SMART ROOM ACCOMMODATION AND TRACKING SYSTEM

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

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Under the esteemed guidance of

Mrs S. Radha
Sr Assistant Professor

By
ABBURI BHAVYA SRI (21R11A05A5)
KOSURU BHARATH KUMAR (21R11A05C8)
BUSHOLLA SRINATH (21R11A05B5)

Department of Computer Science and Engineering
Accredited by NBA

Geethanjali College of Engineering and Technology
(UGC Autonomous)
(Affiliated to J.N.T.U.H, Approved by AICTE, New Delhi)
Cheeryal (V), Keesara (M), Medchal.Dist.-501 301.

August-2024

Geethanjali College of Engineering & Technology

(UGC Autonomous)

(Affiliated to JNTUH, Approved by AICTE, New Delhi)
Cheeryal (V), Keesara(M), Medchal Dist.-501 301.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Accredited by NBA

CERTIFICATE

This is to certify that the B.Tech Mini Project report entitled **“SMART ROOM ACCOMADATION AND TRACKING SYSTEM”** is a bonafide work done by **Abburi Bhavya Sri (21R11A05A5), Kosuru Bharath Kumar (21R11A05C8), Busholla Srinath (21R11A05B5)**, in partial fulfillment of the requirement of the award for the degree of Bachelor of Technology in “**Computer Science and Engineering**” from Jawaharlal Nehru Technological University, Hyderabad during the year 2023-2024.

Internal Guide	HOD - CSE
Mrs S. Radha	**Dr A SreeLakshmi**
Sr Assistant Professor	Professor

External Examiner

Geethanjali College of Engineering & Technology

(UGC Autonomous)

(Affiliated to JNTUH Approved by AICTE, New Delhi)
Cheeryal (V), Keesara(M), Medchal Dist.-501 301.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Accredited by NBA

DECLARATION BY THE CANDIDATE

We, **Abburi Bhavya Sri, Kosuru Bharath Kumar, Boshulla Srinath,** bearing Roll Nos. **21R11A05A5, 21R11A05C8, 21R11A05B5** hereby declare that the project report entitled **"Smart Room Accommodation and Tracking System"** is done under the guidance of **Mrs. S. Radha**, Sr **Assistant Professor**, Department of Computer Science and Engineering, Geethanjali College of Engineering and Technology, is submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering**.

This is a record of bonafide work carried out by me/us in **Geethanjali College of Engineering and Technology** and the results embodied in this project have not been reproduced or copied from any source. The results embodied in this project report have not been submitted to any other University or Institute for the award of any other degree or diploma.

Abburi Bhavya Sri(21R11A05A5)

Kosuru Bharath Kumar(21R11A05C8)

Boshulla Srinath(21R11A05B5)

Department of CSE,

Geethanjali College of Engineering and Technology,

Cheeryal.

ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr. A. Sree Lakshmi, Professor, Head of Department of Computer Science,** Geethanjali College of Engineering and Technology, Cheeryal, whose motivation in the field of software development has made us to overcome all hardships during study and successful completion of project.

We would like to express our profound sense of gratitude to all for having helped us in completing this dissertation. We would like to express our deep-felt gratitude and sincere thanks to our guide **S. Radha, Sr Assistant Professor,** Department of Computer Science, Geethanjali College of Engineering and Technology, Cheeryal, for her skillful guidance, timely suggestions, and encouragement in completing this project successfully.

We would like to express our sincere gratitude to our **Principal Prof. Dr. S. Udaya Kumar** for providing the necessary infrastructure to complete our project. We are also thankful to our Chairman **Mr. G. R. Ravinder Reddy** for providing an interdisciplinary and progressive environment.

Finally, we would like to express our heartfelt thanks to our parents who were very supportive both mentally and financially for their encouragement to achieve our set goals.

Abburi Bhavya Sri(21R11A05A5)

Kosuru Bharath Kumar(21R11A05C8)

Boshulla Srinath(21R11A05B5)

ABSTRACT

Hostels play a crucial role in providing accommodations for students, faculty, and staff in educational institutions. Effective management of hostel facilities is essential to ensure optimal resource utilization, comfortable living conditions, and efficient operations. The Hostel Management System is designed to streamline room allocation, occupancy tracking, and resource planning processes. At its core, the system maintains a detailed database of the hostel's physical infrastructure, including the number of floors, rooms, and their respective dimensions. This information serves as the foundation for intelligent room allocation algorithms. The room allocation module employs advanced optimization techniques to assign rooms based on predefined criteria, such as occupant preferences, room size, and occupancy limits. The system dynamically updates the room occupancy status, using a color-coded system to provide a clear visual representation: green for vacant rooms, ash for fully occupied rooms, and other colors for partially occupied rooms. One of the key features of the system is the ability to accept advance bookings for rooms. The system intelligently manages these reservations, preventing double bookings and optimizing room utilization. Furthermore, the system incorporates robust reporting and analytics capabilities, enabling hostel administrators to generate detailed reports on occupancy rates, room utilization, and resource allocation. These insights facilitate data-driven decision-making and informed planning for maintenance or renovation. The system's user-friendly web-based interface provides seamless access to hostel residents, allowing them to view room details, occupancy status, and make reservations or report maintenance issues. Administrators can easily manage room assignments, update occupancy records, and monitor overall hostel operations through an intuitive dashboard.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Acronym	Abbreviations
CPU	Central Processing Unit
CSS	Cascading Style Sheet
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
JS	JavaScript
OS	Operating System
PHP	Hypertext Preprocessor
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator
XAMPP	Cross platform, Apache, MySQL, PHP, Pearl

TABLE OF CONTENTS

1. INTRODUCTION

1.1 About the project

Hostels play a crucial role in providing essential living accommodations for students, faculty, and staff within and around educational institutions. For those joining Geetanjali College of Engineering and Technology, finding suitable nearby hostels, and checking for vacancy can be a discouraging task. The lack of an efficient system often results in confusion and delays, making the transition to college life more challenging.

In addition to the difficulties in finding accommodations, managing hostel facilities effectively presents several challenges, including optimizing room allocation and tracking occupancy status. Traditional methods of managing hostels often rely on manual processes or outdated software systems that may be prone to errors, inefficiencies, and delays. These shortcomings can lead to issues such as double bookings, underutilized or overcrowded rooms, and a lack of transparency for both administrators and residents.

The motivation behind the Smart Room Accommodation and Tracking System project is to overcome these challenges by developing an innovative solution that simplifies the process of finding and managing hostel accommodations. This system aims to provide a reliable, user-friendly platform that enhances the efficiency of hostel operations and improves the overall experience for both administrators and residents.

1.2 Objective

1. **Real-Time Occupancy Tracking**: Implement a dynamic system that updates room occupancy status in real time. This feature will enable both administrators and residents to easily view the current status of rooms—whether they are vacant, fully occupied, or partially occupied—thus ensuring transparency and accurate tracking.
2. **Optimize Room Allocation**: Utilize advanced algorithms to allocate rooms based on predefined criteria such as room size, occupant preferences, and availability. This optimization ensures that all available space is used efficiently, reducing the chances of overbooking or underutilization.
3. **Prevent Double Bookings and Conflicts**: Introduce mechanisms to manage advance bookings and prevent double bookings. The system will automatically check for conflicts and adjust room allocations accordingly, ensuring a smooth booking process.
4. **Ensure Security and Privacy**: Implement secure login systems and data encryption to protect the personal information of users. The system will ensure that sensitive data, such as room assignments and booking details, are accessible only to authorized users.
5. **Improve User Experience with a Web-Based Interface:** Develop a user-friendly, web-based interface that provides seamless access to all functionalities, including room search, booking, and status updates. The interface will be designed to be intuitive, allowing users to navigate and use the system with minimal guidance.
6. **Promote Sustainability and Efficiency**: By reducing the reliance on paper-based processes and optimizing resource allocation, the system aims to promote sustainability and operational efficiency within the hostel management process.

2. SYSTEM ANALYSIS

2.1 Existing System

The existing hostel management systems offer essential features for managing bookings and occupancy, but they also come with certain limitations. **ResDiary** is one of the commonly used systems that focuses on providing real-time updates on room availability and occupancy status. It allows hostel managers and guests to see the exact status of rooms at any given time. Additionally, ResDiary has a feature for advanced booking management that automates the booking process, ensuring that rooms are automatically updated and confirmed in real-time. One of its more advanced features is dynamic pricing, which adjusts room rates based on demand, making it responsive to market trends. However, this flexibility can sometimes result in fluctuating prices that can be unpredictable for users.

On the other hand, **Cloudbeds** provides a unified dashboard that consolidates data on bookings, occupancy rates, and revenue, allowing hostel managers to monitor operations at a glance. This system automates room allocation based on real-time availability, streamlining the process of assigning rooms to guests. Cloudbeds also integrates with multiple online booking platforms, ensuring that availability and rates are consistent across all channels. Additionally, the system includes a guest management feature that handles check-ins, check-outs, and guest profiles in real-time, improving the overall guest experience. Despite these advantages, Cloudbeds' reliance on real-time data and system integrations can sometimes result in synchronization failures.

Hostelworld serves a slightly different market, emphasizing user-friendliness and customer reviews. It offers real-time room availability for hostel beds and rooms, along with secure booking management, which is essential for both hostelers and managers. User reviews and ratings are a major feature of Hostelworld, helping new customers make decisions based on previous guest experiences. In addition to its desktop platform, Hostelworld provides a mobile app, making it easier for users to book, manage, and review their stays from their phones. However, the system can be limited by biased or inaccurate reviews, and the room availability updates on Hostelworld may not be as robust, leading to potential issues like overbooking.

Disadvantages of the Existing System:

1. **ResDiary**: The dynamic pricing feature may cause frequent price changes, leading to frustration among guests who expect stable rates. The system also requires significant technical maintenance for real-time data and integration with other platforms.
2. **Cloudbeds**: Synchronization issues due to real-time data reliance can result in problems such as overbooking or displaying outdated information to guests, which impacts guest satisfaction and hostel operations.
3. **Hostelworld**: Since it heavily depends on user feedback, biased or unverified reviews may skew perceptions, affecting hostel bookings. Additionally, its room availability updates are not as frequent, leading to potential overbooking during peak times.

2.2 Proposed System

The **Smart Room Accommodation and Tracking System** is designed to overcome many of the limitations present in existing hostel management platforms. At its core, this system focuses on room and bed management, organizing availability, allocation, and booking status with real-time updates. This allows hostel managers and hostelers alike to have a clear view of available spaces, ensuring smooth room assignment.

A standout feature of this system is its approach to **Hostelers Management**, where check-ins, check-outs, and profiles are handled seamlessly in real-time. This ensures that hostel managers can always keep track of who is in the hostel and manage occupancy efficiently. Additionally, the system introduces a **color-coding** scheme, which visually displays the status of room occupancy. This makes it easy for managers to quickly assess which rooms are occupied, vacant, or under maintenance at a glance. The real-time status updates combined with these visual cues make day-to-day management significantly easier.

