ONLINE HOSTEL ROOM BOOKING

Project report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Artificial Intelligence & Data Science

By

DONGARI AJITH KUMAR (21UEAD0016)
CHAMARTHI LAHAREE (21UEAD0009)
PYDIMUKKALA SRAVANI (21UEAD0052)

10212AD212 - WEB AND MOBILE APPLICATION DEVELOPMENT

SUMMER 2023-2024

Under the guidance of
FACULTY NAME,Degree.,
ASSISTANT PROFESSOR

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & DATA SCIENCE
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

October,2023

CERTIFICATE

It is certified that the work contained in the project report titled ”ONLINE HOSTEL ROOM BOOKING” by ”DONGARI AJITH KUMAR (21UEAD0016),CHAMARTHI LAHAREE (21UEAD0009), PYDIMUKKALA SRAVANI (21UEAD0052)” has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor
Supervisor name
Designation
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr.Sagunthala R&D
Institute of Science & Technology
October, 2023

Signature of Head of the Department
Artificial Intelligence & Data Science
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
October,2023

Signature of the Dean
Dr. V. Srinivasa Rao
Professor & Dean
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
October, 2023

DECLARATION

We declare that this written submission represents my ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

DONGARI AJITH KUMAR

Date: / /

(Signature)

CHAMARTHI LAHAREE

Date: / /

(Signature)

PYDIMUKKALA SRAVANI

Date: / /

APPROVAL SHEET

This project report entitled ”ONLINE HOSTEL ROOM BOOKING” by DONGARI AJITH KUMAR (21UEAD0016), CHAMARTHI LAHAREE (21UEAD0009), (PYDIMUKKALA SRAVANI (21UEAD0052) is approved for the degree of B.Tech in Artificial Intelligence & Data Science.

Examiners

Minu Inba Shanthini Watson Benjamin.,M.E.,

Date: / /

Place:

ABSTRACT

The ”Hostel Room Booking” project is a web-based application developed using HTML, CSS, and the Bootstrap framework, designed to simplify the process of booking hostel rooms. This project offers an intuitive and user-friendly interface for both guests and hostel owners to manage room reservations efficiently. It encompasses features such as real-time room availability, secure payment processing, and seamless communication between users and hostel administrators. The project aims to provide a modern and hassle-free solution to cater to the increasing demand for online hostel room bookings.

Keywords: Hostel Room Booking, HTML, CSS, Bootstrap, Web Application, Accommodation, User Interface, Payment Processing, Availability Updates, Communication.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

S.NO	ABBREVIATIONS	DEFINITION
1.	JS	JavaScript
2.	HTTP	HyperText Transfer Protocol
3.	CSS	Cascading Style Sheet
4.	HTML	Hypertext Markup Language
5.	CDN	Content Delivery Network
6.	PHP	Personal Home Page
7.	DOM	Document Object Model
8.	AJAX	Asynchronous JavaScript And XML
9.	SQL	Structured Query Language
10.	API	Application Programming Interface
11.	RDBMS	Relational DataBase Management Systems

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Traditional methods of booking hostel rooms have become outdated in today's fast-paced digital world. The "Hostel Room Booking" project addresses this issue by offering a web-based solution that streamlines the entire booking process. With an intuitive user interface and robust features, this project aims to revolutionize how hostels manage reservations and how guests secure their accommodations.

1.2 Aim of the project

The primary aim of the "Hostel Room Booking" project is to develop a user-friendly web application that simplifies the booking of hostel rooms. This application will enable users to search for hostels, check room availability in real-time, make secure payments, and receive instant booking confirmations. Additionally, the project aims to provide hostel administrators with a management system to efficiently handle bookings and guest communication.

1.3 Project Domain

The project falls within the domain of web development and hostel room management. It involves the integration of web technologies such as HTML, CSS, and the Bootstrap framework to create an online platform for booking hostel accommodations. Additionally, it incorporates elements of e-commerce and secure payment processing to facilitate transactions.

1.4 Scope of the Project

The scope of the "Hostel Room Booking" project includes the following key features are User registration and login functionality Hostel listings with detailed descriptions, images, and room availability status Real-time room availability updates Secure payment processing through various payment gateways Search functionality based on location, price range, and amenities Booking management

for both users and hostel administrators Guest reviews and ratings for hostels Admin panel for site management and monitoring.

1.5 Methodology

Front-End Development:

1.HTML

2.CSS

3.JavaScript

4.Bootstrap

5.Angular

Back-End Development:

1.Node.js

2.Express.js

3.MySQL Database

4.XAMPP(Apache,PHP,MySQL

Chapter 2

REQUIREMENT SPECIFICATION

2.1 User characteristics

The ”Hostel Room Booking” project caters to a diverse set of users, each with specific characteristics and needs:

1. Guests/Customers:

- Typically individuals seeking accommodation in hostels.
- May range from budget travelers to tourists and business travelers.
- May have varying levels of familiarity with technology.
- Need an easy-to-use interface for browsing hostels, checking availability, and making bookings.

2. Hostel Owners/Administrators:

- Managers or owners of hostels.
- Require access to a secure admin panel.
- Need to manage room availability, bookings, and communicate with guests.
- May not be tech-savvy, so the admin panel should be user-friendly.

3. System Administrators (IT):

- Responsible for maintaining the web application’s server and database.
- Need technical expertise in server maintenance and troubleshooting.

4. Developers/Programmers (During Development):

- The project development team responsible for creating the web application.
- Proficient in web development technologies, including HTML, CSS, Bootstrap, and back-end technologies.

2.2 Dependencies

The ”Hostel Room Booking” project relies on several dependencies to function effectively. These dependencies encompass various aspects, including technology, data sources, and external services. Here are the key dependencies for the project:

1. Internet Connectivity:

- The project depends on a stable internet connection for users to access the web application and make bookings. Hostel owners/administrators also require internet access to manage bookings and update room availability.

2. Web Hosting Service:

- A reliable web hosting service or server infrastructure is essential to host and deploy the web application. The hosting service should provide sufficient resources and uptime to ensure the application's availability.

3. Database Management System (DBMS):

- The project relies on a DBMS to store and manage data related to user accounts, hostel listings, room availability, and booking records. The choice of DBMS (e.g., MySQL, PostgreSQL) is a critical dependency.

4. Payment Gateway Integration:

- To facilitate secure online payments for room bookings, the project depends on integration with one or more payment gateways (e.g., PayPal, Stripe, or others). These payment gateways provide APIs and services for handling financial transactions.

5. Development Tools and Frameworks:

- Front-end development relies on HTML, CSS, and the Bootstrap framework. The availability of these technologies and development tools is crucial for building the user interface and ensuring responsive design.

2.3 Hardware specification

The hardware specifications for hosting and accessing the ”Hostel Room Booking” project, which uses HTML, CSS, and the Bootstrap framework, are relatively modest.hardware requirements are:

Server:

1. Processor (CPU): A multi-core processor, such as a modern Intel Core or AMD Ryzen processor, is sufficient for hosting the web application. The exact CPU requirements depend on the expected traffic and workload.

2. Memory (RAM): The server should have sufficient RAM to handle concurrent user requests and database operations. A minimum of 4GB of RAM is recommended, but higher capacities are preferable for better performance.

3. Storage: Adequate storage space is required to store the web application files, databases, and any associated assets. SSDs (Solid State Drives) are recommended for improved data access speed.

4. Network Connectivity: A reliable and high-speed internet connection is necessary for hosting the web application and ensuring its responsiveness to user requests.

Client Devices:

1. Computers and Laptops: Users and hostel owners/administrators can access the web application using standard desktop or laptop computers with modern web browsers.

2. Mobile Devices: The web application should be responsive and accessible on various mobile devices, including smartphones and tablets. Users should have access to a compatible web browser on their mobile devices.

Web Browsers:

The web application should be compatible with modern web browsers, including but not limited to:

- Google Chrome
- Mozilla Firefox
- Apple Safari
- Microsoft Edge

Development and Maintenanc:

For developers and administrators involved in the project's development and maintenance, the following hardware is recommended:

1. Development Computer: A computer with sufficient processing power, RAM, and storage for coding, testing, and debugging tasks.

2. Backup and Redundancy: Implementing backup solutions and redundancy measures, such as RAID configurations, can help ensure data reliability and availability.

The hardware specifications can vary based on the scale and expected traffic of the project. It's important to periodically assess and upgrade hardware as needed to accommodate growth in user traffic and data volume. Additionally, consider cloud hosting solutions for scalability and ease of management.

2.4 Software specification

The software specifications for the "Hostel Room Booking" project, which utilizes HTML, CSS, and the Bootstrap framework, encompass a range of software components and tools required for both development and deployment. Here are the essential software specifications:

Server-Side Software:

1. Operating System: The choice of the server's operating system depends on your development team's familiarity and the hosting environment. Common choices include:
- Linux distributions (e.g., Ubuntu, CentOS) for open-source solutions.
- Windows Server for environments requiring Microsoft technologies.

2. Web Server:A web server software is required to host and serve the web application. Common options include:
- Apache HTTP Server
- Nginx
- Microsoft Internet Information Services (IIS)

Front-End Software:

1. HTML: The project relies on HTML (Hypertext Markup Language) for structuring web content.

2. CSS: Cascading Style Sheets (CSS) are used for styling the web application's user interface and ensuring a visually appealing design.

