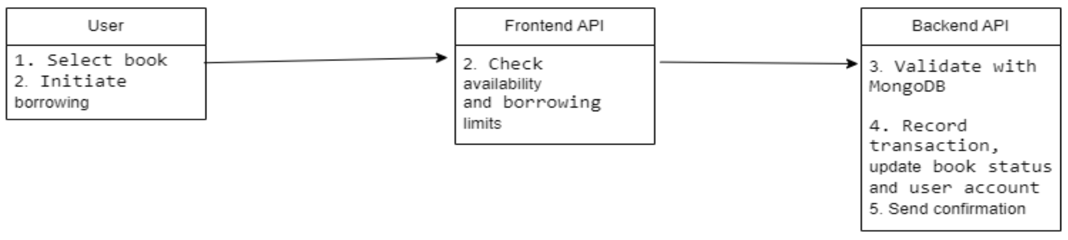
2. Dashboard Access:


User
1. Access Dashboard (Librarian/User)
System
2. Direct to Dashboard (Librarian/User)

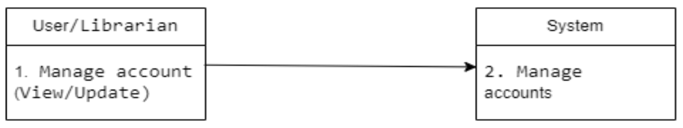
3. Book Search:


User
1. Initiate search via UI
Frontend API
2. Forward request to Backend API
Backend API
3. Query MongoDB for book matches
4. Return results to Frontend API

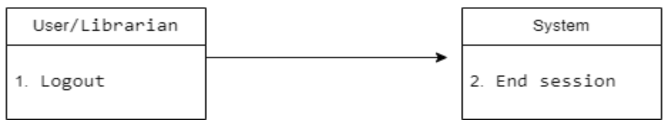
4. Book Selection and Borrowing:



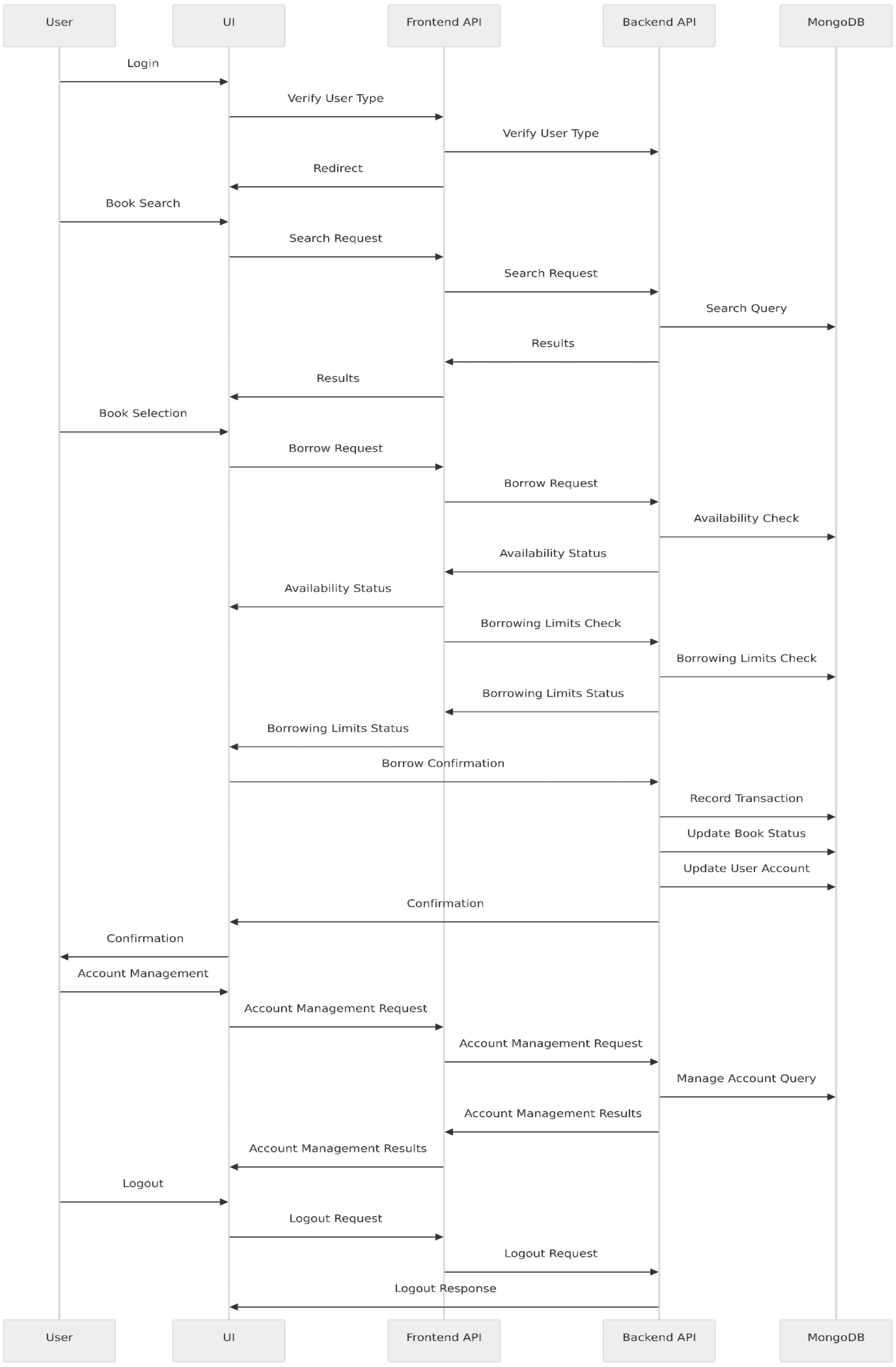
5. Account Management:



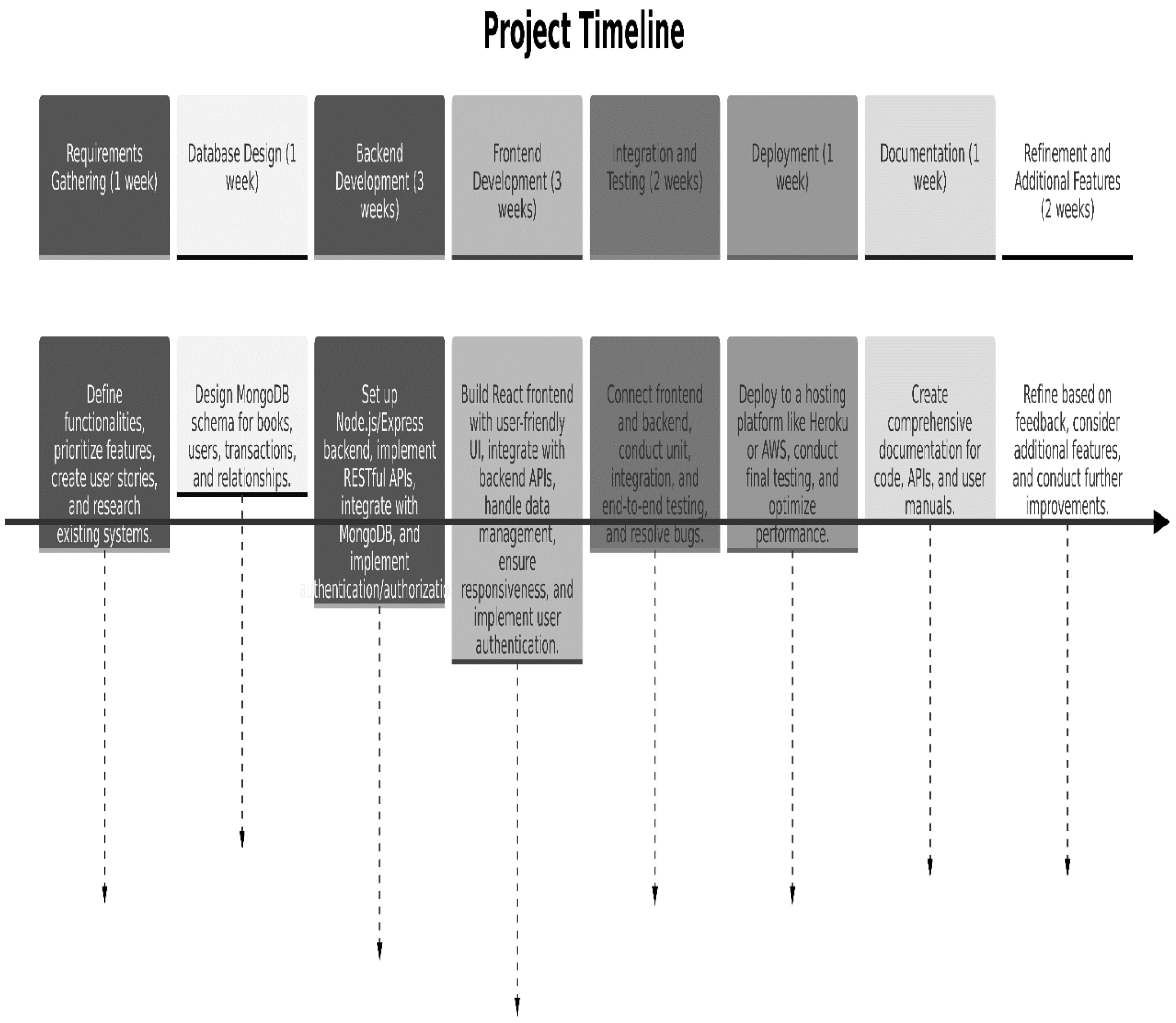
6. Logout:



6. Sequnce diagram



Timeline Diagram


Project Timeline
Requirements Gathering (1 week)
Database Design (1 week)
Backend Development (3 weeks)
Frontend Development (3 weeks)
Integration and Testing (2 weeks)
Deployment (1 week)
Documentation (1 week)
Refinement and Additional Features (2 weeks)
Define functionalities, prioritize features, create user stories, and research existing systems.
Design MongoDB schema for books, users, transactions, and relationships.
Set up Node.js/Express backend, implement RESTful APIs, integrate with MongoDB, and implement authentication/authorizatio
Build React frontend with user-friendly UI, integrate with backend APIs, handle data management, ensure responsiveness, and implement user authentication.
Connect frontend and backend, conduct unit, integration, and end-to-end testing, and resolve bugs.
Deploy to a hosting platform like Heroku or AWS, conduct final testing, and optimize performance.
Create comprehensive documentation for code, APIs, and user manuals.
Refine based on feedback, consider additional features, and conduct further improvements.

.

Implementation

To implement this system, we have used the MERN stack in the full stack development. React framework was used for the client-side rendering. For the server-side routing and API interactions Node.js Express frameworks was used. For robust and scalable data storage MongoDB was used.

For the security measures, the passwords were bult using hashing bycrypt. To manage the site, we have used jwt authentication. This JWT used to display login logout of their user. Used to manage users.