Furthermore, the system features **automated room allocation**, which assigns rooms based on availability and hosteler preferences, ensuring that every guest's needs are met. Along with **real-time room availability**, this feature provides up-to-date information on occupancy status, allowing for efficient room assignments and preventing overbooking. By integrating these real-time updates and automated processes, the Smart Room Accommodation and Tracking System enhances both operational efficiency and user experience.

Advantages of the proposed System:

1. **Room and Bed Management**: Organizes room and bed availability, allocation, and booking status.

2. **Mobile Access**: Provides mobile-friendly interfaces for both admin and users to manage bookings and check-ins.

3. **Colour coding:** Shows the status of the room occupancy.

4. **Automated Room Allocation**: Assigns rooms based on availability and hostelers preferences, with real-time updates.

5. **Real-Time Room Availability**: Displays up-to-date information on room availability and occupancy status.

2.3 Feasibility Study

1. Details

1.Technical Feasibility

The **Smart Room Accommodation and Tracking System** will be developed using a well-established technology stack to ensure reliability and ease of development. For the frontend, HTML, CSS, and JavaScript will be used to create a responsive and intuitive user interface that works seamlessly across different devices. The backend will be powered by PHP, providing server-side functionality to manage data processing and application logic. The system will use MySQL as the database for data storage, which is ideal for handling the large volumes of data typically associated with hostel management, such as bookings, hosteler profiles, and room availability. Together, this technology stack offers a scalable and cost-effective solution that meets the system's technical requirements.

2.Operational Feasibility

To ensure **user-friendliness**, the system will be designed to be intuitive and easy to use for all stakeholders, including students, hostel managers, and administrators. A simple, well-organized interface with clear navigation will be critical for user adoption, making it easier for users to quickly learn how to use the system. Additionally, training sessions or user manuals will be provided to guide users through the features and functionalities, ensuring a smooth transition from manual or outdated processes. The system will replace or enhance current paper-based records, spreadsheets, and other outdated software, integrating all essential hostel management functions. This includes room booking, check-ins, check-outs, maintenance requests, and reporting, making the entire process more efficient. A key component of user acceptance will involve conducting pilot tests and gathering feedback from a select group of users to ensure the system aligns with their needs. Addressing concerns about change management and highlighting the benefits and ease of use will also be essential in overcoming any resistance to new technology.

3.Benefits:

The system offers several benefits, the first of which is **improved efficiency**. By automating room allocation, tracking, and other administrative tasks, the system will significantly reduce the time spent on manual processes. This automation will also minimize errors related to manual record-keeping, such as double bookings or incorrect data entries. Another key benefit is **enhanced transparency and communication**, as the system will provide real-time updates on room availability, payments, and other essential information, ensuring all stakeholders have access to up-to-date data. The built-in messaging and notification features will facilitate better communication between hostelers and administrators. Additionally, the system will help optimize resource utilization, allowing better management of hostel capacities to prevent underutilized or overcrowded rooms. This data-driven approach will enable better decision-making regarding room assignments and resource allocation. The **improved user experience** will make it easier for students and staff to find and secure accommodations, while the centralized platform will streamline all hostel-related activities. Finally, the system will enhance **security and compliance** by providing secure login mechanisms and safeguarding sensitive data, ensuring compliance with data privacy regulations.

4.Economic Feasibility:

From an economic standpoint, the system has the potential to generate revenue by optimizing room utilization and reducing vacancies. Ensuring that residents make timely payments can further boost financial efficiency. Additionally, the system could manage and bill for additional services, such as premium room options or meal plans, creating new revenue streams. In terms of cost analysis, initial **development costs** will include the investment required for software development, hardware procurement, and integration with existing systems. **Operational costs** will include ongoing expenses such as server hosting, system maintenance, software updates, and technical support. There will also be **training costs** associated with educating staff and users on how to effectively operate the new system. Despite these expenses, the system is expected to offer long-term financial benefits by improving operational efficiency and generating new revenue opportunities.

2. Ethics

The **Smart Room Accommodation and Tracking System** adheres to the highest standards of software ethics, prioritizing transparency and responsible use of technology. Throughout its development, the platform ensures the confidentiality, privacy, and security of user data by implementing robust safeguards. It clearly communicates how data will be collected, stored, and used, ensuring users are informed and protected. The system also respects intellectual property rights, using third-party APIs in compliance with their terms of service. The focus is on creating an ethical, secure, and user-friendly platform that operates with integrity, aligning with core ethical principles at every stage of development.

3. Type

The Smart Room Accommodation and Tracking System is a web-based application designed to serve as a comprehensive solution for managing hostel accommodations at Geethanjali College. This application provides users with functionalities such as searching for nearby hostels, checking room availability, making reservations, and tracking occupancy status in real time. The website features a user-friendly interface built with HTML, CSS, and JavaScript, ensuring an intuitive and responsive experience for all users, including students, faculty, and hostel administrators.

Additionally, the backend is powered by PHP and MySQL, which handle the server-side logic and database management, respectively. The application supports secure user logins, preventing unauthorized access and protecting sensitive information. By offering a centralized platform for booking and managing hostel accommodations, the Smart Room Accommodation and Tracking System enhances operational efficiency, reduces manual effort, and promotes a better experience for all users.

The project belongs to the category of software products aimed at improving accommodation management processes within educational institutions, thus facilitating smoother transitions for students and more efficient operations for administrators.

4. Standards

· **Data Privacy and Security**: Ensure compliance with data protection laws and implement strong encryption and access controls.

· **User-Friendly Interface**: Design a responsive and accessible interface for ease of use across devices.

· **Core Functionalities**: Include modules for student management, room allocation, fee management, and reporting.

· **Integration and Scalability**: Support integration with other systems and ensure scalability to handle growth.

· **Compliance and Support**: Follow legal standards, maintain regular updates, and provide robust technical support and disaster recovery plans.

2.4 SCOPE OF THE PROJECT

This project aims to enhance the efficiency and effectiveness of managing hostel accommodations in educational institutions by automating key processes such as room allocation, occupancy tracking, and resource planning. The system maintains a comprehensive database of the hostel's infrastructure, detailing the number of floors, rooms, and their dimensions, which serves as a basis for intelligent room allocation algorithms. By employing advanced optimization techniques, the system assigns rooms based on predefined criteria, such as occupant preferences, room size, and occupancy limits, ensuring optimal use of available space.

A dynamic color-coded visual system allows for real-time updates on room occupancy status: green for vacant rooms, ash for fully occupied rooms, and additional colors for partially occupied rooms. The HMS also supports advance bookings, effectively managing reservations to prevent double bookings and improve room utilization. Its robust reporting and analytics capabilities provide hostel administrators with detailed insights into occupancy rates, room usage, and resource allocation, which are crucial for data-driven decision-making and planning maintenance or renovations.

The web-based interface of the HMS offers a user-friendly experience for residents, allowing them to view room details, occupancy status, make reservations, and report maintenance issues. Administrators benefit from an intuitive dashboard that facilitates the management of room assignments, occupancy records, and overall hostel operations, ensuring a streamlined and efficient management process.

2.5 SYSTEM CONFIGURATION

1. SOFTWARE REQUIREMENTS

- **Web Browser**: Google Chrome, Mozilla Firefox, or any other modern web browser
- **Code Editor**: Visual Studio Code
- **Web Technologies**: HTML, CSS, JavaScript
- **Server-Side Scripting**: PHP
- **Database Management System**: MySQL
- **Active Internet Connection**: Required for accessing the web application and server

2. HARDWARE REQUIREMENTS

- **Operating System:** Windows
- **Processor:** Intel Core i5
- **Memory:** 8 GB RAM or more
- **Data Storage**: Hostels' data, including room details, occupancy status, user credentials, and booking information, stored in the MySQL database

3. LITERATURE SURVEY

A literature survey is a comprehensive review of existing research, studies, and methodologies related to a particular topic or field of study. It helps in identifying gaps in current research, understanding various approaches to solving similar problems, and determining how to refine or improve upon existing systems. In the context of technological projects, a literature survey is essential for ensuring that the proposed system is grounded in well-established knowledge while addressing gaps or limitations identified in previous work. By reviewing relevant research, the survey also helps in avoiding redundancy and informs the development of a more innovative and efficient solution.

[1] Harina, P., Kavya, K. K., Sharmikha Sree, R., & Meera, S. (2023). *Hostel Management*. IEEE Document.

The overwhelming majority of recently founded educational institutions manage their assets, in particular housing facilities, utilizing outmoded conventional methods. The inherent limitations and restrictions of outdated approaches have significantly harmed the organizational performance of educational institutions in general. The development of an automated system for managing hostel accommodations is advised by this study. According to the web-based system for managing hostels, the university hostel can successfully house college students. The computerized upkeep of the hostel records is useful. The major goal is to assist in automating the crucial chores of hostel management. The created system is more graphical user interface (GUI) focused, dependable, efficient, and secured with access control mechanisms, outperforming the shortcomings of conventional hostel administration techniques.

[2] Nandanwar, T., Bahutule, P., & Buddala, R. (2020). ***A Study on Shift towards Digitization of Hostel Room Allotment for a University*****. IEEE Document.**

The room allotment process in large universities and colleges can be tedious and inefficient, often resulting in delays and dissatisfaction among students. The conventional method, typically conducted manually, is prone to errors, prolonged waiting times, and difficulties in accommodating students' preferences for specific rooms. These drawbacks are especially evident in institutions with large student populations, where coordinating merit-based room assignments and preferences is a time-consuming task. This paper addresses these challenges by developing a digital platform designed to streamline and modernize the hostel room allocation process.

The proposed solution is an online room allotment portal that replaces the manual, conventional approach. By utilizing digital technologies, the portal allows students to select rooms based on their merit and preferences, significantly reducing the administrative burden and eliminating the inefficiencies of manual room selection. Through this platform, students can view available rooms in real-time and make informed decisions, ensuring that they get a room that best fits their needs. This not only enhances the overall experience for students but also minimizes the potential for human errors and delays in the room allotment process.

The project concludes with the successful development of a fully functional website that can be integrated into a university’s existing infrastructure. This digital solution demonstrates a practical application of web-based technologies in solving everyday problems faced by students. The paper also discusses potential areas for improvement, such as adding features for personalized room recommendations or optimizing room allocation algorithms based on various factors. The platform’s scalability and adaptability make it a valuable tool for universities looking to modernize their administrative processes and improve student satisfaction in the room allocation system.

[3] *Hostelworld: Budget Accommodation, Hostels, and Cheap Hotels*.