3.Bootstrap Framework: Bootstrap is a front-end framework that provides pre-designed UI components and responsive layouts, simplifying the development of a mobile-friendly and consistent user interface.

4. JavaScript: JavaScript is used for adding interactivity to the web application. Libraries and frameworks like jQuery may also be utilized for enhanced functionality.

Development Tools:

1. Text Editor or Integrated Development Environment (IDE): Developers use text editors (e.g., Visual Studio Code, Sublime Text) or IDEs (e.g., JetBrains WebStorm, Eclipse) for writing, debugging, and testing code.

2. Package Managers: Depending on the chosen server-side scripting language, package managers like Composer (PHP), npm (Node.js), pip (Python), or gem (Ruby) are used to manage project dependencies and packages.

Database Tools:

1. Database Client: A database client tool (e.g., phpMyAdmin for MySQL) is useful for managing and administering the database, including creating tables, running queries, and performing backups.

Deployment and Hosting:

1. Web Hosting Control Panel: For server management and deployment, web hosting control panels like cPanel, Plesk, or custom server administration tools can be used.

2. Cloud Services: Consider using cloud platforms like Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure for scalable and reliable hosting options.

3. Containerization and Orchestration (Optional): Tools like Docker and Kubernetes can be used to containerize the application for easier deployment and scaling.

Testing and Debugging:

1. Testing Frameworks: Depending on the chosen server-side language, testing frameworks like PHPUnit (PHP), Jasmine (JavaScript), or PyTest (Python) can be used for automated testing.

2. Browser Developer Tools: Web browser developer tools (e.g., Chrome DevTools) are essential for debugging front-end code and troubleshooting issues.

These software specifications provide a foundation for the development, deployment, and maintenance of the ”Hostel Room Booking” project. The specific software choices and configurations may vary based on the project's requirements and the preferences of the development team.

Chapter 3

WEBSITE DESIGN

3.1 Sitemap

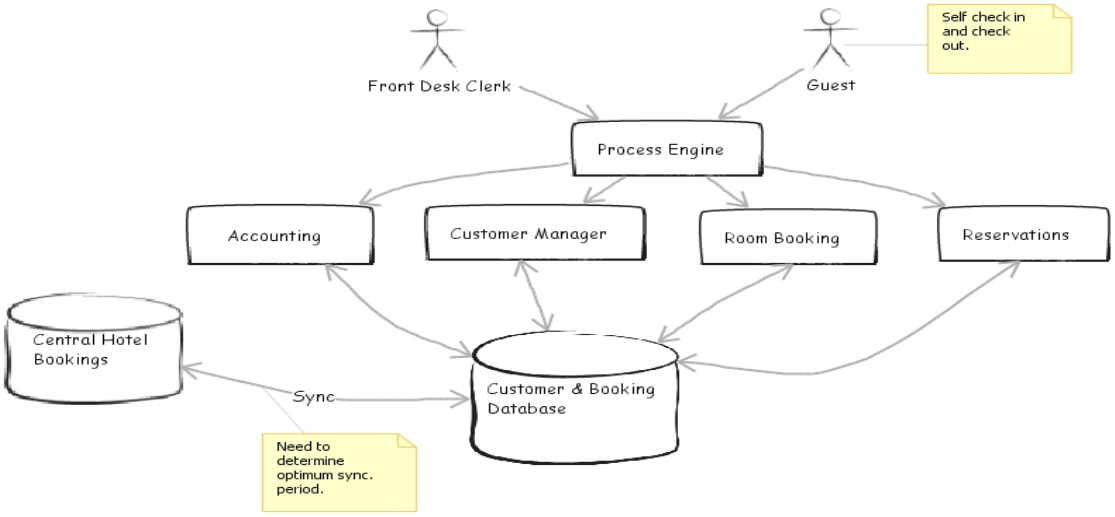

Figure 3.1: **Architecture Diagram**

In figure 3.1 represents the software specifications provide a foundation for the development, deployment, and maintenance of the ”Hostel Room Booking” project. The specific software choices and configurations may vary based on the project’s requirements and the preferences of the development team.

3.2 Design Phase

3.2.1 Data Flow Diagram

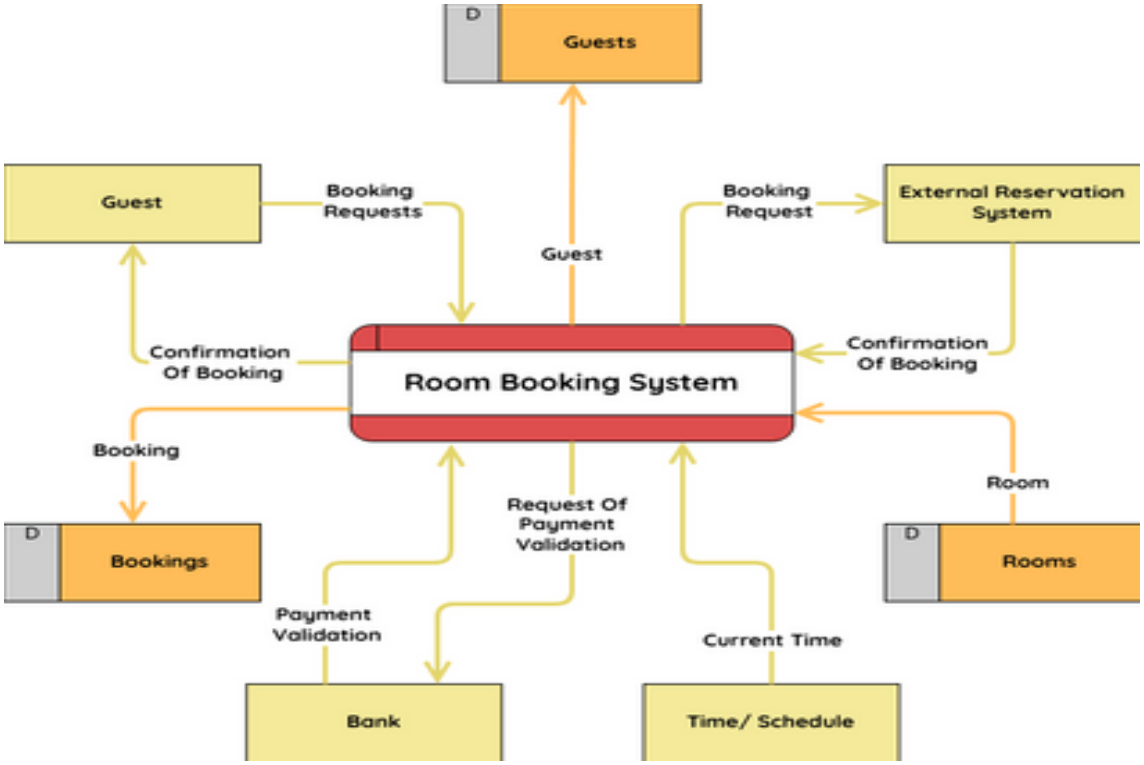

Figure 3.2: **Data Flow Diagram**

In figure 3.2 represents the data-flow diagram is a way of representing a flow of data through a process or a system. The Data flow diagram also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops.

3.3 Front End and Back End Design

3.3.1 Home Page

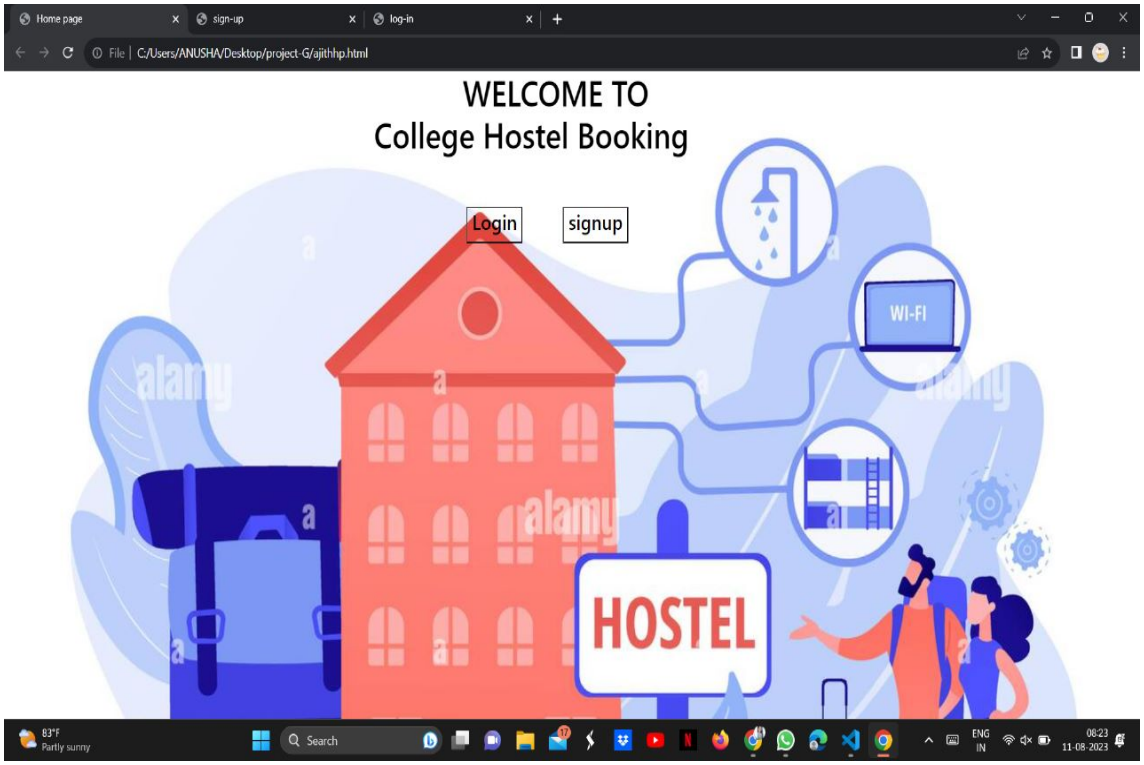

Figure 3.3: **Home Page**

In figure 3.3.1 represents the The Home Page serves as the entry point for users.Design includes an interactive search bar to find hostels, featured hostels, and attractive visuals. Hostel listings with images, descriptions, and availability status are displayed.