Decide to use WebSocket, to facilitate real-time updates between the client and server. Also help to ensure instance feedback upon instance like book burrowing and reserving.

While developing this system we had faced many issues. One of the major issues was database data handling. These issues were solved by doing through testing and by improving interactive code.

Testing strategies used to test the system.

A comprehensive testing process is essential to ensure that the library system works properly. Testing was done in 2 approaches.

1. **Automated Testing**

   An automated testing approach was used to create a consistent code with a good quality and error free. The automated testing was done as follows.

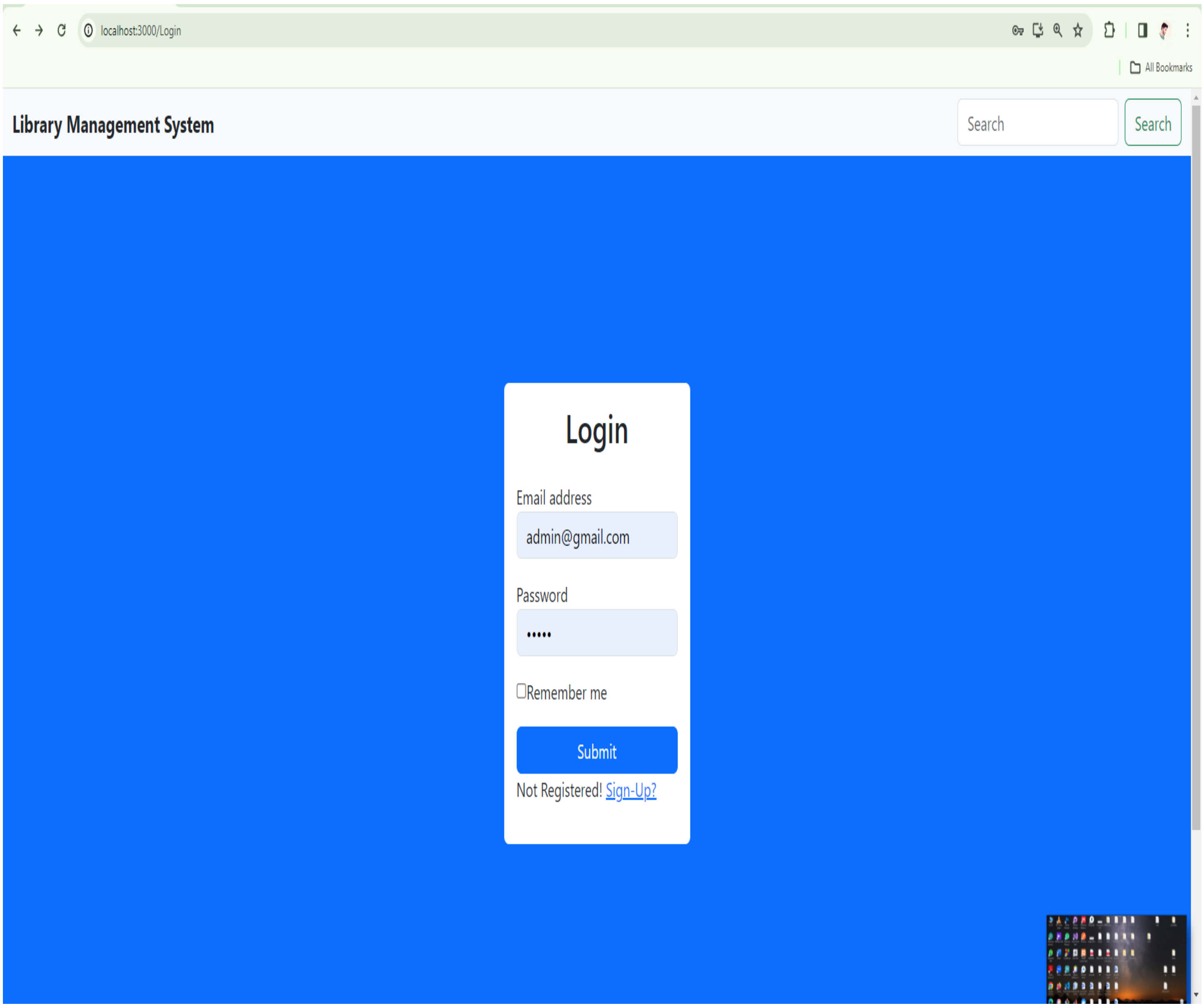
- Unit Testing
  In here we have tested the individual components and functions of the application. We tested the individual React components and Node.js modules to ensure the behaviour of the component. We have use Jest and manual testing to do the testing. For this we have install jest and the react testing library.

- Integration Testing
  In here we have tested the communication between components and servers is successful or not. By doing this testing a seamless data flow was tested. For this we have use Mocha tool tested weather the data flow between components happens well.

- API testing
  API testing was done confirm the accuracy and consistency of the responses in the server.to validate API endpoints Postman tool was used.

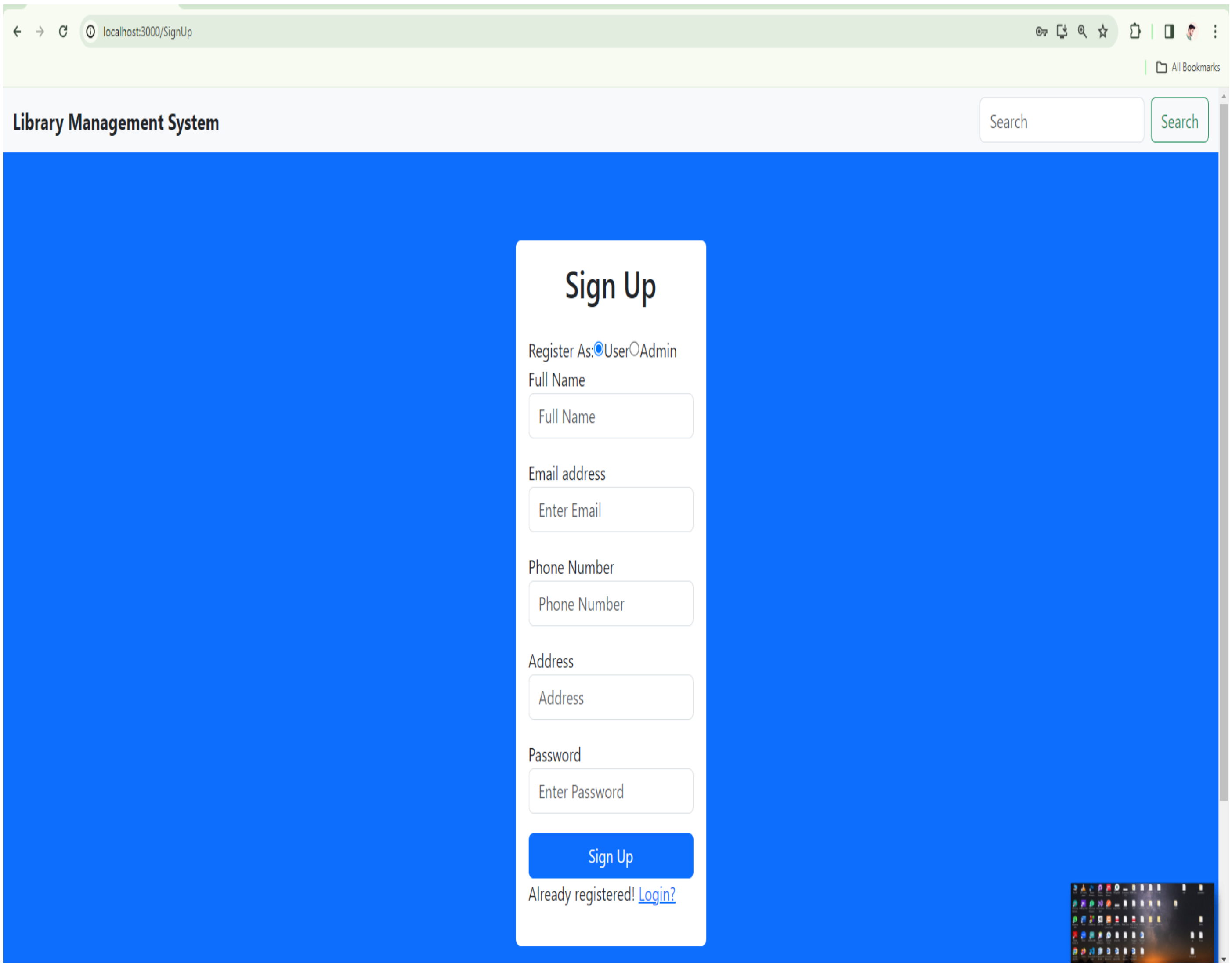
How the testing and system works

- Admin Login



This is the admin login page. By entering username and password they can interact with the system.

- User Sign Up



localhost:3000/SignUp

Library Management System

Search | Search

Sign Up

Register As: [✓ User] Admin

Full Name

Full Name

Email address

Enter Email

Phone Number

Phone Number

Address

Address

Password

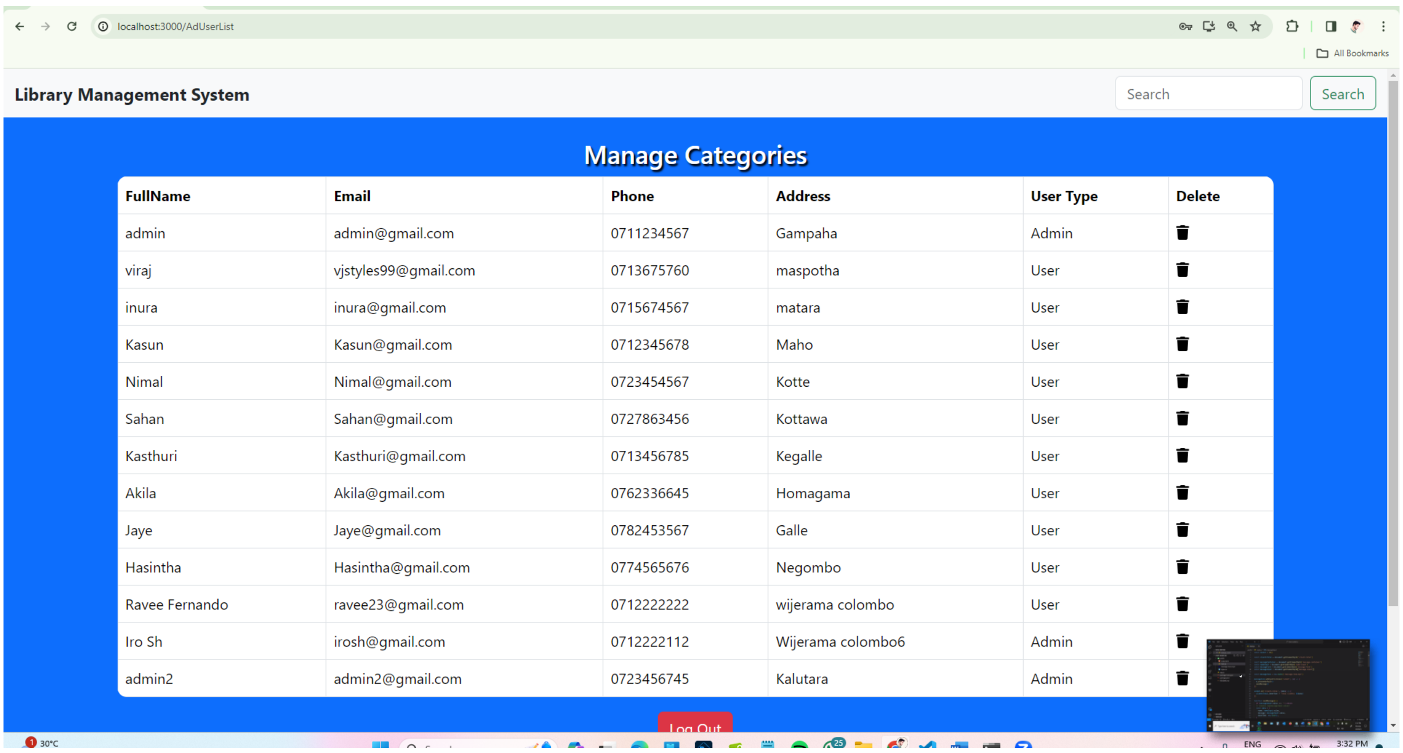
Enter Password

Sign Up

Already registered! Login?