Hostelworld has established itself as a specialized platform in the online travel industry, focusing primarily on the needs of budget-conscious and solo travellers, particularly those drawn to hostel accommodations. Unlike larger and more generalized OTAs like Booking.com and Airbnb, which cater to a wide array of accommodation types and traveller preferences, Hostelworld hones in on the hostel experience, offering a unique value proposition cantered around affordability and social engagement. The platform capitalizes on a growing trend among younger, experience-driven travellers who prioritize meaningful interactions and community-oriented stays over luxury. This distinct focus allows Hostelworld to cater to a specific traveller demographic that seeks not just a place to stay, but a space that encourages social connections and cultural exchange, distinguishing it from its broader competitors.

A key component of Hostelworld's success lies in its ability to build and maintain trust with its users. The platform has integrated essential trust-building features such as peer reviews, secure payment systems, and an intuitive user interface, all of which are critical in fostering a sense of safety and reliability. Peer reviews, in particular, are a major driver of decision-making for travelers seeking budget accommodations, as they rely heavily on the experiences of other like-minded individuals to inform their choices. Furthermore, Hostelworld's secure payment gateways ensure a hassle-free booking experience, while its user-friendly design makes navigation simple, even for those unfamiliar with online travel platforms. These features not only enhance user confidence but also contribute significantly to retaining loyal customers in an industry where competition is fierce and user expectations are high. Beyond user experience, Hostelworld’s global impact on the hospitality industry is noteworthy, especially in promoting visibility for smaller, independent hostels that might otherwise struggle to compete against larger hotel chains. By providing a platform that highlights budget-friendly and often independently-owned accommodations, Hostelworld democratizes travel by making it accessible to a wider audience.

4. SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

The system architecture for the Smart Room Accommodation and Tracking **System** is designed to provide a seamless and efficient experience for managing hostel accommodations. The architecture comprises a robust frontend for user interaction and a powerful backend for handling server-side logic and database management. Below is a breakdown of the architecture:

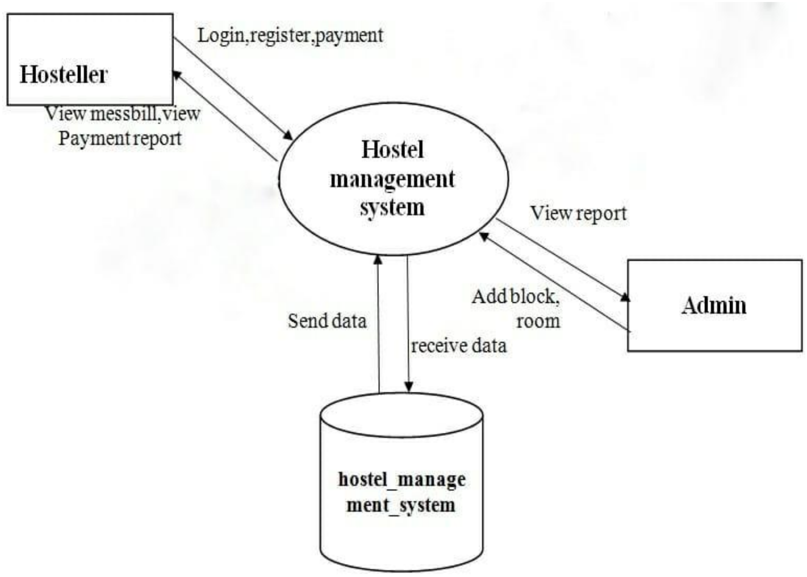


Fig 4.1 System Architecture

1. Frontend

- **Technologies**: HTML, CSS, JavaScript
- **Responsibilities**:

  ○ **User Interface (UI)**: The frontend is built using HTML, CSS, and JavaScript to create a dynamic and responsive user interface. This ensures that users, including students, faculty, and administrators, have an intuitive experience when navigating through the application.

  - **Real-Time Updates**: JavaScript enables real-time updates for room availability and occupancy status, enhancing user experience by providing immediate feedback.
  - **Interactive Elements**: The application uses JavaScript to manage interactive elements, such as booking forms and status updates, allowing users to easily make reservations and check the status of rooms.

2. Backend

- **Technologies**: PHP, MySQL
- **Responsibilities**:
  - **Server-Side Logic**: PHP handles server-side logic, including processing user requests, managing sessions, and executing business rules related to room allocation and booking management.
  - **Database Management**: MySQL is used to manage the system's relational database, which stores critical data such as user credentials, room details, booking information, and occupancy status.
  - **Security and Authentication**: The backend implements secure user authentication and authorization mechanisms to ensure that only authorized users can access specific features and data.

3.Database:

- **Technology**: MySQL
- **Responsibilities**:

- **Data Storage**: MySQL serves as the central repository for storing all data related to hostels, including details about rooms, bookings, users, and occupancy status.
- **Data Integrity**: The database is designed to ensure data integrity and prevent issues like double booking using constraints and transaction management.
- **Scalability**: MySQL supports scalability, allowing the system to handle increasing amounts of data as the number of users and bookings grows.

1. **Modules Description**

This section outlines the various modules implemented in the Smart Room Accommodation and Tracking System:

- **Booking and Reservation Module**:
  - Allows users to make reservations and manage bookings through an interactive interface.
  - Prevents double booking by dynamically updating room availability in real time.
- **Occupancy Tracking Module**:
  - Tracks the occupancy status of each room using a color-coded system (e.g., green for vacant, ash for fully occupied).
  - Provides a visual representation of room status.
- **User Management Module:**
  - User Profiles: Allows users to create and manage their profiles, including personal information, booking history, and preferences.

  - Role-Based Access Control: Implements different access levels for students, hostel managers, and administrators, ensuring that each user has access to relevant functionalities.

- **Admin Dashboard Module:**
  - Offers a centralized dashboard for administrators to manage all aspects of the hostel operations, including bookings, user management, and maintenance requests.

- **Search and Filter Module:**
  - Provides users with advanced search and filtering options to quickly find available rooms based on preferences such as price, amenities, and location.
  - Enhances user experience by streamlining the process of finding suitable accommodations.

4.2 UML DIAGRAMS

Unified Modeling Language (UML) is a standardized modeling language used to visualize the design and architecture of software systems. UML diagrams are crucial in software engineering as they help developers and stakeholders understand the structure, behavior, and interactions within a system. Different types of UML diagrams serve various purposes, allowing for a comprehensive representation of a system's functionalities, components, and workflows.

1. Use case Diagram

The **use case diagram** illustrates the interactions between users (actors) and the system, showcasing the various functionalities the system offers. Each use case represents a specific function or process that the system can perform, such as booking rooms, checking occupancy status, managing user accounts, and generating reports. This diagram helps in identifying user requirements and clarifying the scope of the system, making it easier for developers and stakeholders to understand how users will interact with the application.

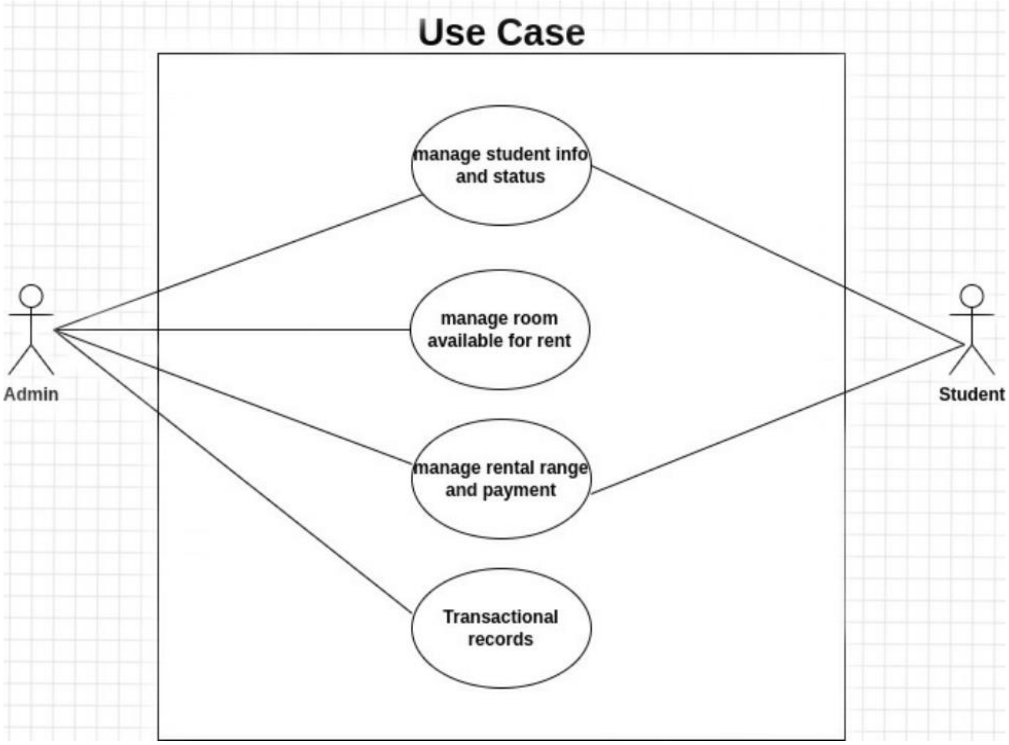

Use Case
manage student info and status
manage room available for rent
manage rental range and payment
Transactional records
Admin
Student

Fig 4.2.1 Use case diagram

2. Class Diagram

The **class diagram** represents the structure of the system by detailing its main classes, such as User, Room, Booking, and Admin, along with their attributes and methods. It illustrates the relationships between these classes, including inheritance, associations, and dependencies. Class diagrams provide a clear overview of how data is organized and how different components interact with each other, serving as a blueprint for the implementation of the system's functionalities.

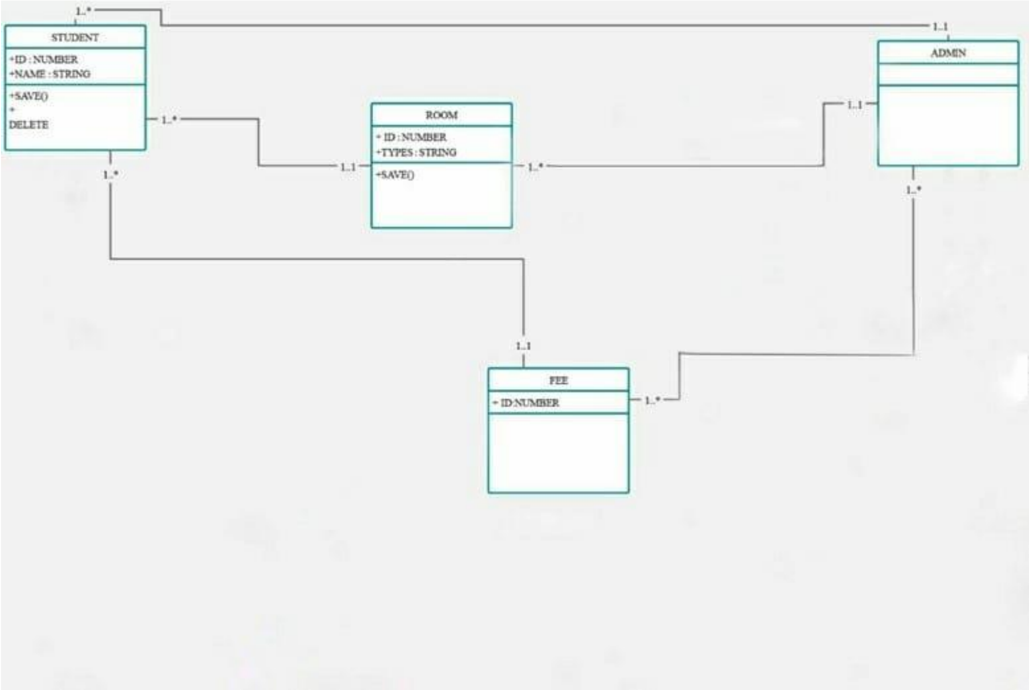


Fig 4.2.3 Class Diagram

3. Sequential Diagram

The **sequence diagram** demonstrates the flow of interactions between different system components during key operations. It provides a detailed view of the sequence of messages exchanged among the frontend, backend, and database layers during processes like making a booking or updating room status. Each component is represented as a lifeline, and arrows indicate the messages or calls being made. This diagram is essential for understanding the timing and order of operations, helping developers ensure that interactions are efficient and correctly implemented.