3.3.2 Signup and Login page

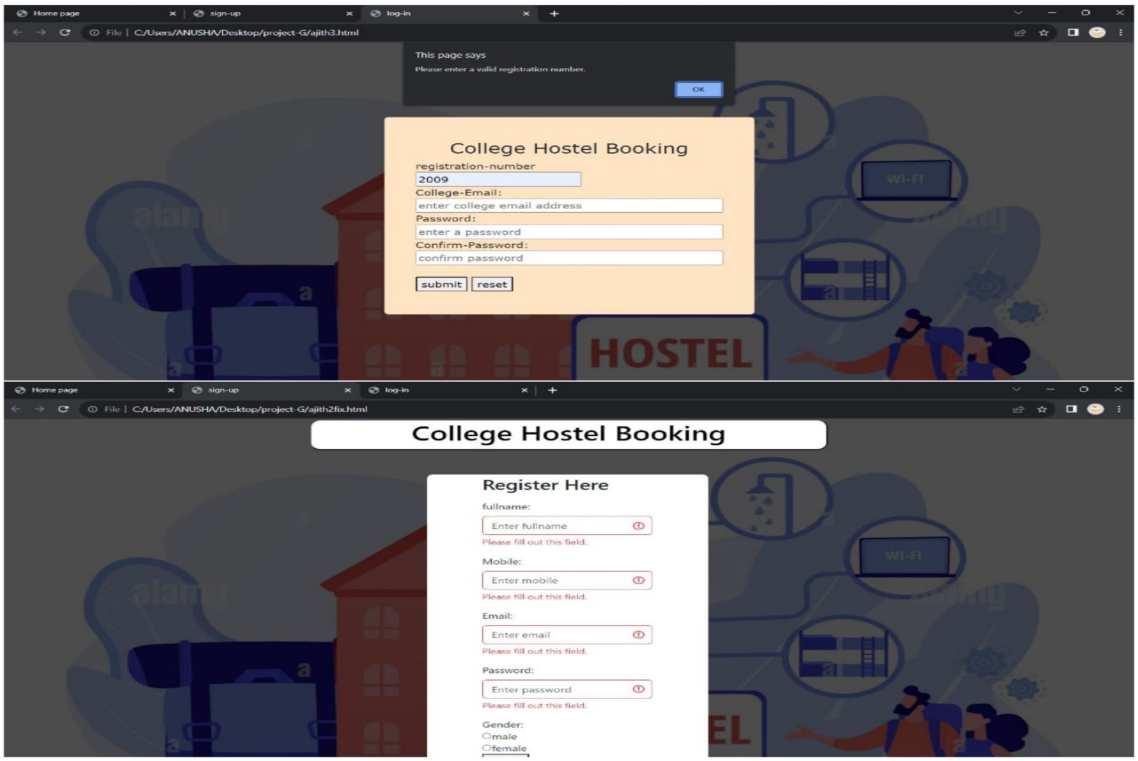

Figure 3.4: **Signup and Login page**

In figure 3.3.2 represents the Signup Page allows new users to create accounts. The Login Page provides authentication for registered users. Both pages feature clean and user-friendly forms with validation.

3.3.3 Form Validation

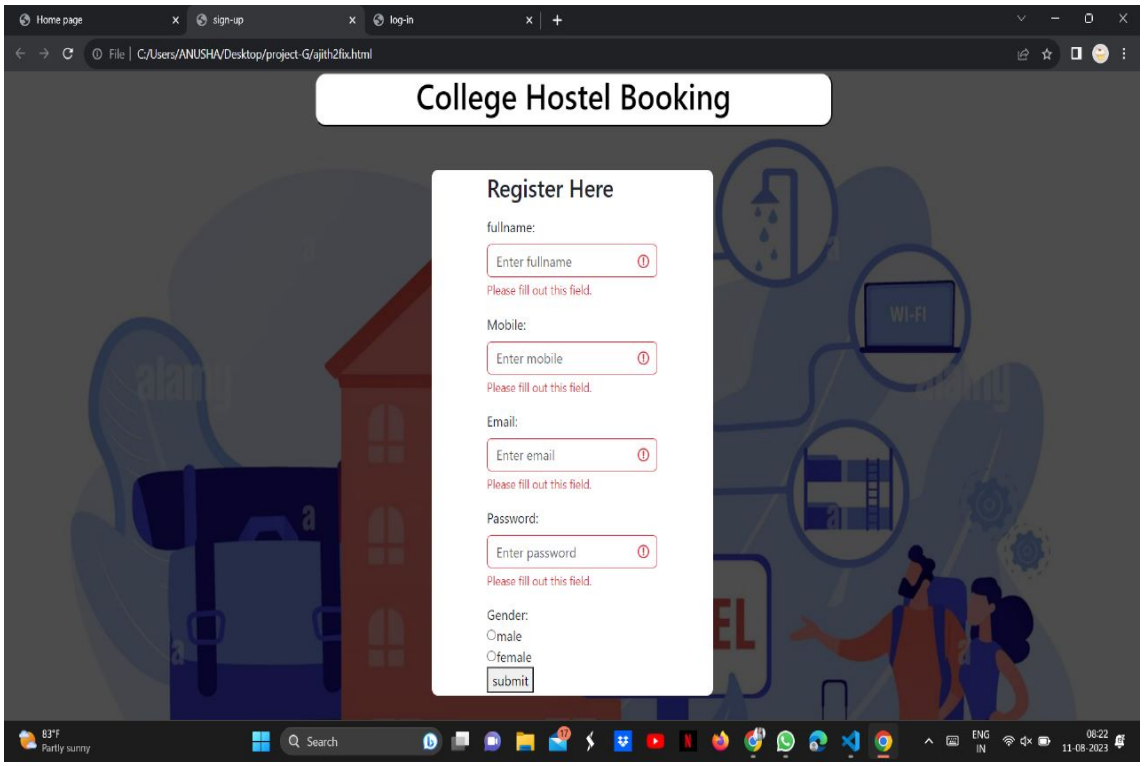

Figure 3.5: **Form Validation**

In figure 3.3,3 represents the Form validation is implemented using JavaScript/jQuery to ensure data accuracy. Validation rules include required fields, email format, password strength, etc. User-friendly error messages guide users in filling out forms correctly.

3.3.4 Parse the webpage using Jquery and DOM

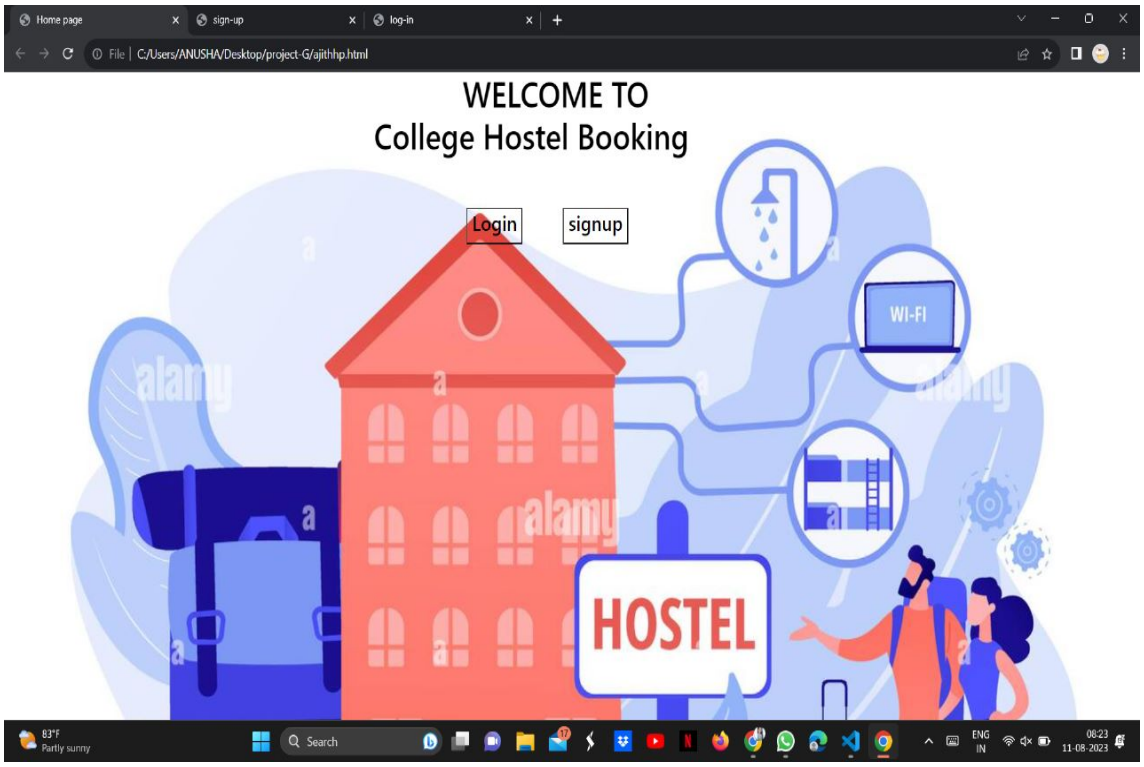

Figure 3.6: **Parse the webpage using Jquery and DOM**

In figure 3.3.4 represents the jQuery that is used to manipulate and interact with the Document Object Model (DOM). Dynamic content loading, filtering, and interactive features are achieved through jQuery.