This is the sign up page. Both admin and members can sign up. After click on sign up button a confirmation message will display.

- Managing user details



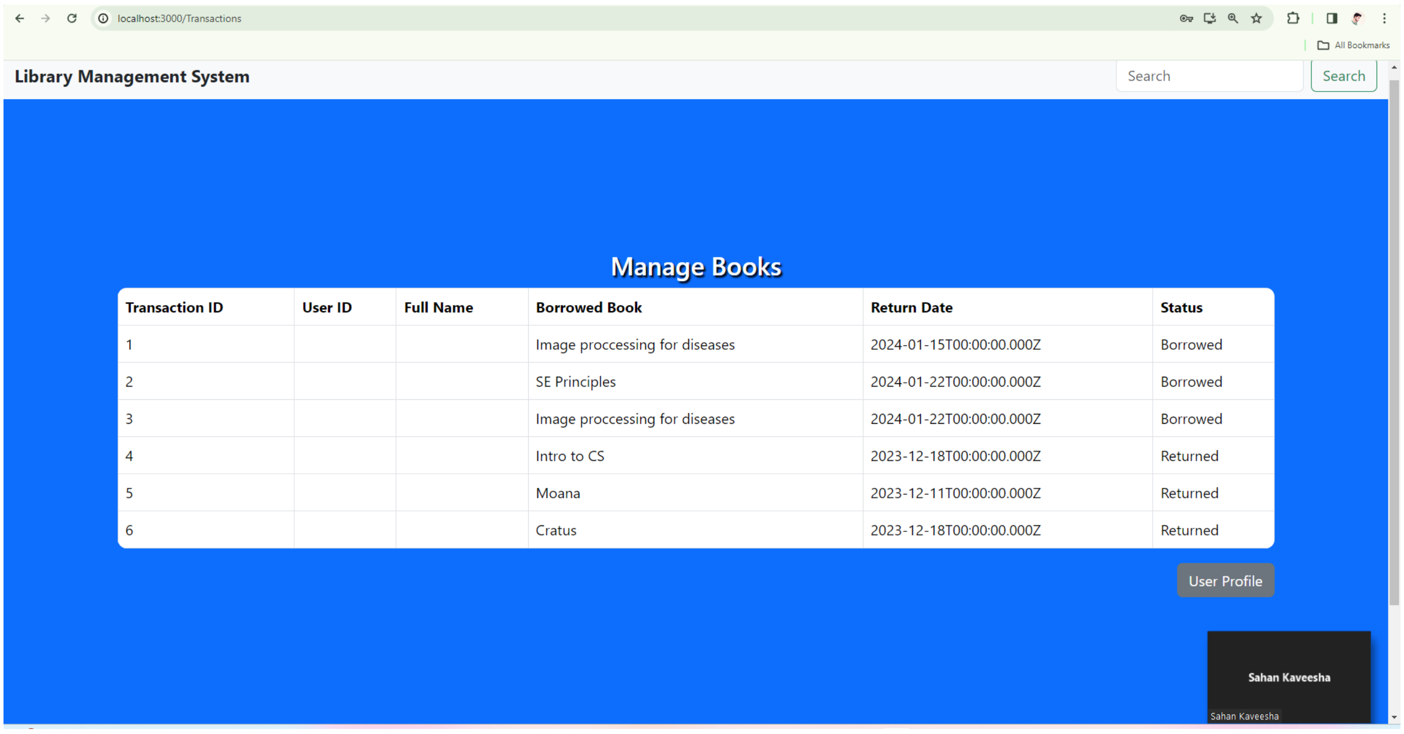
Library Management System

Manage Categories

FullName	Email	Phone	Address	User Type	Delete
admin	admin@gmail.com	0711234567	Gampaha	Admin	🗑
viraj	vjstyles99@gmail.com	0713675760	maspotha	User	🗑
inura	inura@gmail.com	0715674567	matara	User	🗑
Kasun	Kasun@gmail.com	0712345678	Maho	User	🗑
Nimal	Nimal@gmail.com	0723454567	Kotte	User	🗑
Sahan	Sahan@gmail.com	0727863456	Kottawa	User	🗑
Kasthuri	Kasthuri@gmail.com	0713456785	Kegalle	User	🗑
Akila	Akila@gmail.com	0762336645	Homagama	User	🗑
Jaye	Jaye@gmail.com	0782453567	Galle	User	🗑
Hasintha	Hasintha@gmail.com	0774565676	Negombo	User	🗑
Ravee Fernando	ravee23@gmail.com	0712222222	wijerama colombo	User	🗑
Iro Sh	irosh@gmail.com	0712222112	Wijerama colombo6	Admin	🗑
admin2	admin2@gmail.com	0723456745	Kalutara	Admin	🗑

Admin able to manage the user details. Can view and delete records.

- When a user borrowing a book

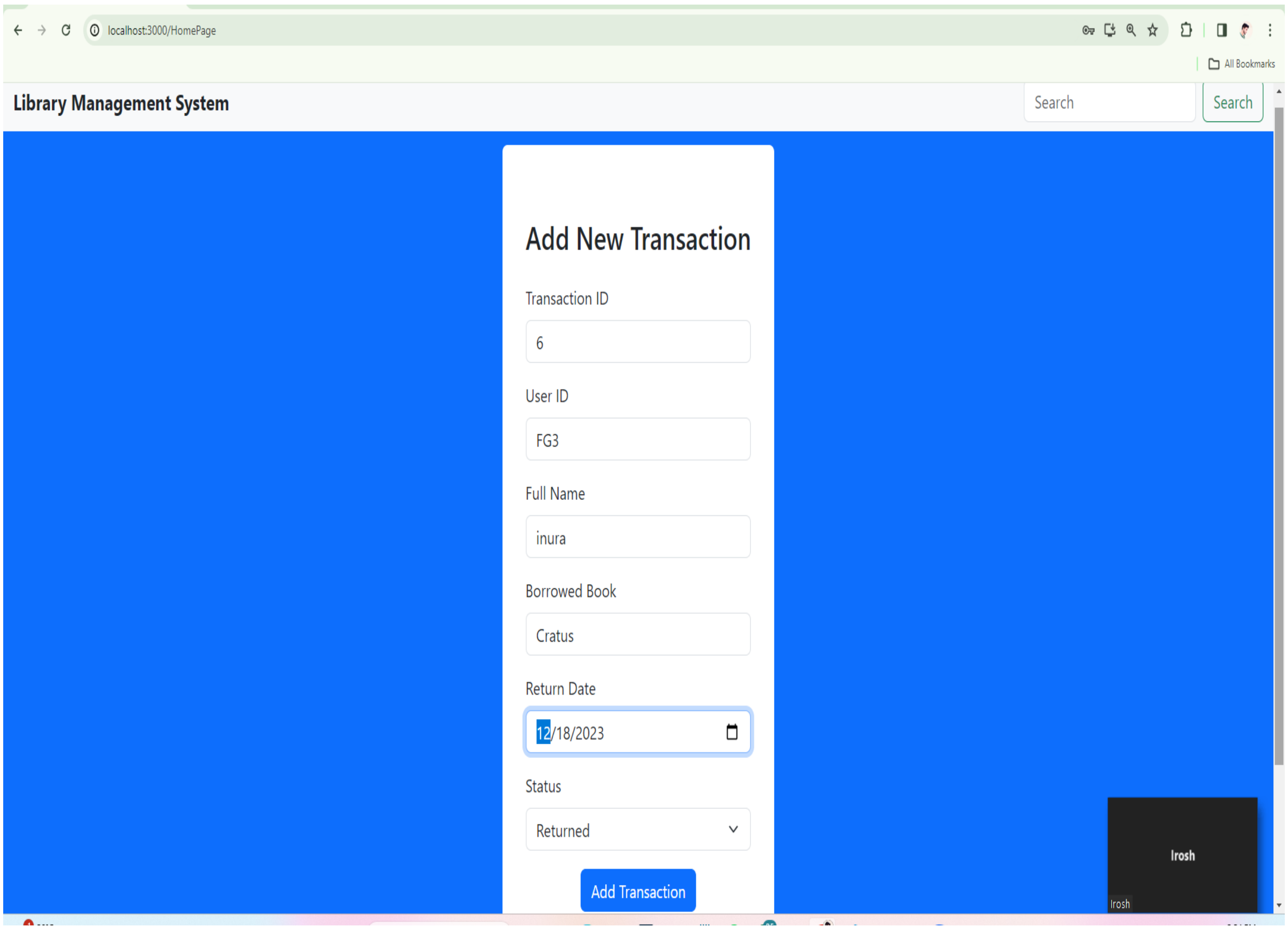


Manage Books

Transaction ID	User ID	Full Name	Borrowed Book	Return Date	Status
1			Image proccessing for diseases	2024-01-15T00:00:00.000Z	Borrowed
2			SE Principles	2024-01-22T00:00:00.000Z	Borrowed
3			Image proccessing for diseases	2024-01-22T00:00:00.000Z	Borrowed
4			Intro to CS	2023-12-18T00:00:00.000Z	Returned
5			Moana	2023-12-11T00:00:00.000Z	Returned
6			Cratus	2023-12-18T00:00:00.000Z	Returned

By using this form users can borrow books.