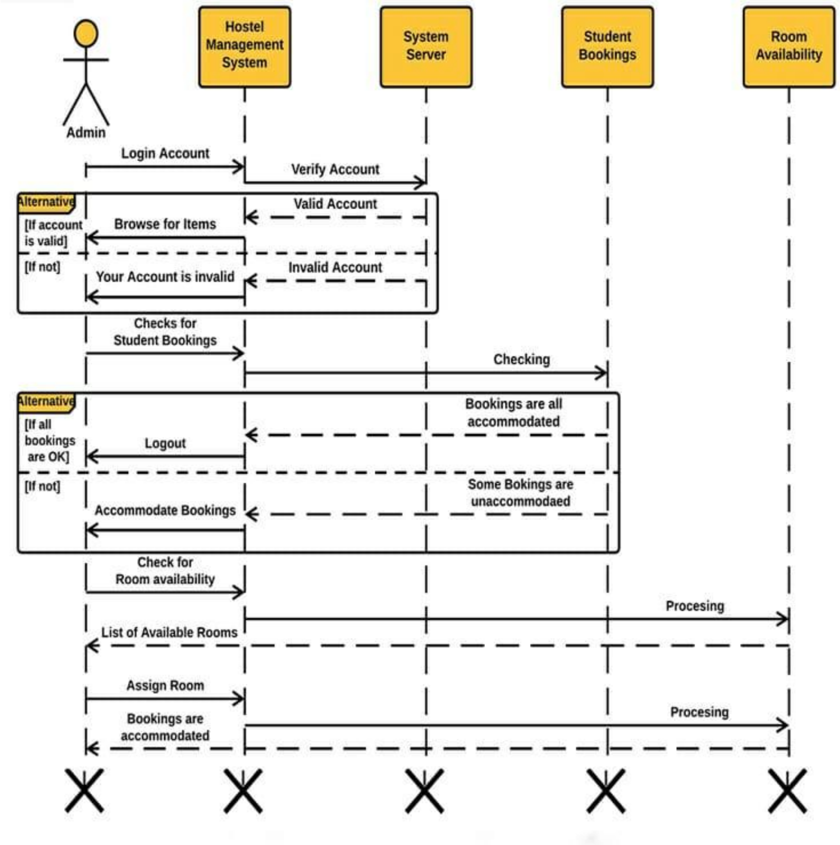


Fig 4.2.2 Sequential Diagram

4. Collaborative Diagram

The collaborative diagram (or communication diagram) focuses on the relationships and interactions between various objects in the system. Unlike sequence diagrams, which emphasize the timing of messages, collaborative diagrams illustrate how different objects collaborate to perform a specific function. They display the objects involved and the messages exchanged among them, often numbered to indicate the sequence. This diagram helps in understanding the overall structure of interactions and the roles of different components in achieving a task.

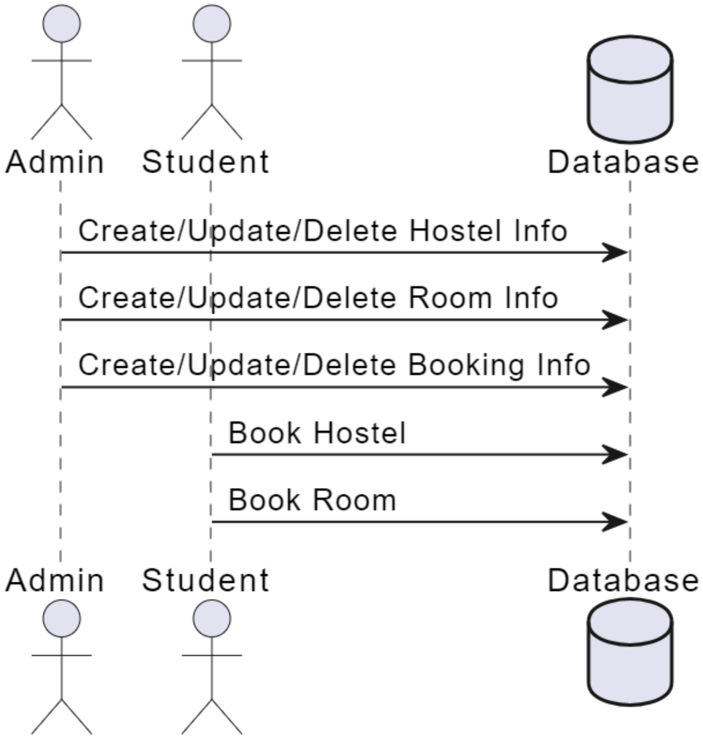

Admin
Student
Database
Create/Update/Delete Hostel Info
Create/Update/Delete Room Info
Create/Update/Delete Booking Info
Book Hostel
Book Room
Admin
Student
Database

Fig: 4.2.4 Collaborative Diagram

5. Activity Diagram

The activity diagram depicts the workflow of a particular process or operation within the system.

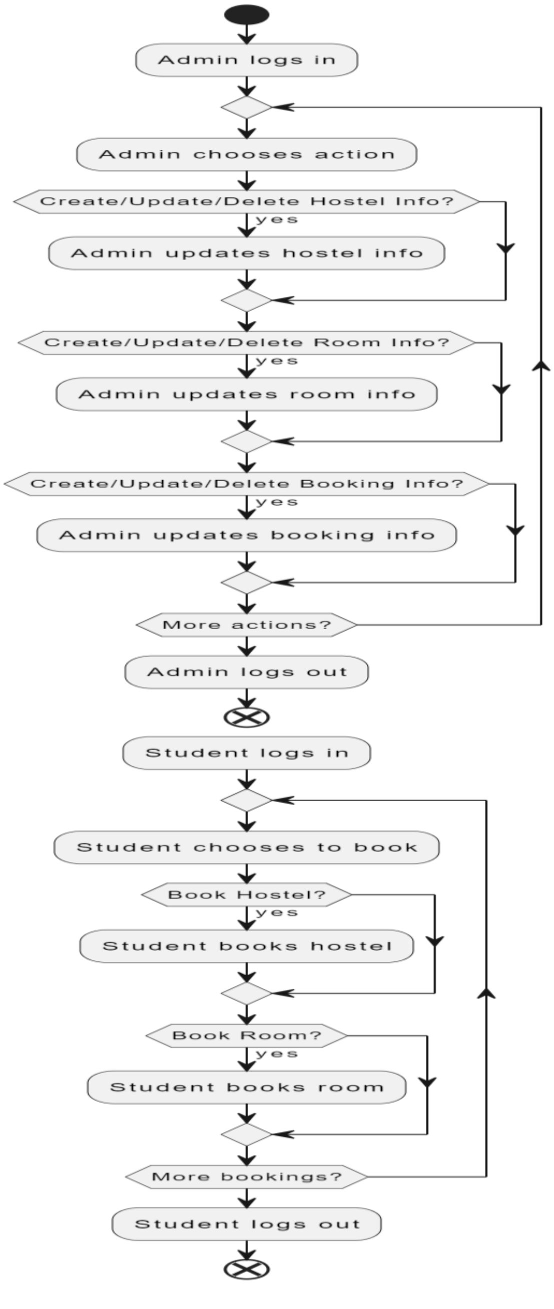


Fig: 4.2.5 Activity Diagram

5. IMPLEMENTATION

5.1 Implementation

The Smart Room Accommodation and Tracking System is a web-based application designed to streamline the management of hostel accommodations. It provides a user-friendly interface for students and administrators, allowing them to efficiently manage bookings, occupancy status, and other related activities. The implementation leverages a combination of frontend and backend technologies, specifically HTML, CSS, JavaScript, PHP, and MySQL.

Frontend Setup:

1. **Initialization:**
   - The frontend was set up using standard web technologies: HTML, CSS, and JavaScript. A well-structured directory was created to organize files logically, ensuring maintainability and scalability of the application.
   - A local development server was initiated using tools like XAMPP or WAMP to enable live testing and facilitate rapid development.
2. **Component Development:**
   - Core components of the application, including the user interface for booking, room status checks, and user account management, were developed using HTML for structure and CSS for styling.
   - JavaScript was utilized to manage dynamic elements such as real-time updates for room availability and interactive booking forms, enhancing user engagement.
3. **Styling and Theming:**
   - CSS was employed to create a visually appealing interface, using responsive design principles to ensure compatibility across various devices.
   - Additional libraries such as Bootstrap or custom CSS frameworks were integrated to streamline the styling process and provide a consistent look and feel throughout the application.

Backend Setup:

1. **Server Configuration:**
   - The backend was initialized with PHP to handle server-side logic. An index.php file was created to set up the server and define routes for various functionalities.
   - MySQL was utilized as the database management system, allowing for efficient storage and retrieval of data related to users, bookings, and room details.
2. **Routing and Middleware:**
   - API routes were configured to handle requests for user authentication, room booking, and occupancy tracking. PHP scripts were created to manage these interactions with the database.
   - Basic security measures, such as input validation and prepared statements, were implemented to safeguard against common vulnerabilities like SQL injection.
3. **Database Management:**
   - A MySQL database was designed to store critical data, including user profiles, room information, booking statuses, and occupancy details.
   - Tables were created with appropriate relationships, ensuring data integrity and preventing issues like double bookings through the use of constraints.

Development Environment

- The development environment was supported by PHP and MySQL for backend operations, with tools like XAMPP providing an easy way to manage the local server. A code editor, such as Visual Studio Code or Sublime Text, was used for writing and organizing the code, benefiting from features like syntax highlighting and debugging capabilities.