3.3.5 Creation of Webserver using Node Js

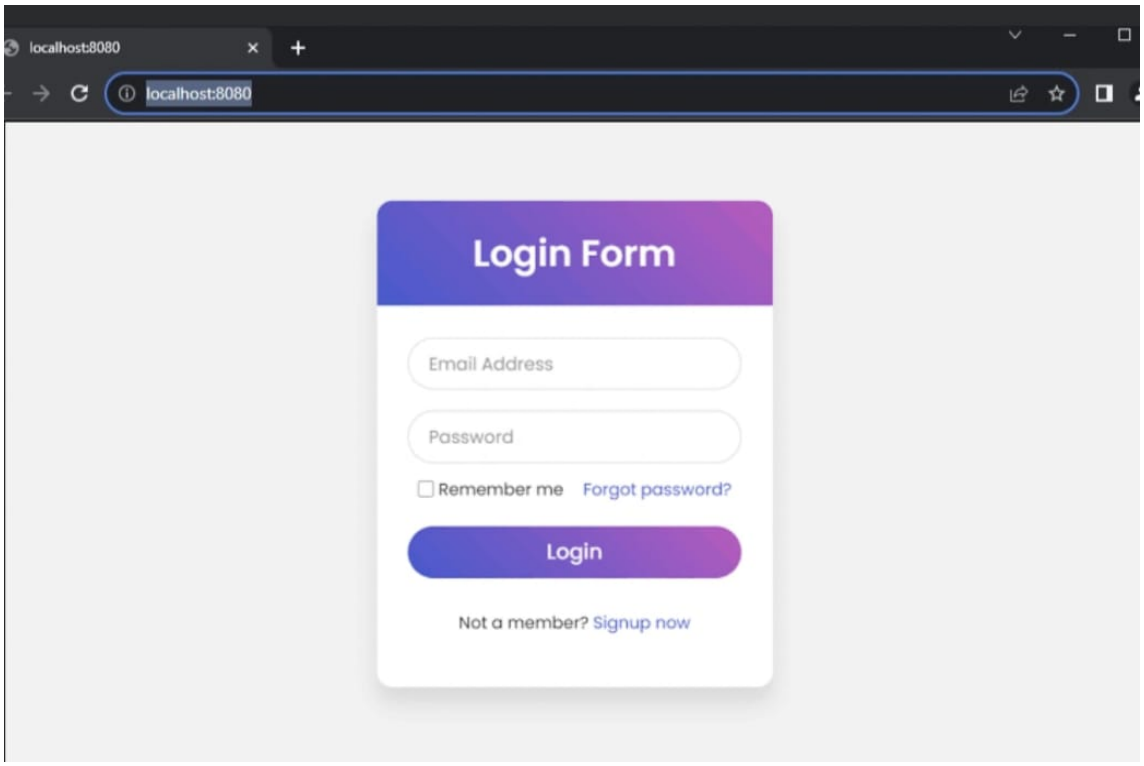

Figure 3.7: **Creation of Webserver using Node Js**

In figure 3.3.5 represents the Node.js that is used to create a web server for hosting the front-end application. Express.js, a Node.js framework, simplifies routing and middleware handling. The server serves HTML, CSS, JavaScript, and handles API requests.

3.3.6 Design of Three Tier application using Node js and MySQL

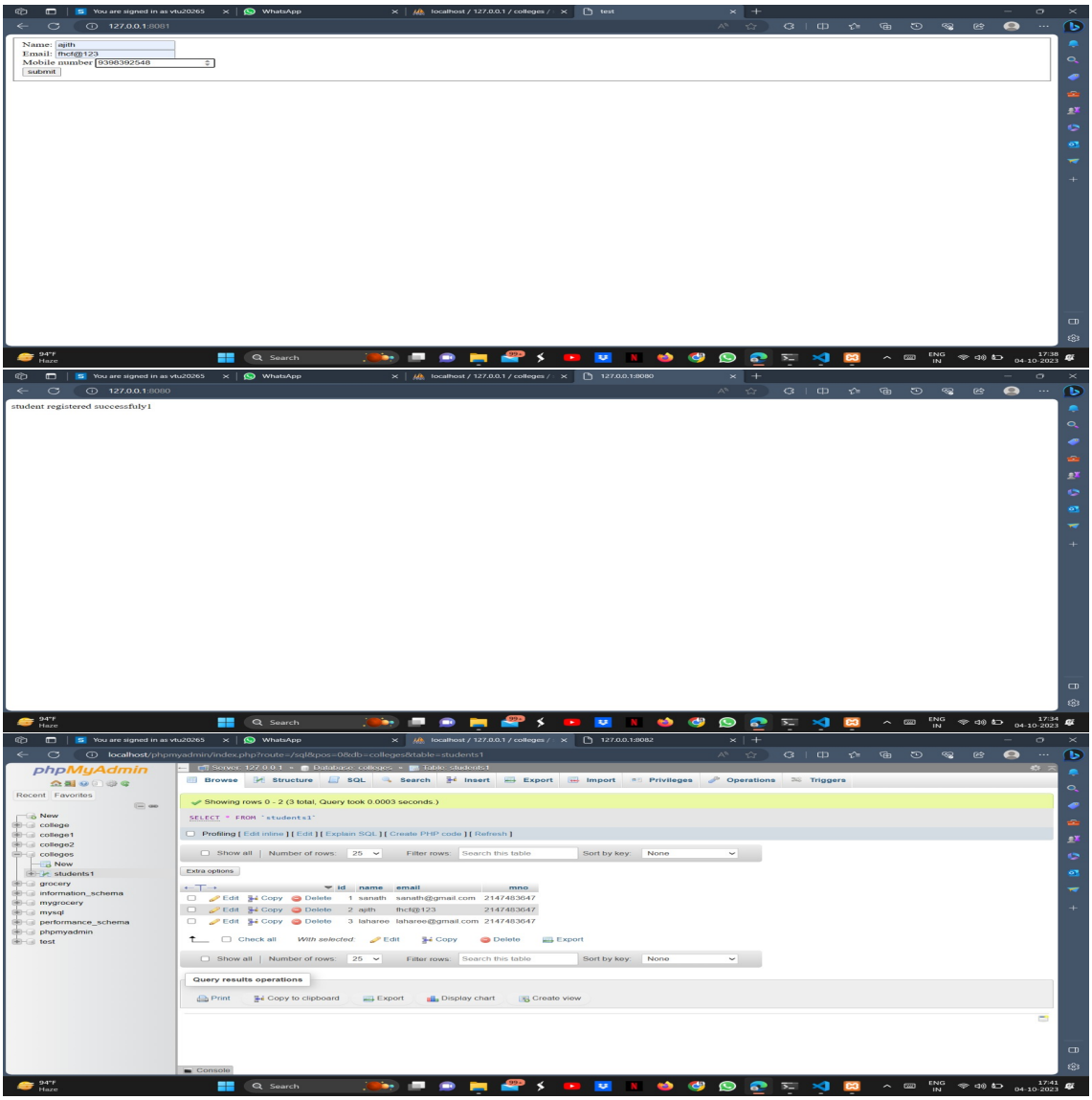

Figure 3.8: **Design of Three Tier application using Node js and MySQL**

In figure 3.3.6 represents the back-end follows a three-tier architecture: Presentation (Front-end), Logic (Node.js), and Data (MySQL). Node.js acts as the application server that communicates with the MySQL database. Database design includes tables for users, hostels, rooms, bookings, etc.