- The transaction history of books



localhost:3000/HomePage

Library Management System

Search

Add New Transaction

Transaction ID: 6

User ID: FG3

Full Name: inura

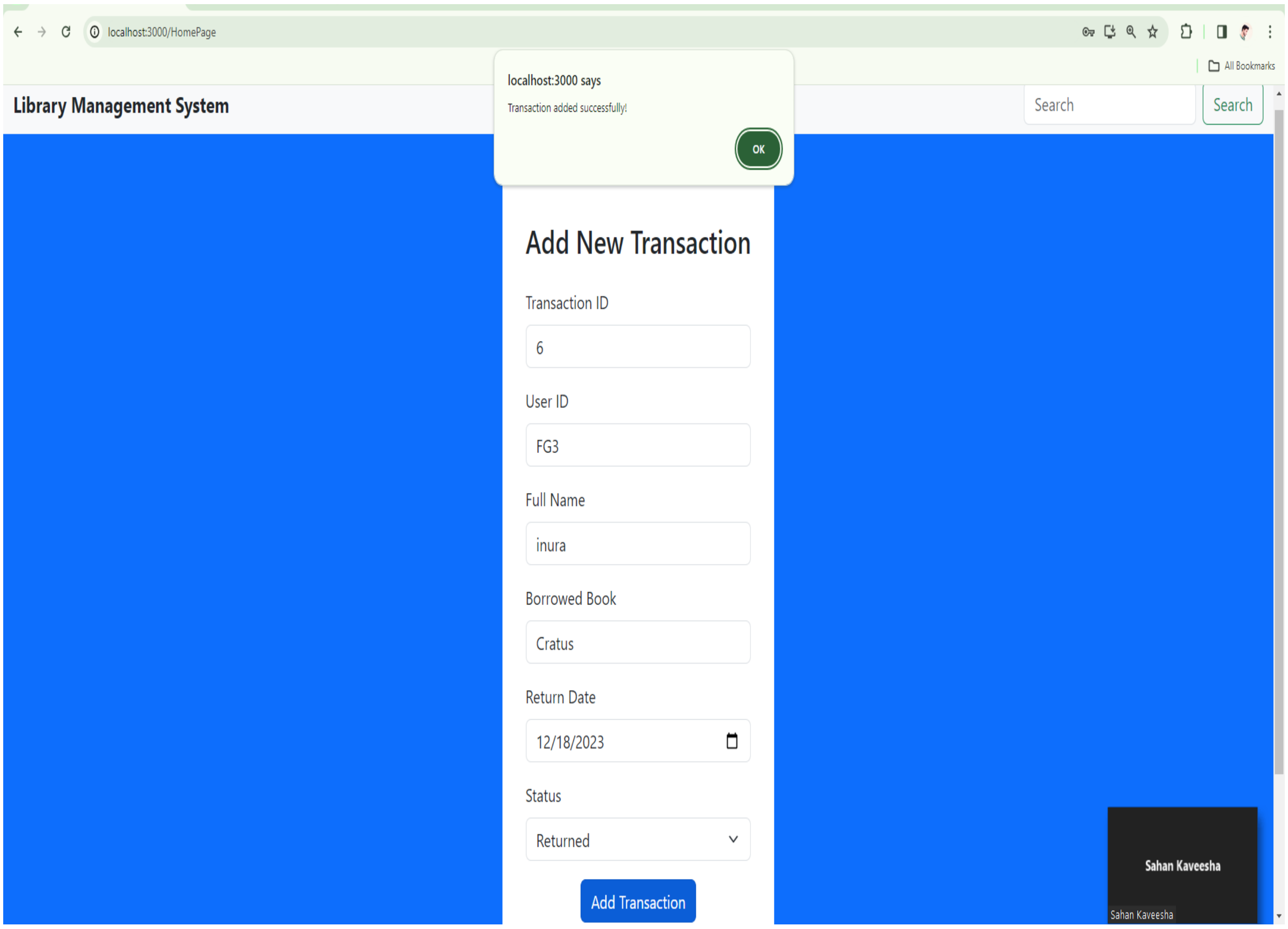
Borrowed Book: Cratus

Return Date: 12/18/2023

Status: Returned

Display the transaction history.

- Display a message when a transaction done.



localhost:3000/HomePage

localhost:3000 says

Transaction added successfully!

Library Management System

Add New Transaction

Transaction ID: 6

User ID: FG3

Full Name: inura

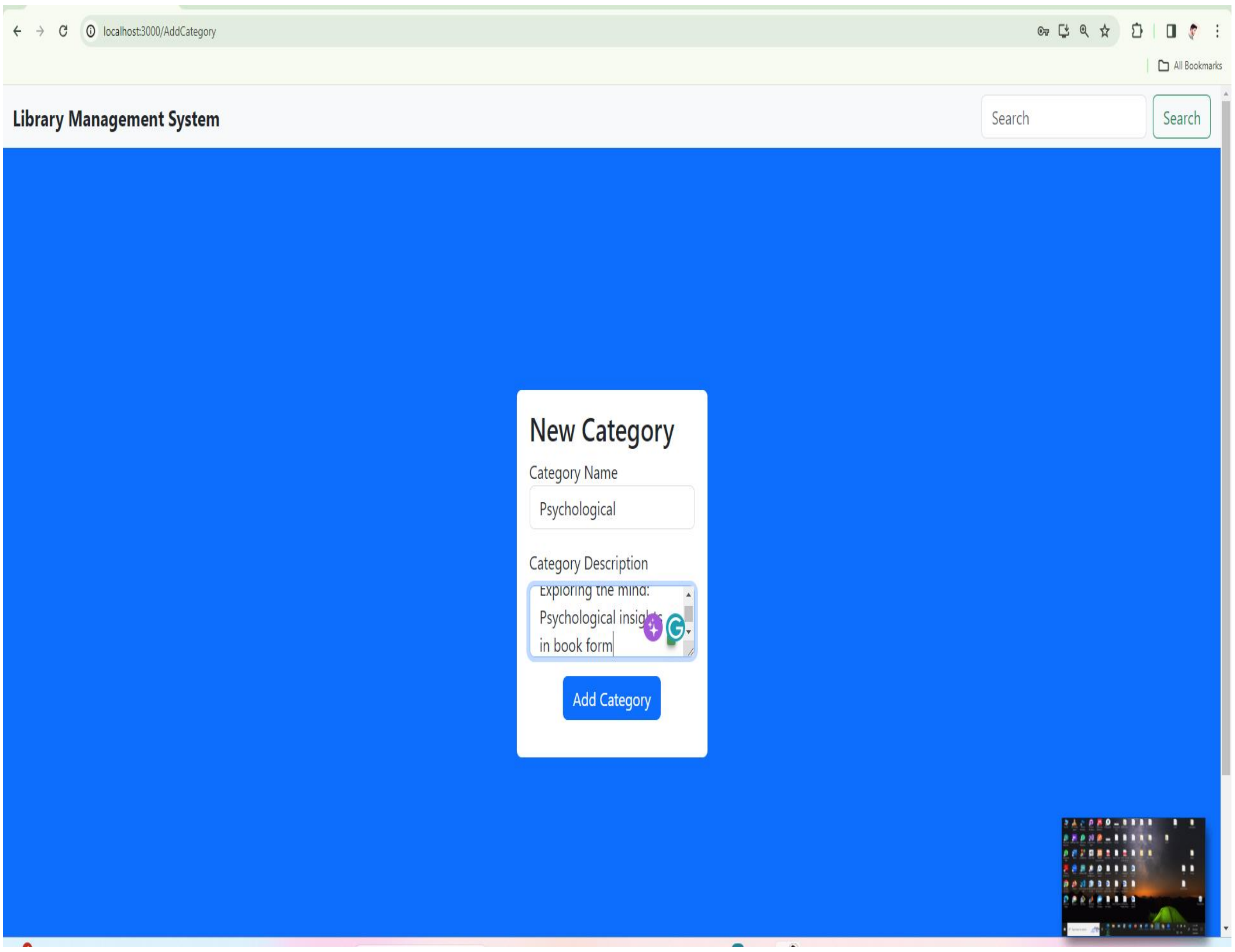
Borrowed Book: Cratus

Return Date: 12/18/2023

Status: Returned

Successful message is display after click on add transaction button.

- Adding category



localhost:3000/AddCategory

Library Management System

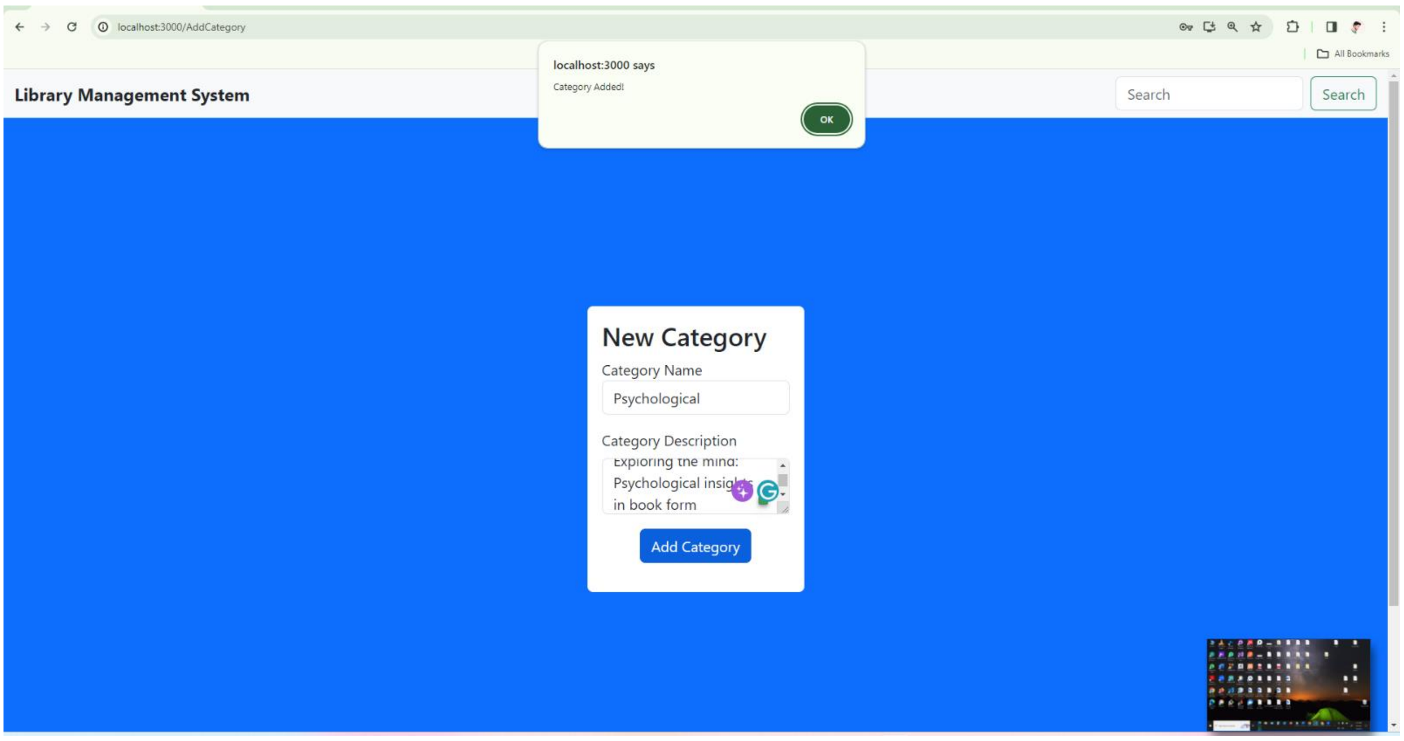
New Category

Category Name: Psychological

Category Description: Exploring the mind: Psychological insights in book form

Allow to add categories.