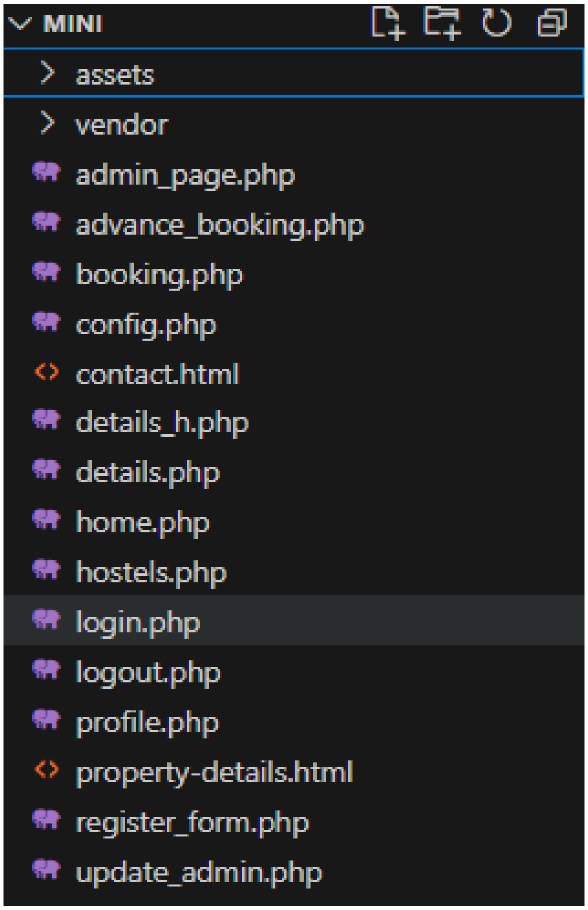
Integration and Testing

1. **Database Integration:**
   - The application was tested to ensure proper integration with the MySQL database, verifying that all data interactions—such as user registrations, bookings, and occupancy updates—functioned as expected.
   - SQL queries were optimized for performance, ensuring fast data retrieval and updates.
2. **Frontend-Backend Communication:**
   - AJAX calls were implemented using JavaScript to facilitate asynchronous communication between the frontend and backend. This allowed for seamless updates without requiring full page reloads.
   - Error handling mechanisms were established to manage responses effectively, providing users with feedback on successful operations or any encountered issues.

User Interaction and Features

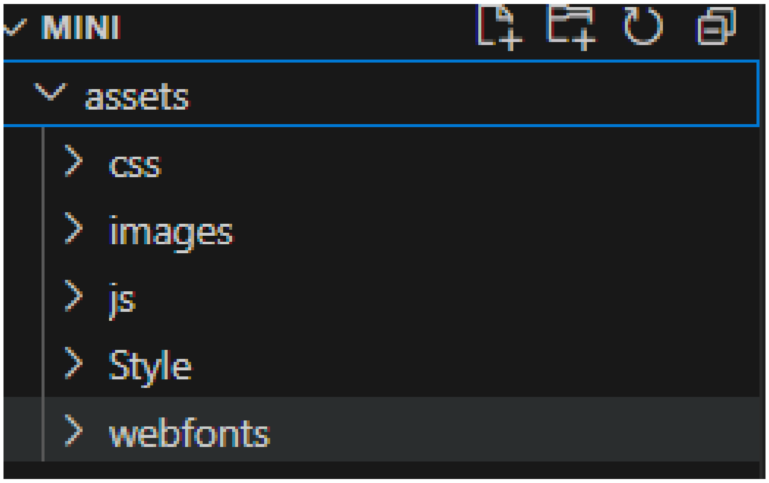
1. **Core Functionalities:**
   - Features such as booking and reservation management, occupancy tracking with a color-coded status system, and user profile management were implemented.
   - JavaScript was used to enhance interactivity, ensuring users could navigate through the application intuitively and efficiently.
2. **Testing and Feedback:**
   - The application underwent thorough testing to identify and resolve any bugs or usability issues, ensuring a smooth user experience.
   - Feedback from initial users was gathered to further refine functionalities and improve the overall interface.

5.2 Sample Code



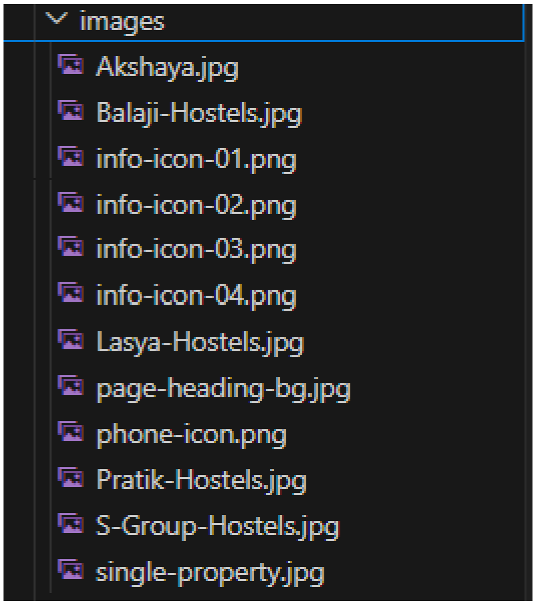
```
MINI
> assets
> vendor
admin_page.php
advance_booking.php
booking.php
config.php
contact.html
details_h.php
details.php
home.php
hostels.php
login.php
logout.php
profile.php
property-details.html
register_form.php
update_admin.php
```

Fig 5.2.1 Project Directory



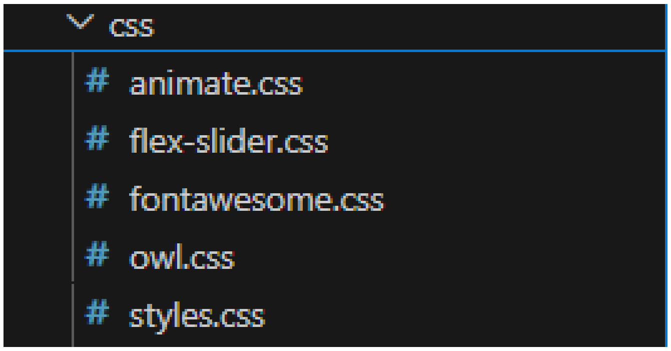
```
MINI
  assets
    css
    images
    js
    Style
    webfonts
```

Fig 5.2.2 assets Folder



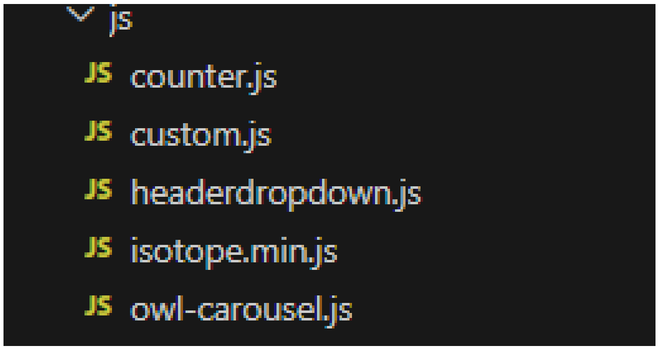
```
images
  Akshaya.jpg
  Balaji-Hostels.jpg
  info-icon-01.png
  info-icon-02.png
  info-icon-03.png
  info-icon-04.png
  Lasya-Hostels.jpg
  page-heading-bg.jpg
  phone-icon.png
  Pratik-Hostels.jpg
  S-Group-Hostels.jpg
  single-property.jpg
```