3.3.7 Design of Reactive form for User Registration using Angular

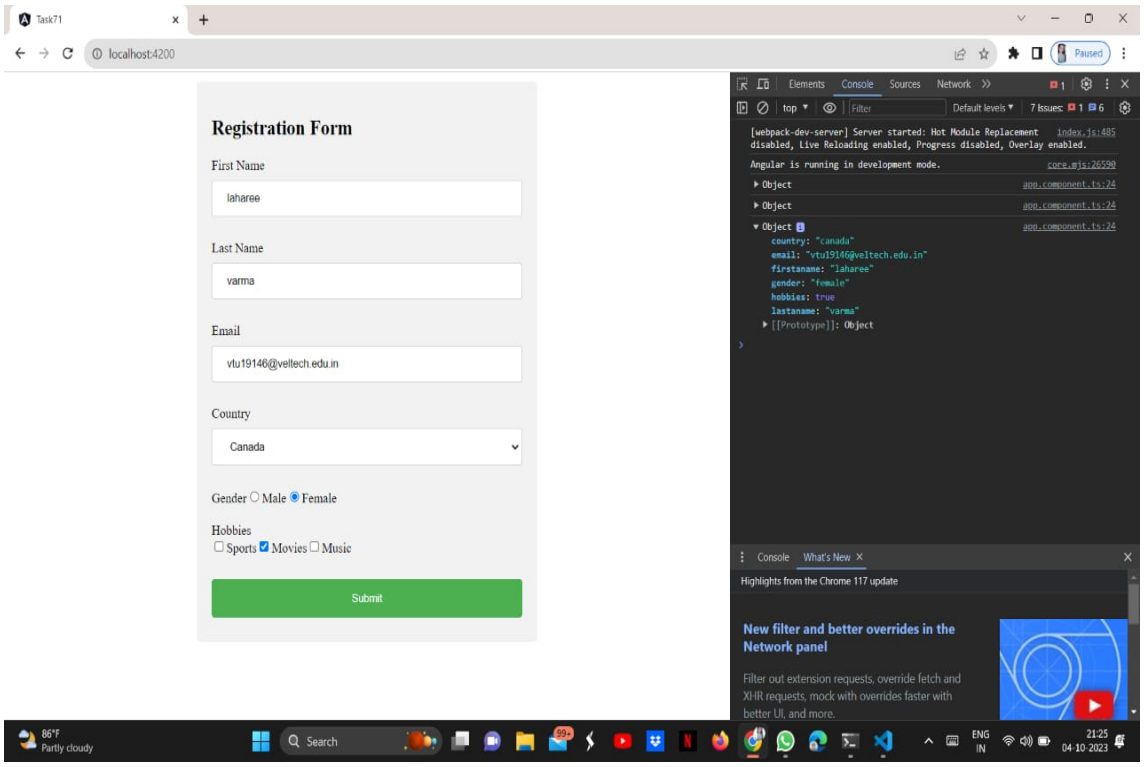

Figure 3.9: **Design of Reactive form for User Registration using Angular**

In figure 3.3.7 represents the Angular that is used to create a reactive form for user registration.Form components, validators, and error handling are implemented. Data binding ensures real-time updates between form fields and data model.

3.3.8 Develop web application to implement routing and navigation in Angular

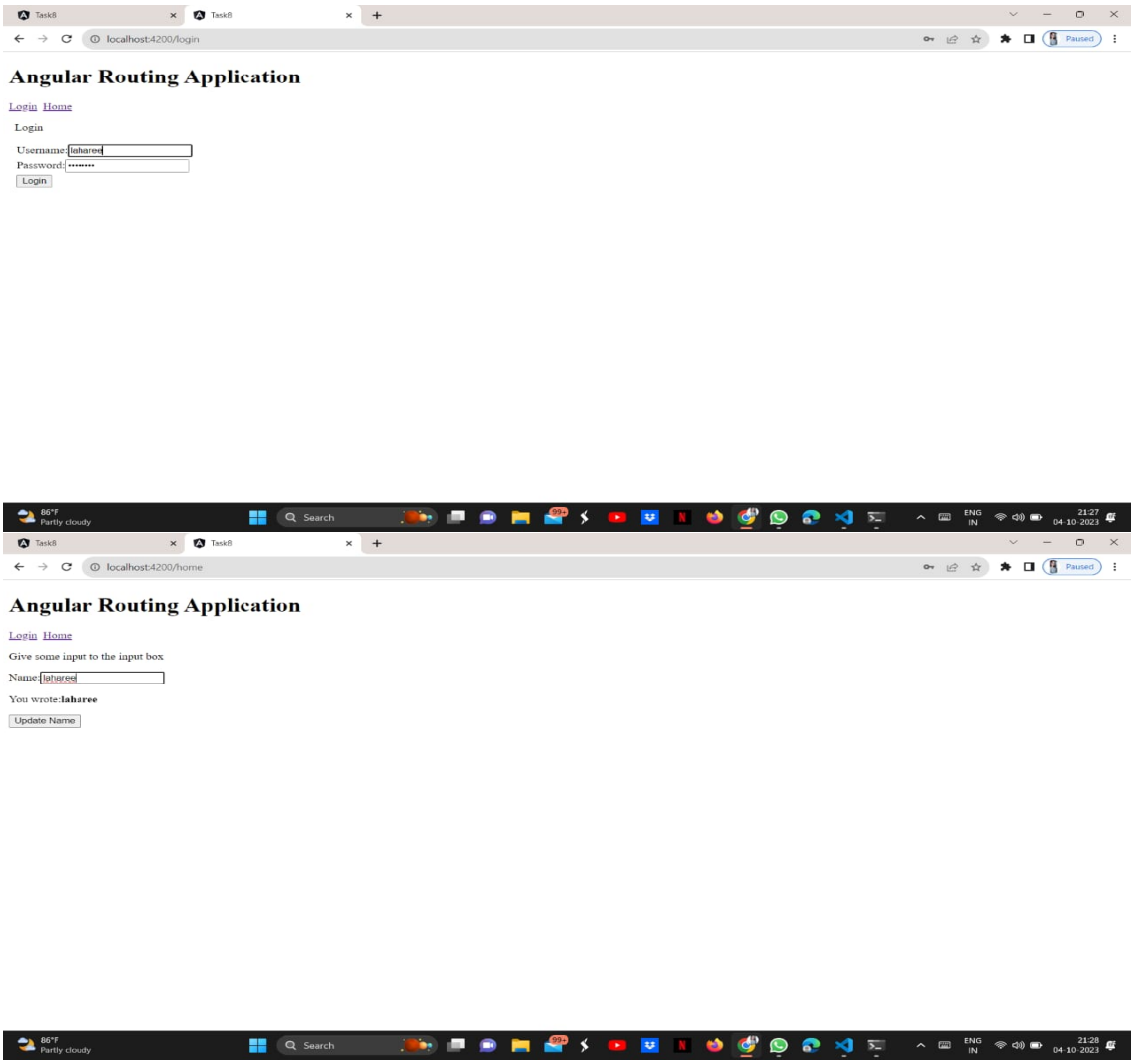

Figure 3.10: **Develop web application to implement routing and navigation in Angular**

In figure 3.3.8 represents the Angular's routing module that is employed for creating a single-page application (SPA). Routing and navigation allow users to move between pages without full page reloads. Hostel details, booking, and user profile pages are navigable.

3.3.9 Creation of Microservices

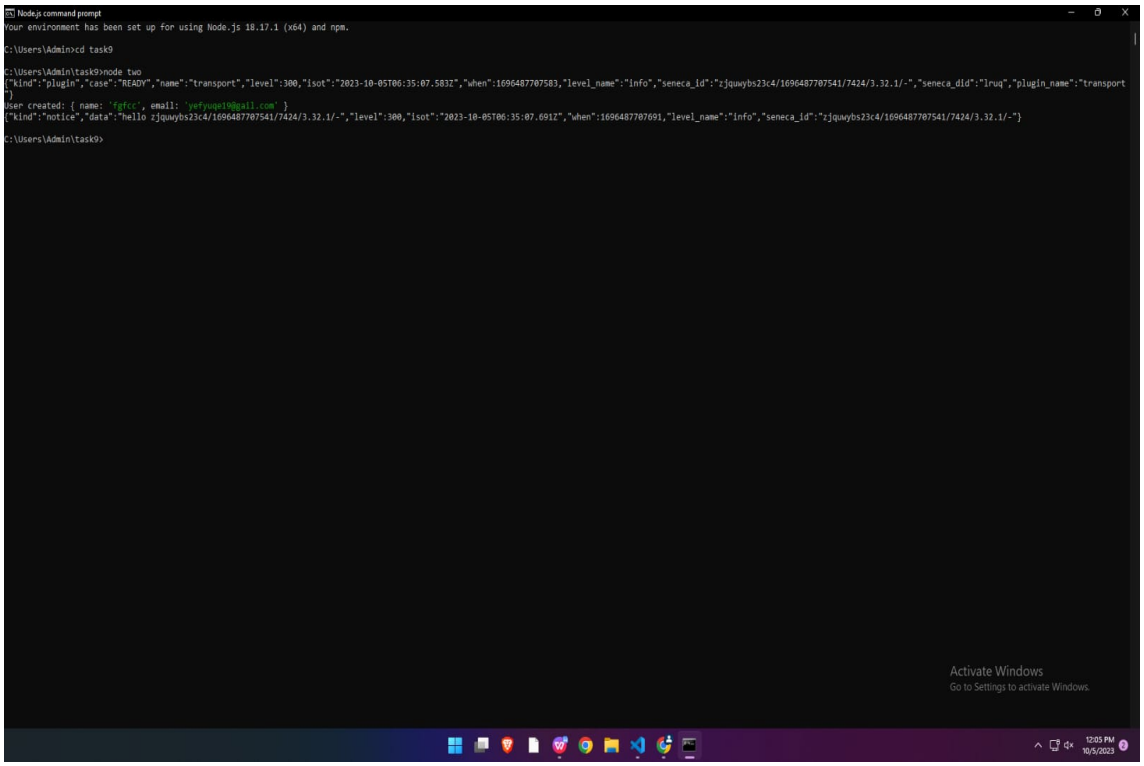

Figure 3.11: **Creation of Microservices**

In figure 3.3.9 represents the Microservices architecture is adopted to break down the application into smaller, independent services. Services may include user authentication, hostel listing, booking management, and payment processing. Each microservice has its database and API endpoints.