- After adding categories display a message.


localhost:3000/AddCategory

localhost:3000 says

Category Added!

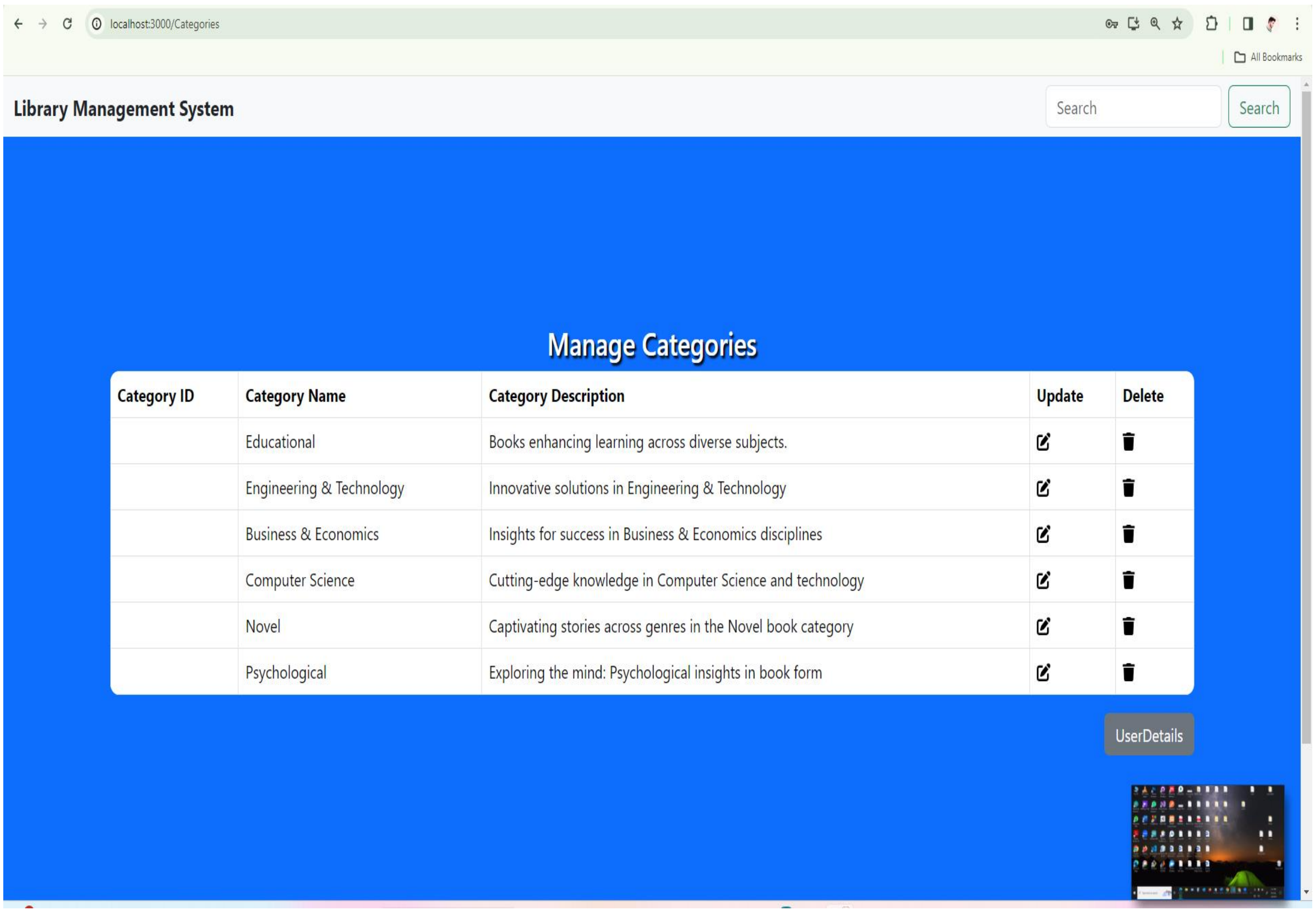
New Category

Category Name: Psychological

Category Description: Exploring the mind: Psychological insights in book form

After click on add category button a confirmation message display.

- Managing Category



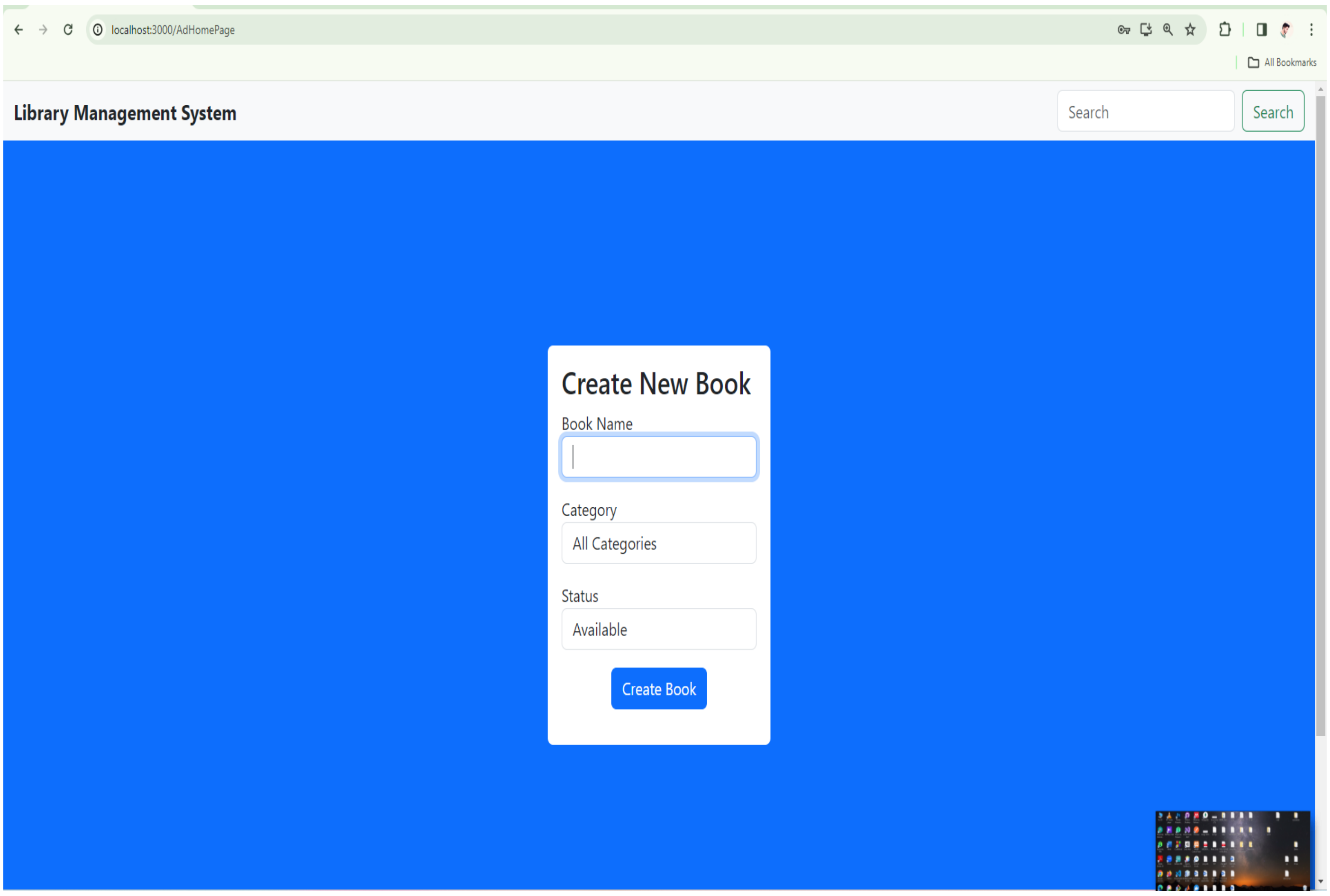
Manage Categories

Category ID	Category Name	Category Description	Update	Delete
	Educational	Books enhancing learning across diverse subjects.		
	Engineering & Technology	Innovative solutions in Engineering & Technology		
	Business & Economics	Insights for success in Business & Economics disciplines		
	Computer Science	Cutting-edge knowledge in Computer Science and technology		
	Novel	Captivating stories across genres in the Novel book category		
	Psychological	Exploring the mind: Psychological insights in book form		

Allow to manage category details.

By click on UserDetai button it navigates to the user details table.