Fig 5.2.3 assets/image Folder



css
animate.css
flex-slider.css
fontawesome.css
owl.css
styles.css

Fig 5.2.4 asset/css Folder



js
counter.js
custom.js
headerdropdown.js
isotope.min.js
owl-carousel.js

Fig 5.2.5 asset/js Folder

1. **details_h.php**

```
<?php
    @include 'config.php';
?>
<?php
   session_start();
   $isLoggedIn = isset($_SESSION['admin_id']) || isset($_SESSION['user_id']);
   $isAdmin = isset($_SESSION['admin_id']);
?>

<!DOCTYPE html>
<html >
    <head>
        <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;400;500;600;700;800;900&display=swap" rel="stylesheet">
```

```
        <title>AccomRez</title>
        <link rel="stylesheet" href="assets/css/fontawesome.css">
        <link rel="stylesheet" href="assets/css/animate.css">
        <link rel="stylesheet"href="https://unpkg.com/swiper@7/swiper-bundle.min.css"/>
    </head>
    <style>
      .nav-link {
    display: block;
    padding: 10px;
    margin: 5px 0;
    background-color: #343a40;
    color: #fff;
    text-align: center;
    border-radius: 5px;
    cursor: pointer;
    transition: background-color 0.3s ease, transform 0.2s;
}
.nav-link:hover {
    background-color: #495057;
    transform: scale(1.05);
}
.floor-container {
    display: flex;
    flex-wrap: wrap;
    gap: 10px;
}
.room {
    padding: 15px;
    border-radius: 5px;
    text-align: center;
    cursor: pointer;
    flex: 1 0 30%;
    transition: transform 0.2s, box-shadow 0.2s;
}
.room:hover {
    transform: scale(1.05);
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
}
.vacant {
    background-color: #28a745; /* Green for vacant rooms */
    color: white;
}
```

```
.partially-occupied {
    background-color: #ffc107; /* Yellow for partially occupied */
    color: black;
}
.fully-occupied {
    background-color: #dc3545; /* Red for fully occupied */
    color: white;
}
.over-occupied {
    background-color: #6c757d; /* Gray for over-occupied */
    color: white;
}
    </style>
    <body>
  <header class="header-area">
    <div class="container">
        <div class="row">
            <div class="col-12">
                <nav class="main-nav">
                    <a href="home.html" class="logo">
                        <h1>AccomRez</h1>
                    </a>
                    <ul class="nav">
                      <li><a href="home.php">Home</a></li>
                      <li><a href="hostels.php">Hostels</a></li>
                      <?php if ($isLoggedIn): ?>
                            <li class="dropdown">
                                <a href="#" class="dropdown-toggle" data-toggle="dropdown">Profile</a>
                                <ul class="dropdown-menu">
                                    <?php if ($isAdmin): ?>
                                        <li><a href="admin_page.php">Admin Dashboard</a></li>
                                    <?php else: ?>
                                        <li><a href="user-profile.php">User Profile</a></li>
                                    <?php endif; ?>
                                    <li><a href="logout.php">Logout</a></li>
                                </ul>
                            </li>
                        <?php else: ?>
                            <!-- If not logged in, show Register/Login -->
                            <li><a href="register_form.php">Register/Login</a></li>
                        <?php endif; ?>
```

```
                  </ul>
                </nav>
            </div>
        </div>
    </div>
  </header>
  <div class="page-heading header-text">
    <div class="container">
      <div class="row">
        <div class="col-lg-12">
          <h3><?php echo htmlspecialchars($hostel['hostel_name']); ?> - Details</h3>
        </div>
      </div>
    </div>
  </div>
  <div class="single-property section">
    <div class="container">
      <div class="row">
        <div class="col-lg-6">
          <div class="main-image">
            <img src="<?php echo htmlspecialchars($hostel['hostel_path']); ?>" alt="Hostel
Image">
          </div>
          <div class="main-content">
            <h2><?php echo htmlspecialchars($hostel['hostel_name']); ?></h2>
            <h5>City: <?php echo htmlspecialchars($hostel['city']); ?></h5>
            <h5><?php echo htmlspecialchars($hostel['description']); ?></h5>
            <ul>
                <li>Floors: <span><?php echo htmlspecialchars($hostel['num_floors']);
?></span></li>
                <li>Rooms: <span><?php echo htmlspecialchars($hostel['num_rooms']);
?></span></li>
                <li>Vacancies: <span><?php echo htmlspecialchars($hostel['total_vacancies']);
?></span></li>
                <li>Contact: <span><?php echo htmlspecialchars($hostel['email']);
?></span></li>
                <li>Phone: <span><?php echo htmlspecialchars($hostel['phone']); ?></span></li>
            </ul>
          </div>
        </div>
        <div class="col-lg-4">
          <div class="info-table">
```

```
            <ul>
              <li>
                <img src="assets/images/info-icon-01.png" alt="" style="max-width: 52px;">
                <h4><?php echo $total_flat_space; ?><br><span>Max Room Space</span></h4>
              </li>
              <li>
                <img src="assets/images/info-icon-02.png" alt="" style="max-width: 52px;">
                <h4>Contract<br><span>Contract Ready</span></h4>
              </li>
              <li>
                <img src="assets/images/info-icon-03.png" alt="" style="max-width: 52px;">
                <h4>Payment<br><span>Payment Process</span></h4>
              </li>
              <li>
                <img src="assets/images/info-icon-04.png" alt="" style="max-width: 52px;">
                <h4>Safety<br><span>24/7 Under Control</span></h4>
              </li>
            </ul>
          </div>
        </div>
      </div>
    </div>
  </div>
  <div class="section best-deal">
    <div class="container">
      <div class="row">
        <div class="col-lg-12">
          <div class="tabs-content">
            <div class="row">
              <div class="tab-content" id="myTabContent">
                <div class="tab-pane fade show active" id="appartment" role="tabpanel" aria-
labelledby="appartment-tab">
                  <div class="row">
                    <div class="col-lg-3">
                      <div class="info-table">
                      <nav class="nav flex-column">
                        <?php
                          $floors = array_unique(array_column($rooms, 'floor'));
                          foreach ($floors as $floor) {
                            echo "<a class='nav-link' href='javascript:void(0);' data-
floor='$floor'>Floor $floor</a>";
                          }
            ?>
           </nav>
           </div>
          </div>
          <div class="col-lg-3">
           <div id="floorContainer" class="floor-container"></div>
          </div>
          <div class="col-lg-3">
          <div id="roomDetails" class="room-details">
           <div id="roomInfo"></div>
           <label for="bookingStart">Start Date:</label>
           <input type="date" id="bookingStart" required>
           <label for="bookingDuration">Duration (months):</label>
           <input type="number" id="bookingDuration" min="1" max="12" value="6"
required>
           <button id="bookNow">Book Now</button>
          </div>
          </div>
         </div>
        </div>
       </div>
      </div>
     </div>
    </div>
   </div>
  </div>
 </div>
 <script src="assets/js/headerdropdown.js"></script>
 <script>
  document.addEventListener("DOMContentLoaded", function() {
  const rooms = <?php echo json_encode($rooms); ?>;
  const floors = <?php echo json_encode($floors); ?>;
  const floorContainer = document.getElementById('floorContainer');
  const roomDetails = document.getElementById('roomDetails');
  const roomInfo = document.getElementById('roomInfo');
  const bookNowButton = document.getElementById('bookNow');
  const bookingStartInput = document.getElementById('bookingStart');
  const bookingDurationInput = document.getElementById('bookingDuration');
  function displayRooms(floor) {
    floorContainer.innerHTML = ''; // Clear previous content
    rooms.filter(room => room.floor == floor).forEach(room => {
      const roomDiv = document.createElement('div');
```

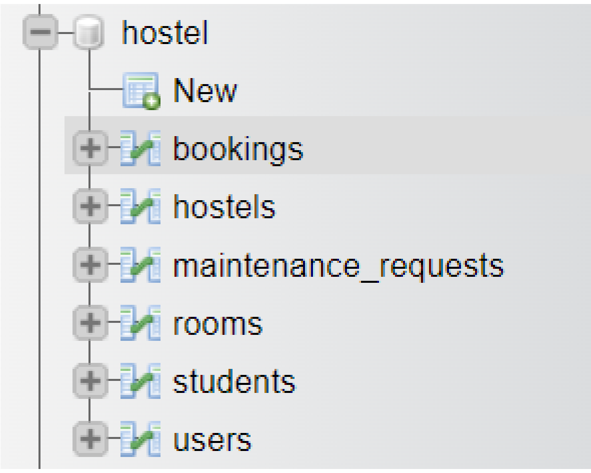
```
            roomDiv.className = `room ${getRoomStatusClass(room.people, room.capacity)}`;
            roomDiv.innerText = `Room ${room.room_number}`;
            roomDiv.onclick = () => showRoomDetails(room);
            floorContainer.appendChild(roomDiv);
        });
    }
    function getRoomStatusClass(people, capacity) {
        if (people == 0) return 'vacant';
        if (people < capacity - 1) return 'partially-occupied';
        if (people == capacity - 1) return 'fully-occupied';
        return 'over-occupied';
    }
    function showRoomDetails(room) {
        roomInfo.innerHTML = `
            <p>Room Number: ${room.room_number}</p>
            <p>Capacity: ${room.capacity}</p>
            <p>Occupied: ${room.people}</p>
            <p>Furniture: ${room.furniture_type}</p>
            <p>Accommodation: ${room.accommodation}</p>
        bookingStartInput.value = ''; // Reset date input
        bookingDurationInput.value = '6'; // Default to 6 months
        bookNowButton.onclick = () => bookRoom(room.id, room.room_number);
        roomDetails.style.display = 'block'; // Show details
    }
    function bookRoom(roomId, roomNumber) {
        const bookingStart = bookingStartInput.value;
        const bookingDuration = bookingDurationInput.value;
        if (!bookingStart) {
            alert("Please select a start date for the booking.");
            return;
        }

        if (bookingDuration <= 0) {
            alert("Please enter a valid duration for the booking.");
            return;
        }
        window.location.href = 
`advance_booking.php?id=${roomId}&start=${bookingStart}&duration=${bookingDuratio
n}`;
        alert(`Booking room ${roomNumber} from ${bookingStart} for ${bookingDuration} 
months.`);
    }
    document.querySelectorAll('.nav-link').forEach(link => {
        link.addEventListener('click', function(event) {
            event.preventDefault(); // Prevent default behavior
            displayRooms(this.getAttribute('data-floor'));
        });
    });
    if (floors.length > 0) {
        displayRooms(floors[0]);
    }
});
  </script>
  </body>
</html>
```

2. **booking.php**

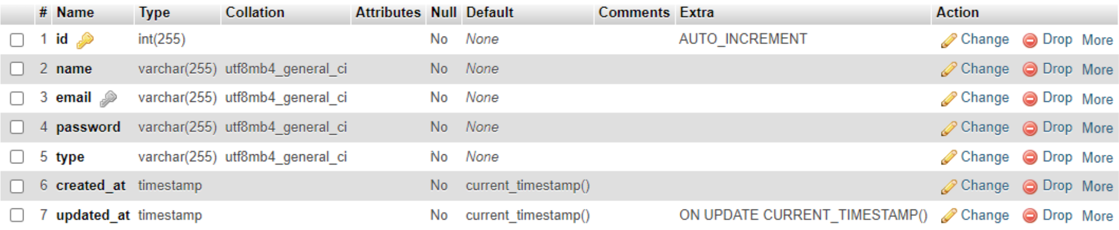
```
<?php
include('config.php');
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $room_id = $_POST['room_id'];
    $user_email = $_POST['user_email'];
    $sql = "SELECT * FROM rooms WHERE id='$room_id' AND people < capacity";
    $result = $conn->query($sql);
    if ($result->num_rows > 0) {
        $sql = "UPDATE rooms SET people = people + 1, occupancy_status = CASE
                WHEN people + 1 = capacity THEN 'over occupied'
                WHEN people + 1 < capacity THEN 'partially occupied'
                END
                WHERE id='$room_id'";
        if ($conn->query($sql) === TRUE) {
            echo "Booking successful;
        } else {
            echo "Error updating record: " . $conn->error;
        }
    } else {
        echo "Room is fully occupied or doesn't exist.";
    }
}
$sql = "SELECT * FROM rooms WHERE people < capacity";
$result = $conn->query($sql);
?>
<!DOCTYPE html>
<html lang="en">
```

```
<head>
  <title>Room Booking</title>
</head>
<body>
  <header>
    <h1>Room Booking</h1>
  </header>
  <div class="container">
    <h2>Available Rooms</h2>
    <?php
    if ($result->num_rows > 0) {
      while($row = $result->fetch_assoc()) {
        echo "
        <div class='room'>
          <div class='room-info'>
            <h3>Room " . $row['room_number'] . "</h3>
            <p>Capacity: " . $row['capacity'] . "</p>
            <p>Occupied: " . $row['people'] . "</p>
            <p>Furniture: " . $row['furniture_type'] . "</p>
          </div>
          <form method='POST' action='booking.php'>
            <input type='hidden' name='room_id' value='" . $row['id'] . "'>
            <label for='user_email'>Email: </label>
            <input type='email' name='user_email' required>
            <button type='submit'>Book Now</button>
          </form>
        </div>
        ";
      }
    } else {
      echo "<p>No available rooms at the moment.</p>";
    }
    ?>
  </div>
</body>
</html>
<?php
$conn->close();
?>
```



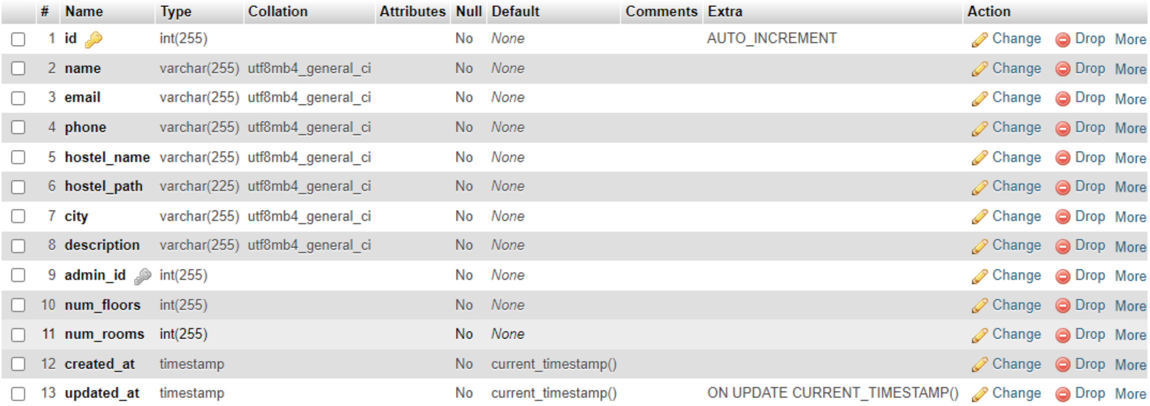
hostel
- New
- bookings
- hostels
- maintenance_requests
- rooms
- students
- users