3.3.10 Converting web application into mobile application

Figure 3.12: **Converting web application into mobile application**

Chapter 4

TESTING

4.1 Testing

Test case ID	MOBILE	E-MAIL	PASSWORD
TC-1	9182593584	ajith@gmail.com	ajith@123
TC-2	45ggth22	ajith@gmail.com	ajith@3946
TC-3	9398392548	ajith.com	ajith@3946
TC-4	9398392548	ajith@gmail.com	ajith
TC-5	————-	————	———
TC-6	————-	ajith@gmail.com	ajith@3946
TC-7	9398392548	————-	ajith@3946
TC-8	9398392548	ajth@gmail.com	————-

Table 4.1: Test Cases

4.1.1 Test Result

Test Case ID	Test Cases	Input	Expected Result
TC-1	Valid Login	Mobile:valid, E-Mail:valid, Password:valid	User is successfully logged in and redirected to dashboard-/homepage.
TC-2	Invalid Mobile	Mobile:Invalid, E-Mail:valid, Password:valid	User sees and error message indicating that the mobile number is invalid.
TC-3	Invalid E-Mail	Mobile:valid, E-Mail:Invalid, Password:valid	User sees and error message indicating that the E-mail is invalid.
TC-4	Invalid Password	Mobile:valid, E-Mail:valid, Password:Invalid	User sees and error message indicating that the password is invalid.
TC-5	Empty Fields	Mobile:Empty, E-Mail:Empty, Password:Empty	User sees and error message indicating that all fields are required.
TC-6	Mobile Field Empty	Mobile:Empty, E-Mail:valid, Password:valid	User sees and error message indicating that the mobile number is required.
TC-7	E-Mail Field Empty	Mobile:valid, E-Mail:Empty, Password:valid	User sees and error message indicating that E-mail field is required.
TC-8	Password Field Empty	Mobile:valid, E-Mail:valid, Password:Empty	User sees and error message indicating that password is required

Table 4.2: Test Results

4.1.2 Test Bugs

1.Validation Bypass:Bug Description: Users can bypass client-side validation by manipulating HTML or JavaScript code. Impact:This could lead to the submission of invalid or malicious data to the server, potentially causing data integrity issues.

2. Cross-Site Scripting (XSS): Bug Description: Inadequate input sanitization allows users to inject malicious scripts into input fields. Impact: Attackers can execute scripts on other users' browsers, potentially stealing their session data or spreading malware.

3. Session Management Issues: Bug Description: The application fails to properly manage user sessions, leading to security vulnerabilities. Impact: Users may experience session hijacking or unauthorized access to other accounts.

4. Payment Gateway Integration Flaws: Bug Description: Improper integration with payment gateways may result in payment processing errors. Impact: Users could be charged incorrectly, leading

to financial disputes and user dissatisfaction.

5. Unsecured APIs: Bug Description: APIs used for backend communication lack proper authentication or authorization. Impact: Unauthorized users can access or manipulate data, potentially exposing sensitive information.

6. UI/UX Issues: Bug Description: Visual glitches, layout problems, or broken links can affect the user experience. Impact: Users may find it difficult to navigate the application or encounter frustration due to visual inconsistencies.

7. Database Vulnerabilities: Bug Description: Inadequate database security can lead to SQL injection vulnerabilities. Impact: Attackers can manipulate database queries to access, modify, or delete data, compromising system integrity.

8. Broken Links and Redirects: Bug Description:Links and redirects do not work as expected, leading to navigation issues. Impact:Users may be directed to non-existent pages or experience frustrating navigation problems.

9. Session Timeout Handling: Bug Description: Session timeouts may not be handled gracefully, leading to user confusion. Impact: Users may lose unsaved data or experience abrupt logouts without proper notification.

10. Incomplete or Missing Error Handling: Bug Description:Error messages may be missing or incomplete, making it challenging for users to understand and resolve issues. Impact:Users may encounter errors without clear guidance on how to proceed.

Chapter 5

WEBSITE LAUNCH

Figure 5.1: **Test Image**

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Website performance

6.1.1 Page Load Speed: Ensure that web pages load quickly to provide a seamless user experience. Optimize image sizes, use content delivery networks (CDNs), and minimize HTTP requests to reduce page load times.

6.1.2 Caching: Implement browser caching for static assets like CSS and JavaScript to reduce server load and speed up subsequent page loads.

6.1.3 Database Optimization: Optimize database queries and indexing to ensure efficient data retrieval and storage. Prevent database bottlenecks that can slow down the application.

6.1.4 Load Testing: Perform load testing to assess how the application handles concurrent users. Identify performance bottlenecks and scale the infrastructure accordingly.

6.1.5 Content Compression: Enable compression (e.g., GZIP) to reduce the size of data transferred between the server and clients, improving page load times.

6.2 Security

6.2.1 Data Encryption: Implement SSL/TLS encryption (HTTPS) to protect sensitive data during transmission, including user credentials and payment information.

6.2.2 Input Validation: Sanitize and validate user inputs to prevent common vulnerabilities like SQL injection and Cross-Site Scripting (XSS).

6.2.3 Authentication and Authorization: - Ensure robust user authentication and authorization mechanisms to prevent unauthorized access to sensitive areas and data.

6.2.4 Session Management: - Implement secure session management to prevent session hijacking and enforce session timeouts.

6.2.5 Payment Security: - Comply with Payment Card Industry Data Security Standard (PCI DSS) requirements when handling payment information. Use trusted payment gateways with strong security measures.

6.2.6 Security Updates: - Regularly update server software, frameworks, and libraries to patch

known security vulnerabilities.

6.3 Responsiveness and mobile-friendliness

6.3.1 Responsive Design: Ensure that the application's layout and content adapt gracefully to various screen sizes, from large desktop monitors to mobile devices, by using responsive web design techniques.

6.3.2 Mobile Testing: Thoroughly test the application on different mobile devices and browsers to verify that all functionalities work as expected.

6.3.3 Touch-Friendly Interface: Optimize user interface elements for touch interactions on mobile devices, such as larger buttons and intuitive gestures.

6.3.4 Performance on Slow Connections: Ensure that the application performs acceptably on slower mobile data connections by minimizing unnecessary data transfers and using efficient loading strategies.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

In conclusion, the hostel room booking project developed using HTML, CSS, Bootstrap framework, and JavaScript offers an efficient and user-friendly platform for booking hostel accommodations. The project's front-end design provides an intuitive and visually appealing interface for users to search for hostels, view room details, check availability, and make bookings. Meanwhile, the back-end design ensures data integrity, security, and seamless functionality.

Key features such as user registration, login, room booking, and payment processing have been successfully implemented. The application also focuses on security measures, including data encryption, input validation, and robust authentication, to protect user information and financial transactions.

Additionally, the project emphasizes responsiveness and mobile-friendliness, ensuring a consistent and enjoyable user experience across various devices and screen sizes. Performance optimizations, including page load speed and database efficiency, have been incorporated to enhance the application's speed and reliability.

7.2 Future Enhancements

1. User Reviews and Ratings: Implement a feature that allows users to leave reviews and ratings for hostels and rooms, providing valuable feedback for future guests.

2. Advanced Search Filters: Enhance the search functionality with advanced filters such as price range, room type, amenities, and user ratings.

3. Multi-Language Support: Add support for multiple languages to cater to a broader audience of international travelers.

4. Social Media Integration: Allow users to sign up and log in using their social media accounts, and enable sharing hostel information on social platforms.

5. Hostel Owner Dashboard: Create a dashboard for hostel owners/administrators to manage their

listings, update room availability, and respond to guest reviews.

6. Recommendation Engine: Implement a recommendation engine that suggests hostels and rooms based on user preferences and booking history.

7. Real-Time Chat Support: Integrate a real-time chat support system to assist users with inquiries and booking assistance.

8. Wishlist and Favorites: Allow users to create wishlists and mark their favorite hostels for future reference.

9. Data Analytics: Implement data analytics to gain insights into user behavior and booking patterns, helping in decision-making and marketing strategies.

10. Progressive Web App (PWA): Develop the application as a Progressive Web App to enable offline access and faster loading on mobile devices.