- Adding a book



Create New Book

Book Name

Category

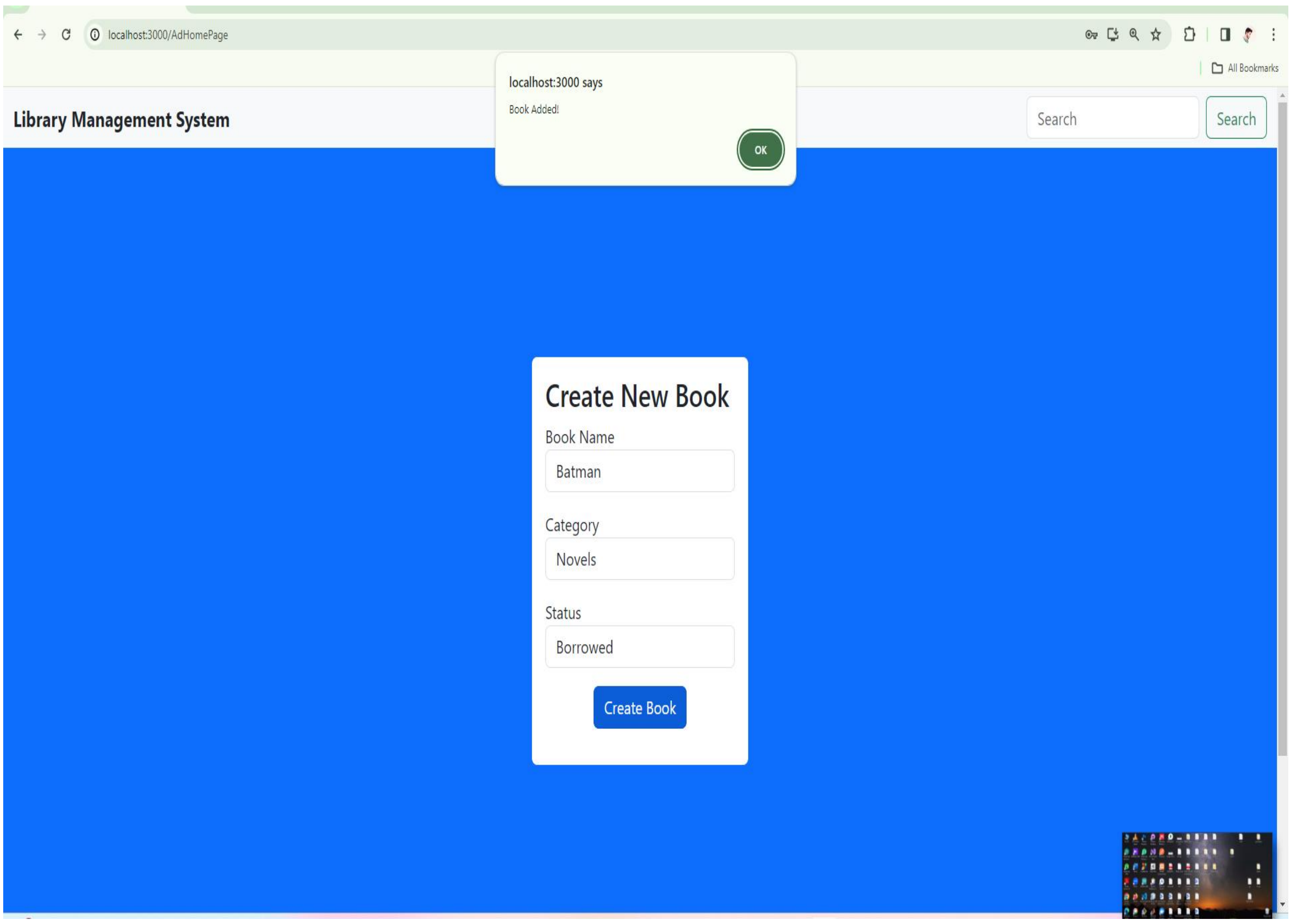
All Categories

Status

Available

Allow to add new books.

- Message showing after a book was added


localhost:3000/AdHomePage

localhost:3000 says

Book Added!

Create New Book

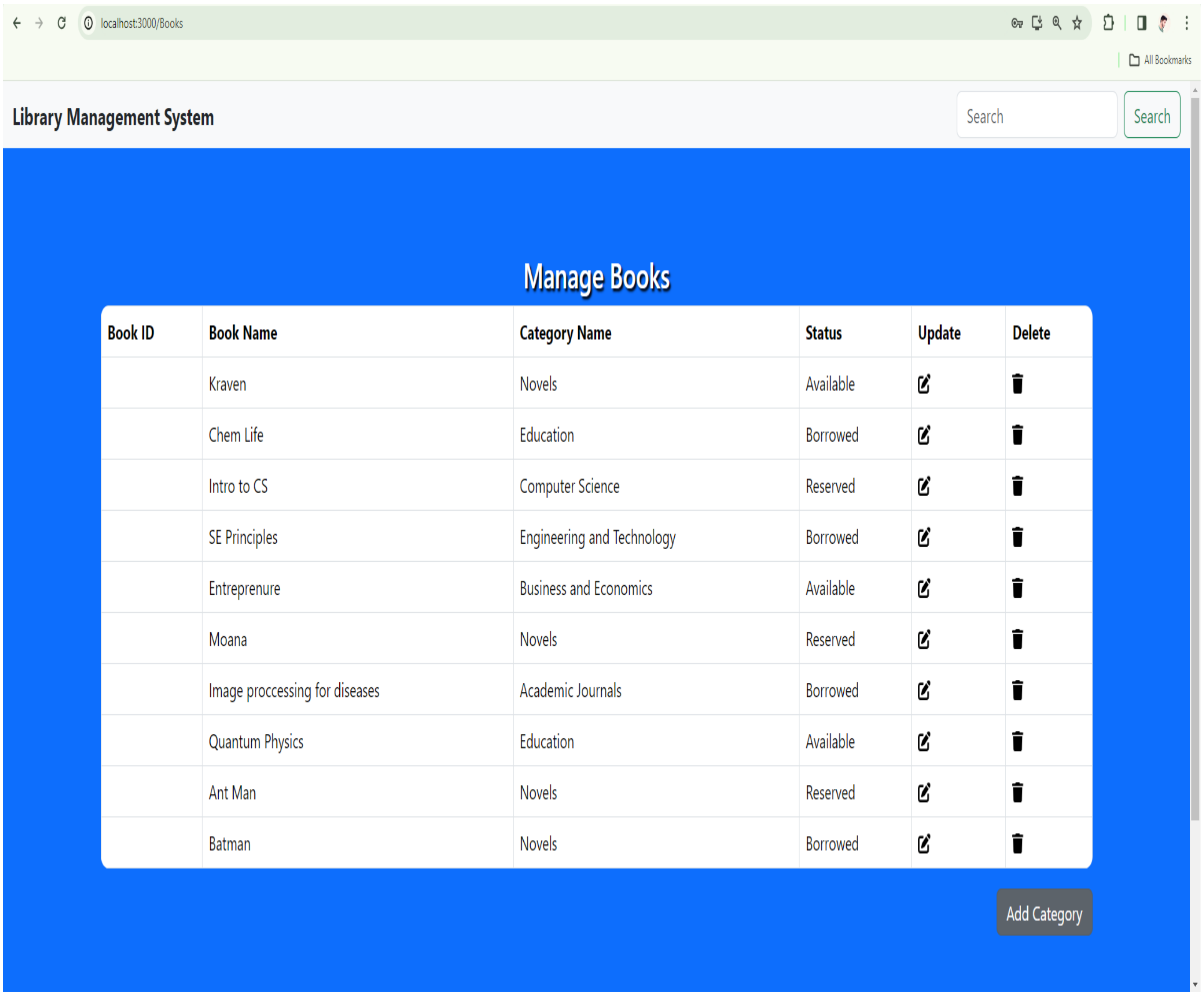
Book Name: Batman

Category: Novels

Status: Borrowed

A confirmation message display after click on Create book button.

- Displaying Book and Category

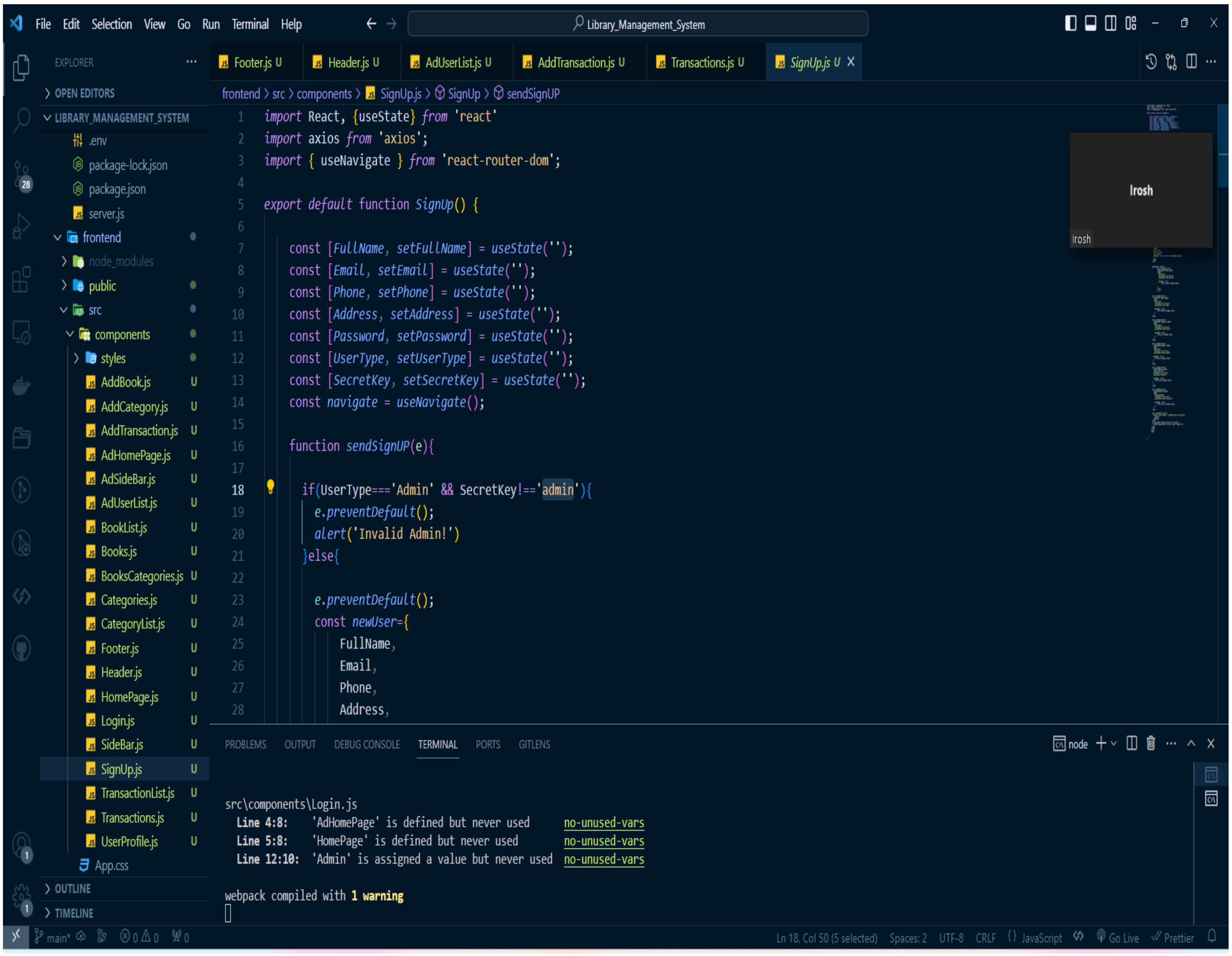


Manage Books

Book ID	Book Name	Category Name	Status	Update	Delete
	Kraven	Novels	Available		
	Chem Life	Education	Borrowed		
	Intro to CS	Computer Science	Reserved		
	SE Principles	Engineering and Technology	Borrowed		
	Entreprenure	Business and Economics	Available		
	Moana	Novels	Reserved		
	Image proccessing for diseases	Academic Journals	Borrowed		
	Quantum Physics	Education	Available		
	Ant Man	Novels	Reserved		
	Batman	Novels	Borrowed		

By clicking on Add Category icon new data can be added. Also, can update and delete by using the relevant icon.