Fig 5.2.6 Database



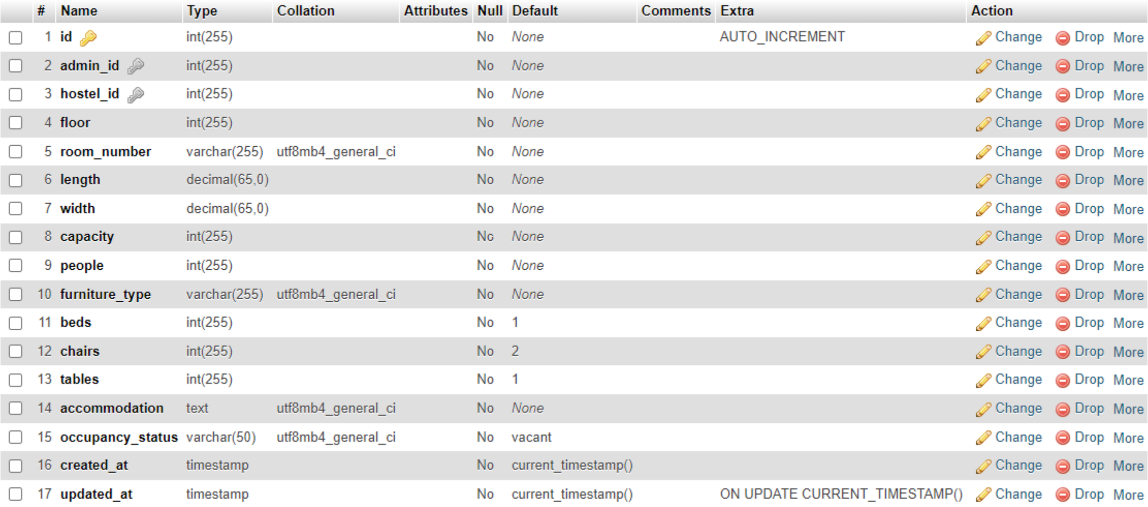
	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1	**id**	int(255)			No	*None*		AUTO_INCREMENT	Change Drop More
□	2	**name**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	3	**email**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	4	**password**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	5	**type**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	6	**created_at**	timestamp			No	current_timestamp()			Change Drop More
□	7	**updated_at**	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

Fig 5.2.7 User Table



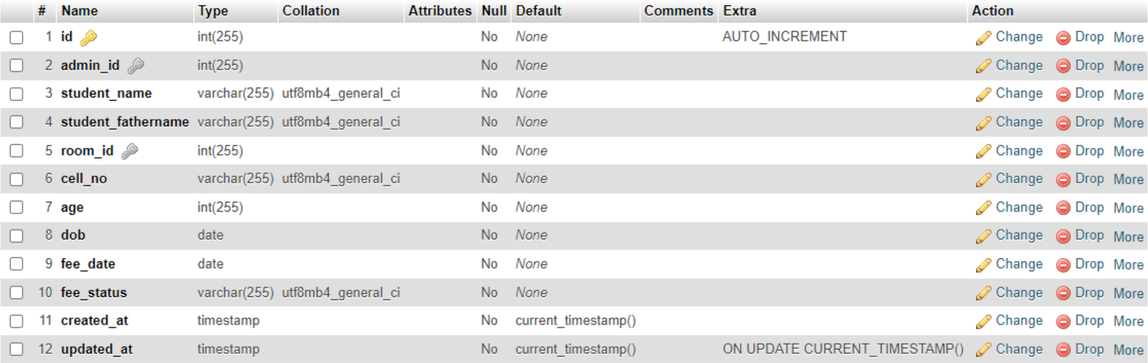
	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1	**id**	int(255)			No	*None*		AUTO_INCREMENT	Change Drop More
□	2	**name**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	3	**email**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	4	**phone**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	5	**hostel_name**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	6	**hostel_path**	varchar(225)	utf8mb4_general_ci		No	*None*			Change Drop More
□	7	**city**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	8	**description**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	9	**admin_id**	int(255)			No	*None*			Change Drop More
□	10	**num_floors**	int(255)			No	*None*			Change Drop More
□	11	**num_rooms**	int(255)			No	*None*			Change Drop More
□	12	**created_at**	timestamp			No	current_timestamp()			Change Drop More
□	13	**updated_at**	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

Fig 5.2.8 Hostel Table



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1	**id** 🔑	int(255)			No	*None*		AUTO_INCREMENT	Change Drop More
□	2	**admin_id** 🔑	int(255)			No	*None*			Change Drop More
□	3	**hostel_id** 🔑	int(255)			No	*None*			Change Drop More
□	4	**floor**	int(255)			No	*None*			Change Drop More
□	5	**room_number**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	6	**length**	decimal(65,0)			No	*None*			Change Drop More
□	7	**width**	decimal(65,0)			No	*None*			Change Drop More
□	8	**capacity**	int(255)			No	*None*			Change Drop More
□	9	**people**	int(255)			No	*None*			Change Drop More
□	10	**furniture_type**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	11	**beds**	int(255)			No	1			Change Drop More
□	12	**chairs**	int(255)			No	2			Change Drop More
□	13	**tables**	int(255)			No	1			Change Drop More
□	14	**accommodation**	text	utf8mb4_general_ci		No	*None*			Change Drop More
□	15	**occupancy_status**	varchar(50)	utf8mb4_general_ci		No	vacant			Change Drop More
□	16	**created_at**	timestamp			No	current_timestamp()			Change Drop More
□	17	**updated_at**	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

Fig 5.2.9 Room Table



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
□	1	**id** 🔑	int(255)			No	*None*		AUTO_INCREMENT	Change Drop More
□	2	**admin_id** 🔑	int(255)			No	*None*			Change Drop More
□	3	**student_name**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	4	**student_fathername**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	5	**room_id** 🔑	int(255)			No	*None*			Change Drop More
□	6	**cell_no**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	7	**age**	int(255)			No	*None*			Change Drop More
□	8	**dob**	date			No	*None*			Change Drop More
□	9	**fee_date**	date			No	*None*			Change Drop More
□	10	**fee_status**	varchar(255)	utf8mb4_general_ci		No	*None*			Change Drop More
□	11	**created_at**	timestamp			No	current_timestamp()			Change Drop More
□	12	**updated_at**	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()	Change Drop More

Fig 5.2.10 Booking Table

6. TESTING

6.1 Purpose of Testing

Purpose of Testing: The purpose of testing the Smart Room Accommodation and Tracking System is to ensure that the web platform operates as intended, offering users a seamless experience in managing hostel accommodations. Testing is aimed at verifying that the system is user-friendly, reliable, and capable of handling all necessary functionalities like room booking, occupancy tracking, and user management. It also ensures that the system integrates well with its backend, providing accurate real-time data and safeguarding against errors such as double bookings.

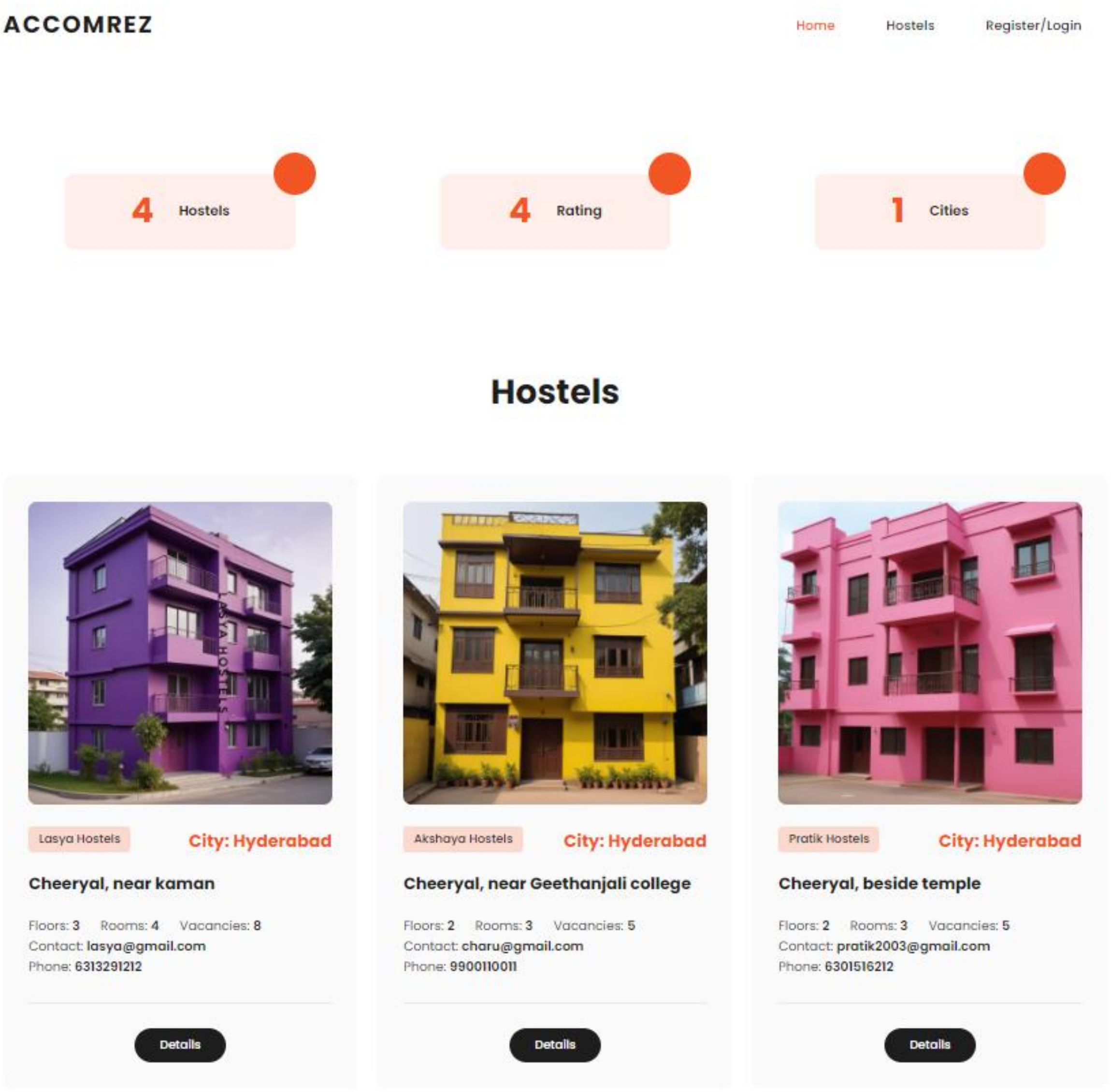
Types of Testing Performed

- **Unit Testing:** The goal of unit testing is to test individual components of the system to ensure they function correctly in isolation. For instance, specific modules such as the booking form, occupancy tracking, and user authentication are tested independently.
- **Integration Testing:** Integration testing is conducted to verify that different components of the system, particularly the frontend and backend, work together harmoniously. This testing ensures that user actions (e.g., booking a room) interact correctly with the database and are reflected in real-time across the platform.
- **Functional Testing:** Functional testing ensures that all system features operate as per the requirements. It verifies that each function, such as room booking, user authentication, profile management, and real-time occupancy updates, behaves as expected.