Chapter 8

SOURCE CODE

8.0.1 HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css"
    rel="stylesheet"
     integrity="sha384-4bw+/aepP/YC94hEpVNVgiZdgIC5
+VKNBQNGCHeKRQN+PtmoHDEXuppvnDJzQIu9"
     crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@
5.3.1/dist/js/bootstrap.bundle.min.js"
 integrity="sha384-
HwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmnlMuBnhbgrkm"
 crossorigin="anonymous"></script>
  <title>Home page</title>
  <style>
body{
          background-image: url('hos.jpg');
          background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;
       }
       .b2{
         margin-top: 50px;
         margin-left: 625px;
         background-color: transparent;
         border-color:black;
       }
       .b3{
       margin-top: 50px;
       margin-left: 50px;
       background-color: transparent;
       color: black;
       border-color:black;

      }
       .index{
         margin-left: 500px;
         color:black;
       }
  </style>
</head>
<body>
  <div class="index"><h1>   WELCOME TO <br>College Hostel
Booking</h1></div>


    <button class="b2"><a href="file:///C:/Users/ANUSHA/Desktop/project-G/ajith3.html"
style="text-decoration: none;color:black;" target="_blank"><h4>Login</h4></a></button>
    <button class="b3"><a href="file:///C:/Users/ANUSHA/Desktop/project-G/ajith2fix.html"
style="text-decoration: none;color: black;" target="_blank"><h4>signup</h4></a></button>
</body>
</html>
```

8.0.2 CSS

```
<!DOCTYPE html>
<html>
<head>
   <title>sign-up</title>
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css"
   rel="stylesheet"
    integrity="sha384-4bw+/aepP/YC94hEpVNVgiZdgIC5
+VKNBQNGCHeKRQN+PtmoHDEXuppvnDJzQIu9"
    crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@
5.3.1/dist/js/bootstrap.bundle.min.js"
 integrity="sha384-
HwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmnlMuBnhbgrkm"
 crossorigin="anonymous"></script>
  <style>
   body{
   background: linear-gradient(rgba(0,0,0,0.7),rgba(0, 0, 0,0.7)),url('hos.jpg');
   background-repeat: no-repeat;
   background-attachment: fixed;
 background-size: cover;
}
.button2{
   width:700px;
   border-radius: 15px;
   background-color: white;
}
.button1{
   padding-left: 420px;
}
.container{
   background-color: white;
   display: flex;
   justify-content: center;
   align-items:center;
   flex-wrap: wrap;
   width:380px;
   height: 600px;
   border-radius: 8px;
   margin-top: 50px;
}
  </style>
</head>
<body>
   <div class="button1">
```

```
  <div class="button1">
    <button class="button2"><h1>College Hostel Booking</h1></button>
  </div>
  <div class="container">
      <form  class="was-validated">
        <div class="index"><h3>Register Here</h3></div>
        <div class="mb-3 mt-3">
            <label for="fame" class="form-label">fullname:</label>
            <input type="text" class="form-control" id="fame" placeholder="Enter fullname"
name="fullname" required>
            <div class="valid-feedback">Valid.</div>
            <div class="invalid-feedback">Please fill out this field.</div>
          </div>
            <div class="mb-3 mt-3">
          <label for="mobile" class="form-label">Mobile:</label>
            <input type="number" class="form-control" id="mobile" placeholder="Enter
mobile" name="mobile" required>
            <div class="valid-feedback">Valid.</div>
            <div class="invalid-feedback">Please fill out this field.</div>
          </div>
          <div class="mb-3 mt-3">
            <label for="email" class="form-label">Email:</label>
            <input type="text" class="form-control" id="email" placeholder="Enter email"
name="email" required>
            <div class="valid-feedback">Valid.</div>
            <div class="invalid-feedback">Please fill out this field.</div>
          </div>
          <div class="mb-3">
            <label for="pwd"  class="form-label">Password:</label>
            <input type="password" class="form-control" id="pwd" placeholder="Enter
password" name="pswd" required>
            <div class="valid-feedback">Valid.</div>
            <div class="invalid-feedback">Please fill out this field.</div>
          </div>
        Gender:<br>
        <input type="radio" name="option1" value="male">male<br>
        <input type="radio" name="option1" value="female">female<br>
         <div class="button3">
            <button>submit </button>
         </div>
        </form>
         </div>
</body>
</html>
```

8.0.3 JS

```
<!DOCTYPE html>
<html>
<head>
   <title>sign-up</title>
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css"
   rel="stylesheet"
    integrity="sha384-4bw+/aepP/YC94hEpVNVgiZdgIC5
+VKNBQNGCHeKRQN+PtmoHDEXuppvnDJzQIu9"
    crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@
5.3.1/dist/js/bootstrap.bundle.min.js"
 integrity="sha384-
HwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmnlMuBnhbgrkm"
 crossorigin="anonymous"></script>
  <style>
   body{
   background: linear-gradient(rgba(0,0,0,0.7),rgba(0, 0, 0,0.7)),url('hos.jpg');
   background-repeat: no-repeat;
   background-attachment: fixed;
 background-size: cover;
}
.button2{
   width:700px;
   border-radius: 15px;
   background-color: white;
}
.button1{
   padding-left: 420px;
}
.container{
   background-color: white;
   display: flex;
   justify-content: center;
   align-items:center;
   flex-wrap: wrap;
   width:380px;
   height: 600px;
   border-radius: 8px;
   margin-top: 50px;
}
  </style>
</head>
<body>
   <div class="button1">
      <button class="button2"><h1>College Hostel Booking</h1></button>
```