- Front end

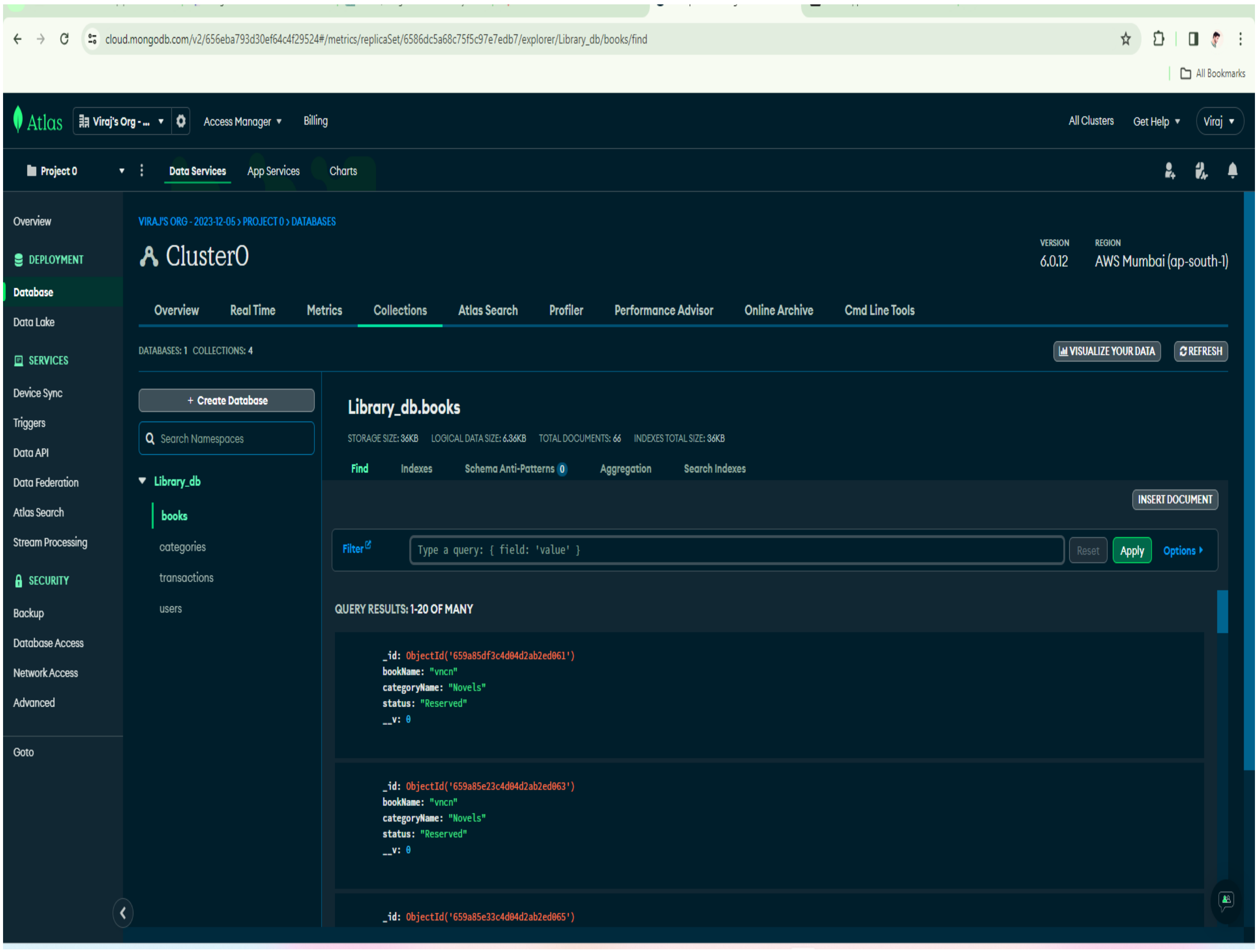


```
import React, {useState} from 'react'
import axios from 'axios';
import { useNavigate } from 'react-router-dom';

export default function SignUp() {

    const [FullName, setFullName] = useState('');
    const [Email, setEmail] = useState('');
    const [Phone, setPhone] = useState('');
    const [Address, setAddress] = useState('');
    const [Password, setPassword] = useState('');
    const [UserType, setUserType] = useState('');
    const [SecretKey, setSecretKey] = useState('');
    const navigate = useNavigate();

    function sendSignUP(e){

      if(UserType==='Admin' && SecretKey!=='admin'){
        e.preventDefault();
        alert('Invalid Admin!')
      }else{

        e.preventDefault();
        const newUser={
            FullName,
            Email,
            Phone,
            Address,
```

```
src\components\Login.js
  Line 4:8:    'AdHomePage' is defined but never used      no-unused-vars
  Line 5:8:    'HomePage' is defined but never used        no-unused-vars
  Line 12:10:  'Admin' is assigned a value but never used  no-unused-vars

webpack compiled with 1 warning
```

Shows that front end running successfully.

- The MongoDB connection.



cloud.mongodb.com/v2/656eba793d30ef64c4f29524#/metrics/replicaSet/6586dc5a68c75f5c97e7edb7/explorer/Library_db/books/find

VIRAJ'S ORG - 2023-12-05 > PROJECT 0 > DATABASES

Cluster0

VERSION 6.0.12 REGION AWS Mumbai (ap-south-1)

DATABASES: 1 COLLECTIONS: 4

Library_db.books

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 6.36KB TOTAL DOCUMENTS: 66 INDEXES TOTAL SIZE: 36KB

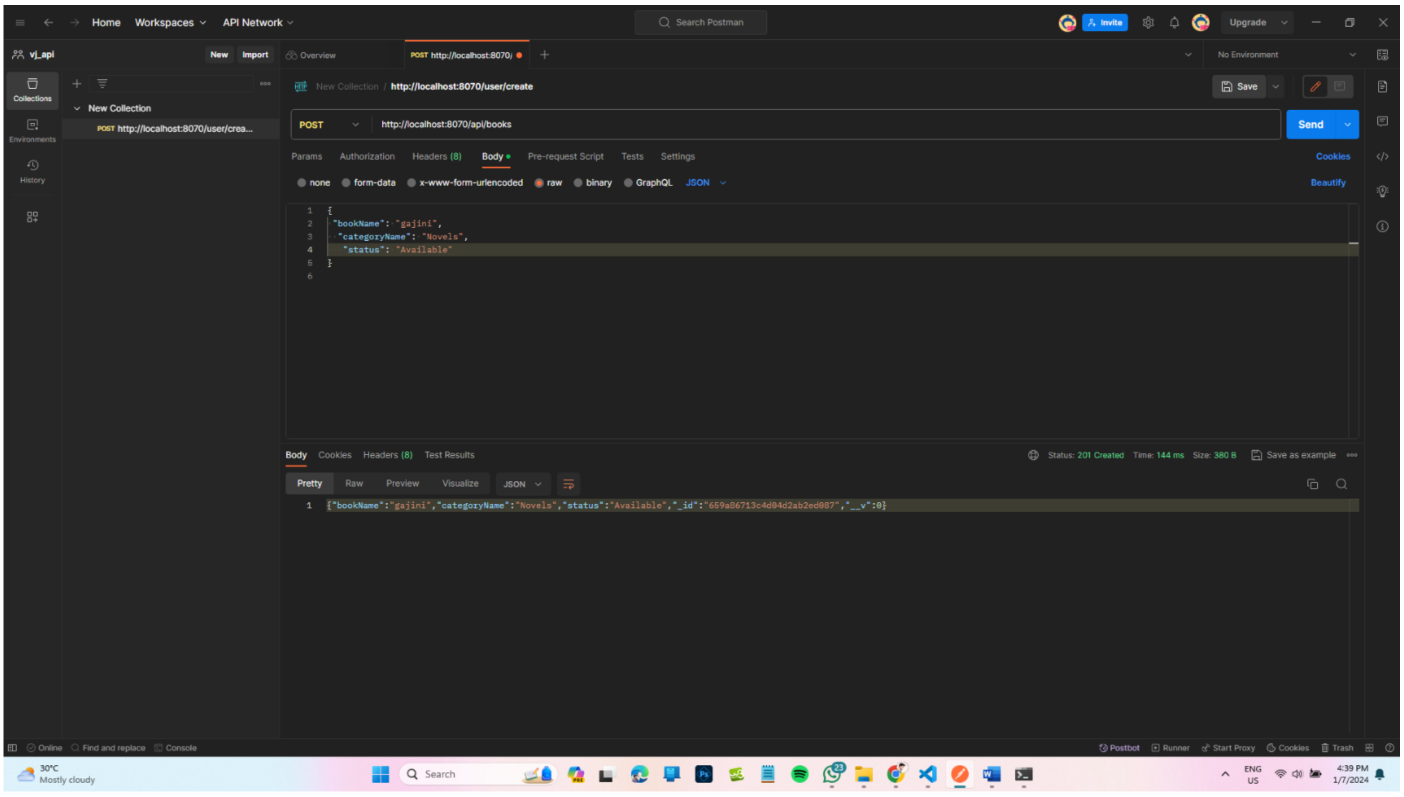
QUERY RESULTS: 1-20 OF MANY

```
_id: ObjectId('659a85df3c4d04d2ab2ed061')
bookName: "vncn"
categoryName: "Novels"
status: "Reserved"
__v: 0

_id: ObjectId('659a85e23c4d04d2ab2ed063')
bookName: "vncn"
categoryName: "Novels"
status: "Reserved"
__v: 0

_id: ObjectId('659a85e33c4d04d2ab2ed065')
```

The database is connected properly.

Group 22

- Checking API book creating



New Collection / http://localhost:8070/user/create

POST http://localhost:8070/api/books

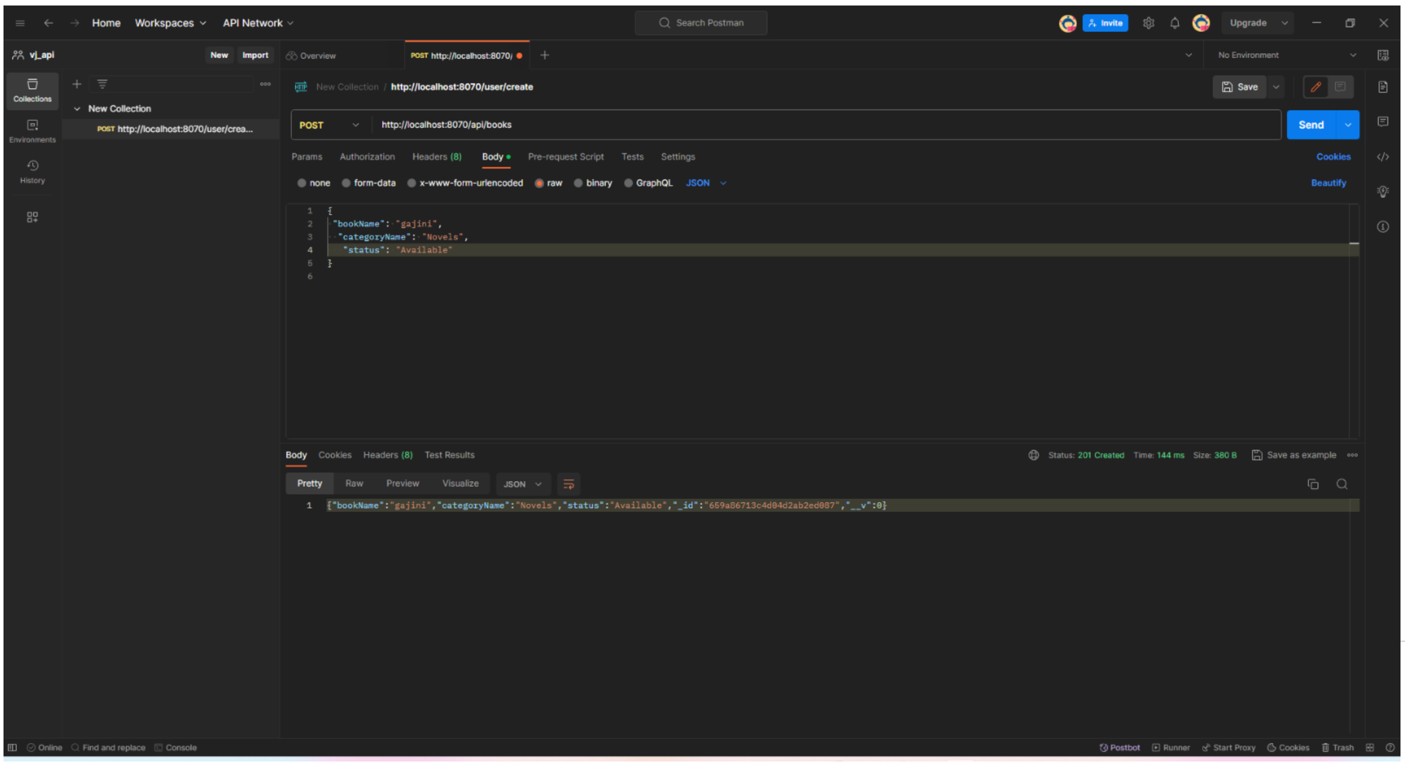
Body: raw JSON

```
{
 "bookName": "gajini",
  "categoryName": "Novels",
   "status": "Available"
}