```
    <div class="button1">
      <button class="button2"><h1>College Hostel Booking</h1></button>
    </div>
    <div class="container">
        <form  class="was-validated">
          <div class="index"><h3>Register Here</h3></div>
          <div class="mb-3 mt-3">
                <label for="fame" class="form-label">fullname:</label>
                <input type="text" class="form-control" id="fame" placeholder="Enter fullname"
name="fullname" required>
                <div class="valid-feedback">Valid.</div>
                <div class="invalid-feedback">Please fill out this field.</div>
               </div>
                <div class="mb-3 mt-3">
              <label for="mobile" class="form-label">Mobile:</label>
              <input type="number" class="form-control" id="mobile" placeholder="Enter
mobile" name="mobile" required>
              <div class="valid-feedback">Valid.</div>
              <div class="invalid-feedback">Please fill out this field.</div>
             </div>
             <div class="mb-3 mt-3">
              <label for="email" class="form-label">Email:</label>
              <input type="text" class="form-control" id="email" placeholder="Enter email"
name="email" required>
              <div class="valid-feedback">Valid.</div>
              <div class="invalid-feedback">Please fill out this field.</div>
             </div>
             <div class="mb-3">
              <label for="pwd"  class="form-label">Password:</label>
              <input type="password" class="form-control" id="pwd" placeholder="Enter
password" name="pswd" required>
              <div class="valid-feedback">Valid.</div>
              <div class="invalid-feedback">Please fill out this field.</div>
             </div>
           Gender:<br>
           <input type="radio" name="option1" value="male">male<br>
           <input type="radio" name="option1" value="female">female<br>
            <div class="button3">
              <button>submit </button>
            </div>
         </form>
          </div>
</body>
</html>
```

Chapter 9

SCREENSHOTS

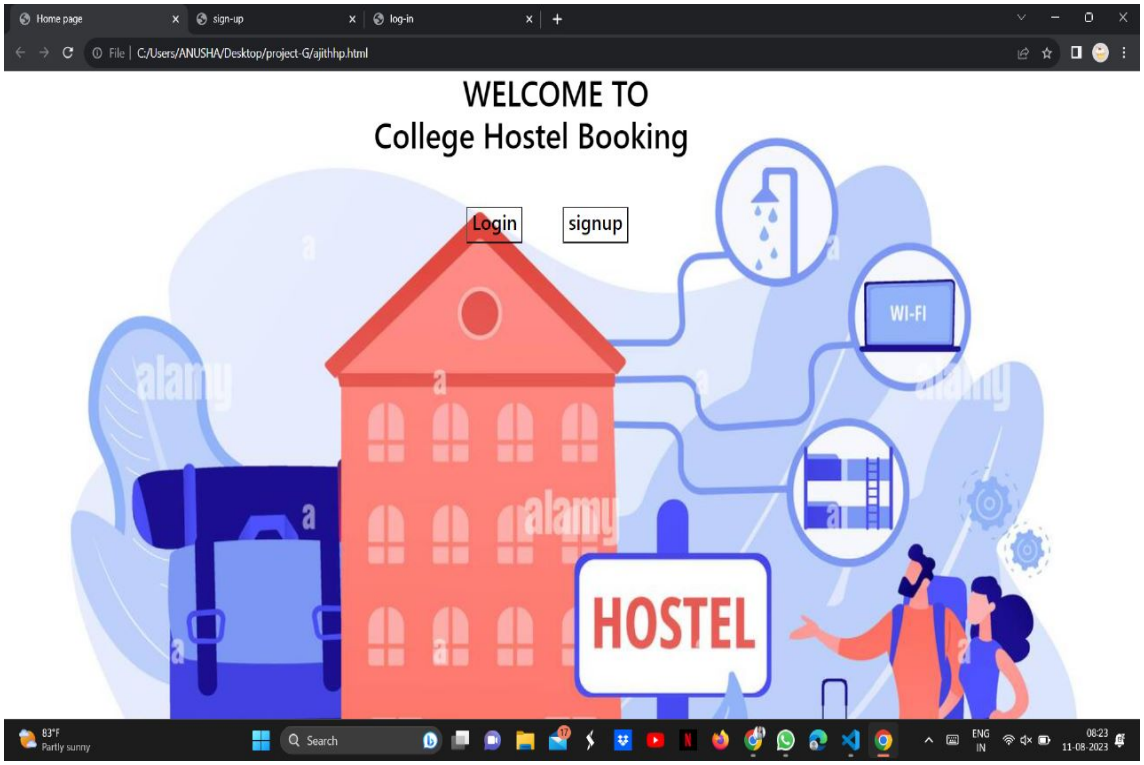

Figure 9.1: **Home Page**

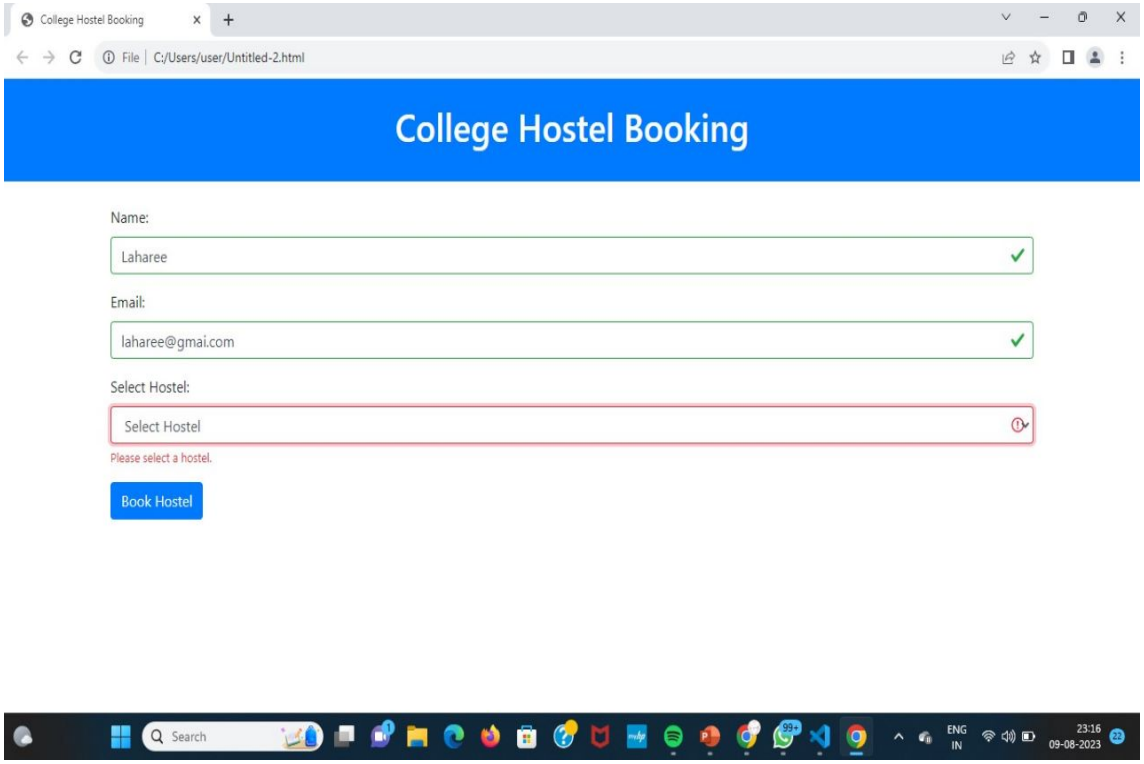

Figure 9.2: **Login Page**

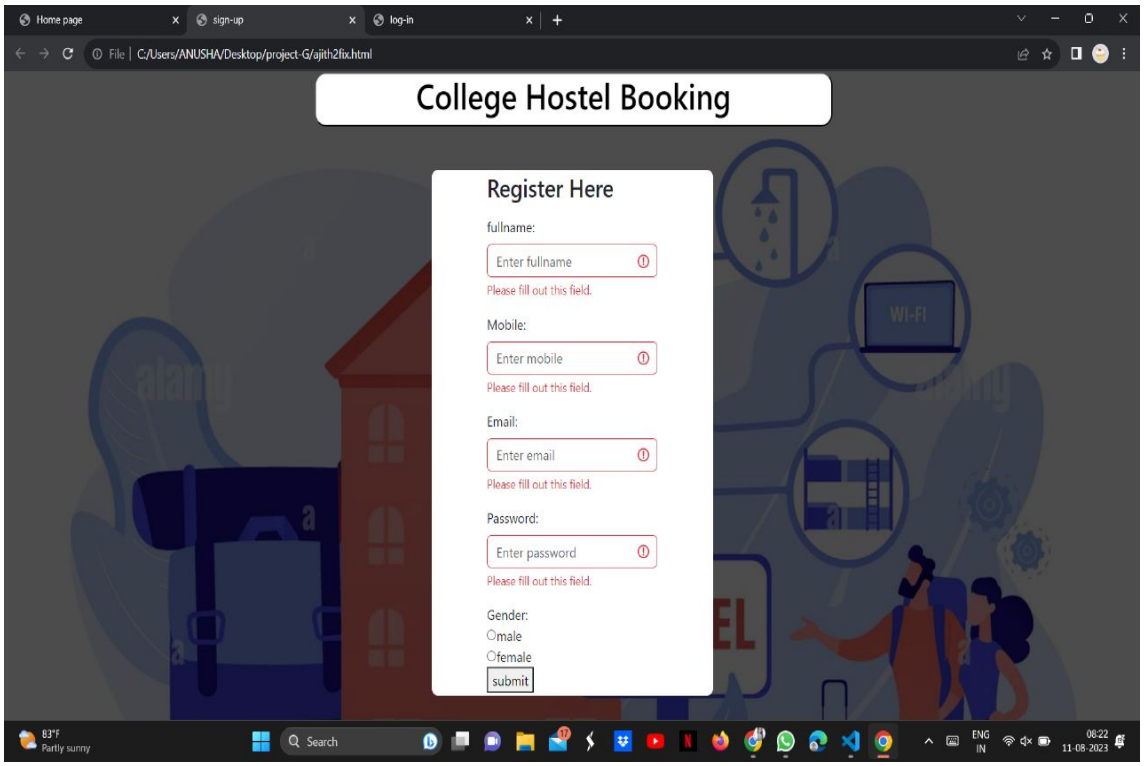

Figure 9.3: **Sign-up page**

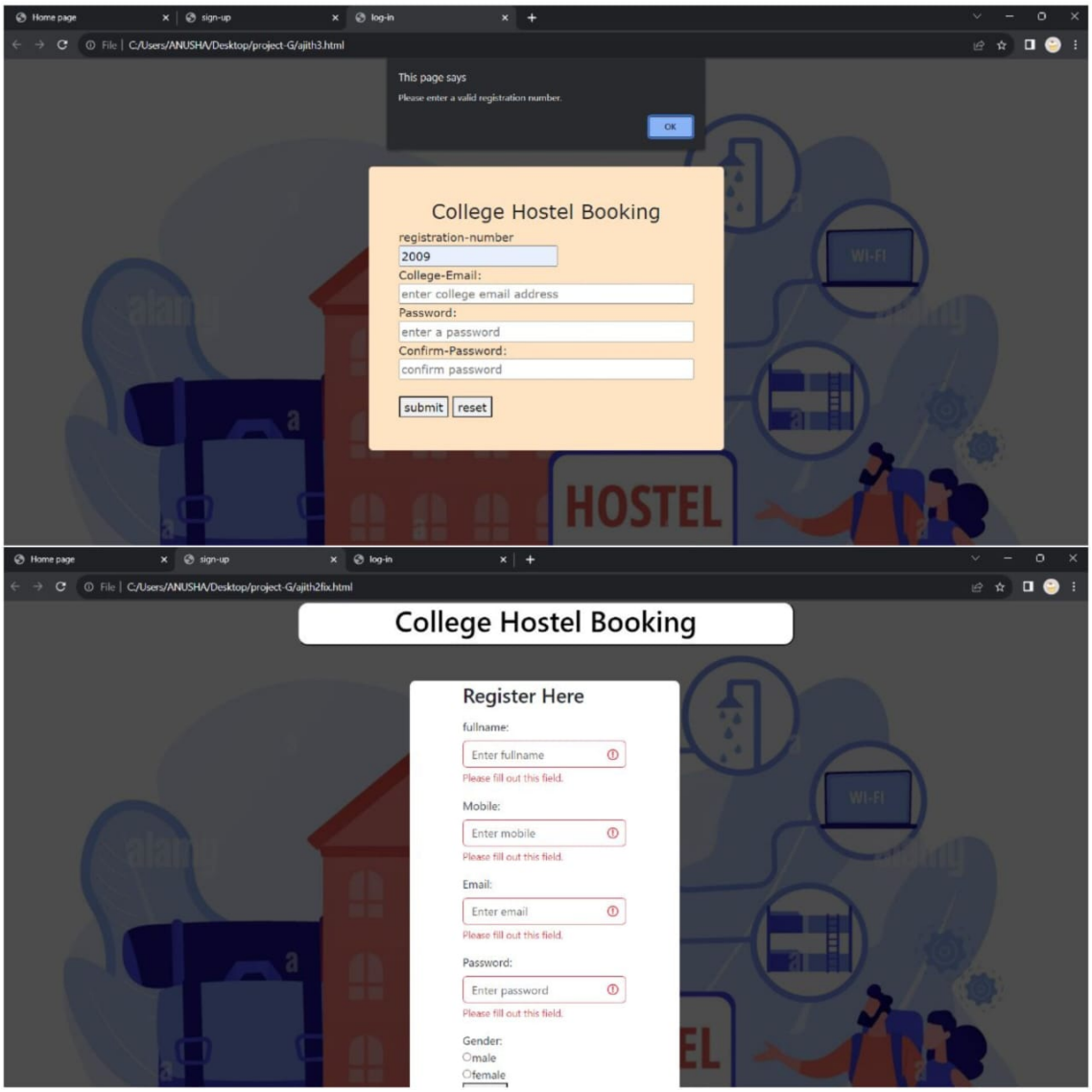

Figure 9.4: **Login and Sign-up page**