```

Status: 201 Created Time: 144 ms Size: 380 B

```
{"bookName":"gajini","categoryName":"Novels","status":"Available","_id":"659a86713c4d04d2ab2ed087","__v":0}
```

- Checking API using postman.



New Collection / http://localhost:8070/user/create

POST http://localhost:8070/api/books

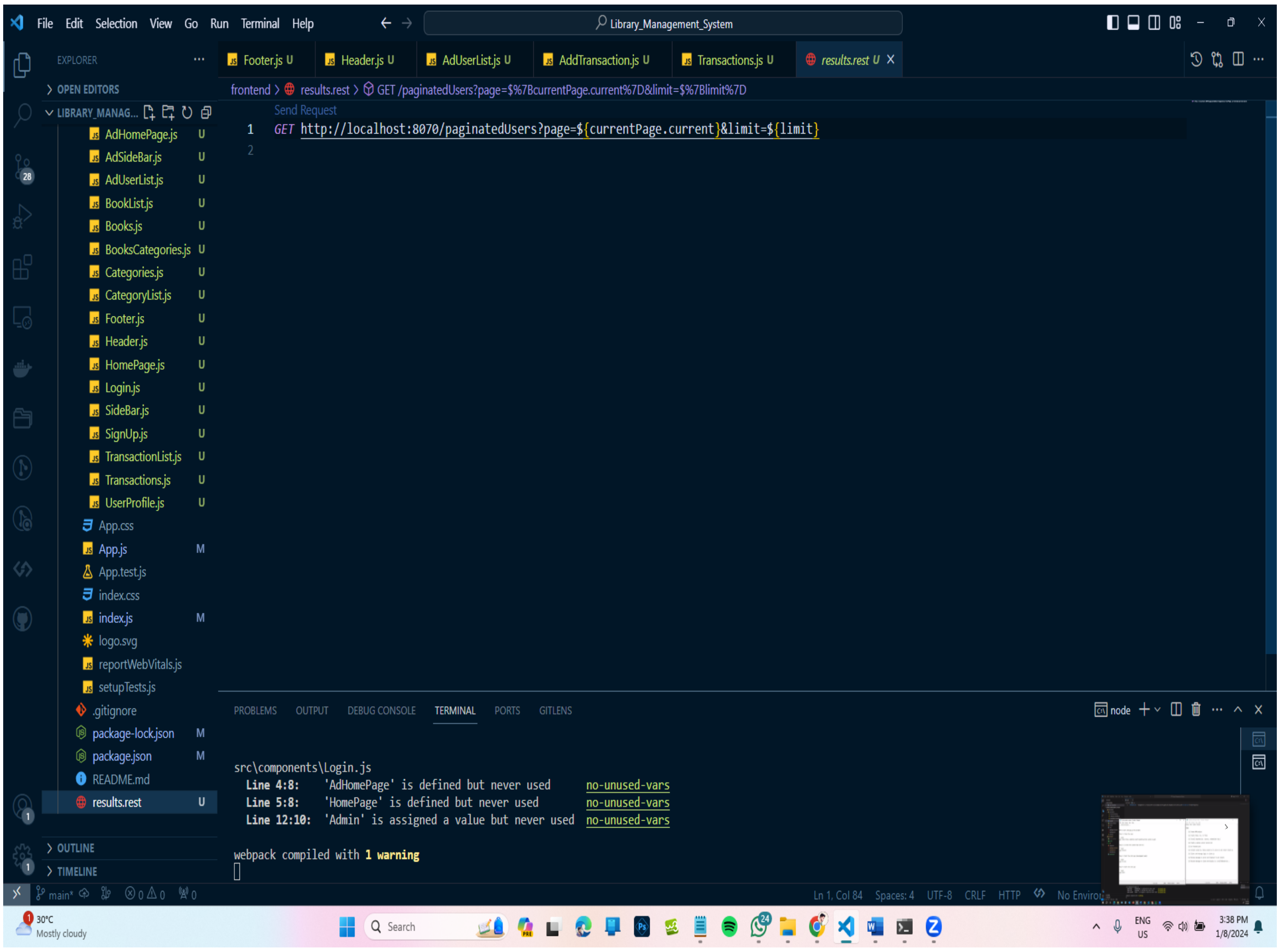
Body: raw JSON

```
{
 "bookName": "gajini",
  "categoryName": "Novels",
   "status": "Available"
}

```

Status: 201 Created Time: 144 ms Size: 380 B

```
{"bookName":"gajini","categoryName":"Novels","status":"Available","_id":"659a86713c4d04d2ab2ed087","__v":0}
```

- Rest extension was used to check http requests.



```
Send Request
GET http://localhost:8070/paginatedUsers?page=${currentPage.current}&limit=${limit}
```

```
src\components\Login.js
  Line 4:8:    'AdHomePage' is defined but never used      no-unused-vars
  Line 5:8:    'HomePage' is defined but never used        no-unused-vars
  Line 12:10:  'Admin' is assigned a value but never used  no-unused-vars

webpack compiled with 1 warning
```

2. Usability testing.

This usability testing approach was done to test the user experience in this system.

- Prototype Testing.
  In here we have tested the early mock-ups and wireframes. Bu using the representative users this testing was done in order t gather feedback on intuitiveness, navigation of the system and the way of information visualization.

- A/B testing.
  This test was done on different UI elements. To identify the most user-friendly options their functionalities have been compared.

- User Interviews.
  The team members tested the user interfaces manually by adding data.

By doing testing as mentioned in above, it helps us to identify the bugs early and fix them. Continuous improvement could be seen. It helps to be confident about the system stability and the quality was enhance. It ensures that users can enjoy the system by the accessibility testing.

DevOps Pipelines

The development Environment.

In the implementation of this system as the Version Control we have use git. Git helped us to track the changes we have done to the code and help for the code collaboration. We use Visual Studio code as our Code Editor because this is the widely used editor it contains many tools which help to develop and debug systems effectively. We have use nmp and yarn package mangers to manage the front-end and back-end dependencies. We have used. While developing this project local; development servers have been used. For the API Node.js servers as used. For the front-end testing React development server was used. For the testing of the system, automated tool like jest, mocha, cypress and postman tools were used as integrated to the pipeline. Environmental variable has been used to store sensitive information like database credentials. Foe this we have generate access tokens.

In this project we have used Continuous integration (CI) and continuous deployment (CD) principles.

Continuous Integration (CI) pipeline.

- Triggering.
  To initiate the pipeline, we have commits new codes automatically.

- Building.
  When building this system, we have built the required dependencies. The code formatting was checks by running linting. We have run the react components. Also, we have run the unit tests and integration tests.

- Testing.
  We have done automated testing for the front-end, back-end and API layers. Usability testing was done to test errors and bugs in the user end.

- Deployment.
  After the test cases pass, for further testing the system deploy to a staging environment. Based on the testing the system approved and use as a product.

Pipeline Tool
As the pipeline tool we have used GitHub Actions. This tool helped us to automate the CI/CD process with the GitHub easily. This helps us to monitor the performance, track the issues in integration and help to maintain adaptable system and do continuous improvements to maintain the system.

Personal reflection

By doing this project we have learn about the MEARN stack. As a new technology we learn about React. We learn about automated testing tools while developing this project. As new things we learn about postman API checking too, use of jest in front-end testing and mocha use in testing. We have learned about Continuous integration pipelines also. MongoDB connection was difficult, but we were able to manage it. Installing some dependencies using npm was difficult.

The lessons that taken from this project was it need clear documentations while developing a system. The continuous improvement is needed in project. We identify that time management is essential because need to develop a system in a small period. The collaboration of team members is essential. A good communication should be there when doing a group project.

Conclusion

To develop the library system using of MERN stack is efficient. The automated testing helps to ensure the user better experience. The prioritizing of core features is important. And integrating all testing and maintain clear documentation is import for continues progress of the project.

References

Bang Nguyen, 2021. *Improving web development process of MERN stack,* s.l.: Metropolia university of applied science.

Inuri Gamage,R. G. I. Nawodya,M. N. T. Ahmed,M. A. M. Aathif,D. I. De Silva,Thisuru Dias, 2022. Effectiveness of Cutting-Edge Technology for Library Management System. *International Journal of Engineering and Management Research,* 12(5), p. 7.

Mohanish Bawane,Ishali Gawande,Vaishnavi Joshi,Rujuta Nikam, Prof. Sudesh A. Bachwani, 2022. A Review on Technologies used in MERN stack. *International Journal for Research in Applied Science & Engineering Technology (IJRASET),* 10(1), pp. 2321-9653.

Samikshya Aryal, 2020. *MERN STACK WITH MODERN – Developers Connecting Application,* s.l.: Turku university of applied science.

Saravanan Raju, S.Soundararajan, V.Loganathan, 2021. MERN Stack Web Application. *Annals of the Romanian Society for Cell Biology,,* 25(6), p. 6325–6332.

Appendix

- Requirements

  https://docs.google.com/document/d/1yNXkY25oZP4LkJxg9EOR59fLn8MInX29r5B9Cu6HE5w/edit?usp=sharing

- Test cases

  https://docs.google.com/spreadsheets/d/1-5twtZTi1QFuxUZdyDM2qbTCV7Dxc1iz1TdQNLipBts/edit?usp=sharing