6.2 Testcases

Test Case No	Test Objective	Steps	Expected Result	Result
1	Admin Login	Open the web page. Click on the login. Enter the correct admin username and password. Click on the "Login" button	After a successful login, the system should navigate to the admin dashboard.	Success
2	User Navigation	Open the web page. Click on each of the following buttons: "Home", "Hostels", "Register/Login".	Each button should navigate to the respective page without error.	Success
3	Room booking	Log in as a user. Navigate to the hostel. Select a room and click "Book."	The system should confirm the booking and update the room's occupancy status.	Success

7. OUTPUT SCREENS



4 Hostels

4 Rating

1 Cities

Hostels

Lasya Hostels City: Hyderabad

Cheeryal, near kaman

Floors: 3 Rooms: 4 Vacancies: 8
Contact: lasya@gmail.com
Phone: 6313291212

Akshaya Hostels City: Hyderabad

Cheeryal, near Geethanjali college

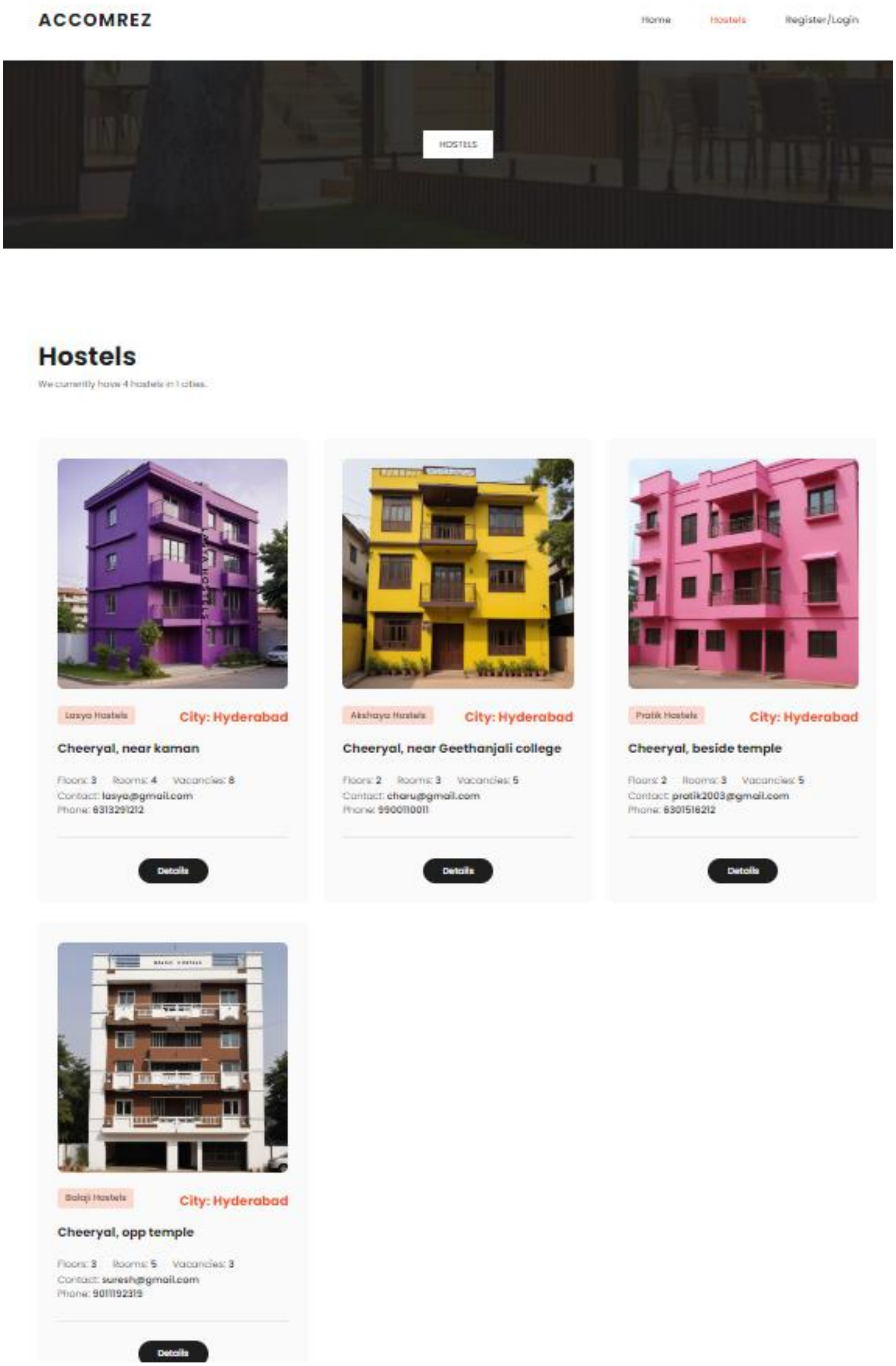
Floors: 2 Rooms: 3 Vacancies: 5
Contact: charu@gmail.com
Phone: 9900110011

Pratik Hostels City: Hyderabad

Cheeryal, beside temple

Floors: 2 Rooms: 3 Vacancies: 5
Contact: pratik2003@gmail.com
Phone: 6301516212

Fig 7.1 Home page



Hostels

We currently have 4 hostels in 1 cities.

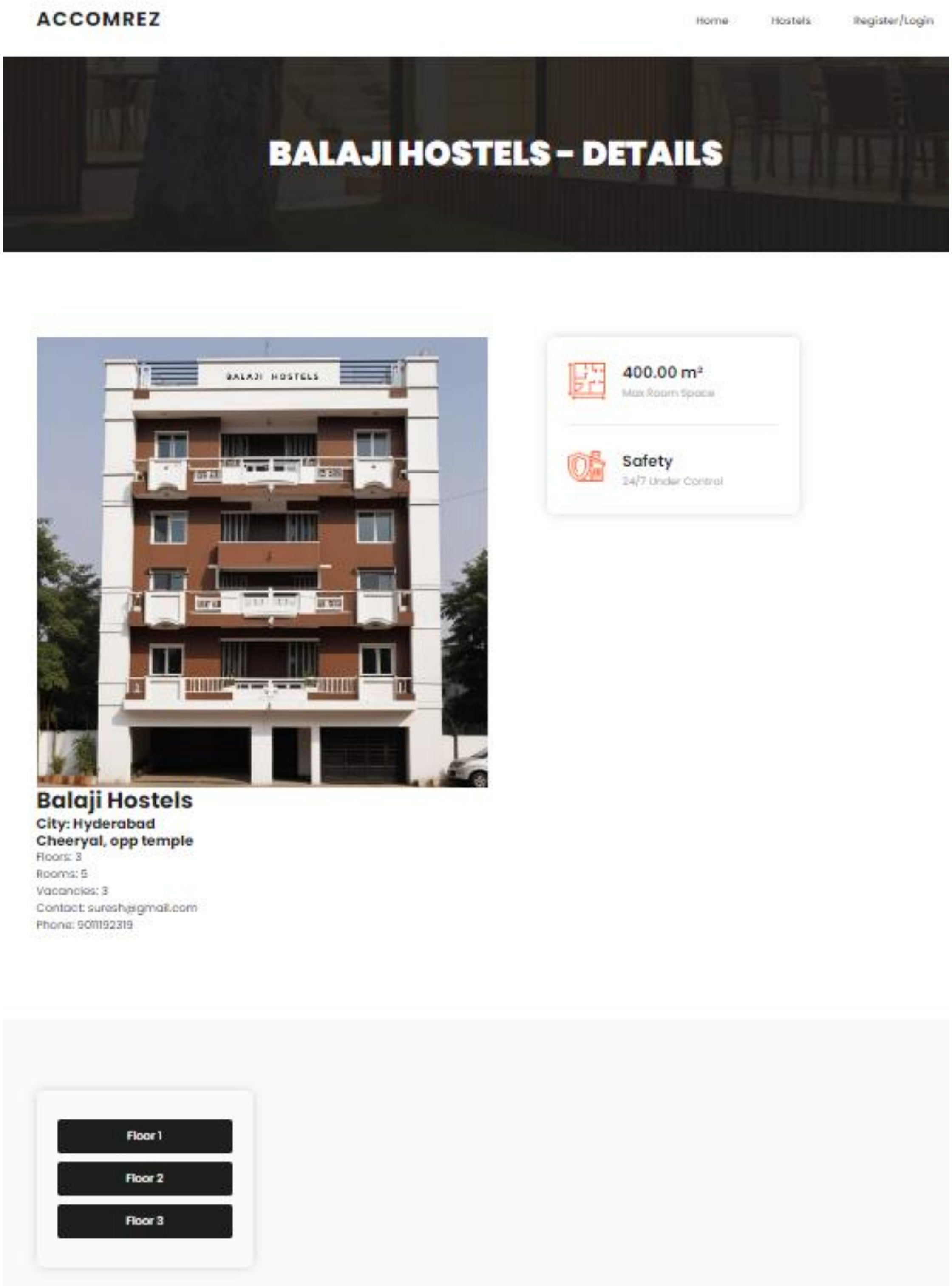
Lasya Hostels
City: Hyderabad
Cheeryal, near kaman
Floors: 3 Rooms: 4 Vacancies: 8
Contact: lasya@gmail.com
Phone: 8313291212

Akshaya Hostels
City: Hyderabad
Cheeryal, near Geethanjali college
Floors: 2 Rooms: 3 Vacancies: 5
Contact: charu@gmail.com
Phone: 9900110011

Pratik Hostels
City: Hyderabad
Cheeryal, beside temple
Floors: 2 Rooms: 3 Vacancies: 5
Contact: pratik2003@gmail.com
Phone: 6301516212

Balaji Hostels
City: Hyderabad
Cheeryal, opp temple
Floors: 3 Rooms: 5 Vacancies: 3
Contact: suresh@gmail.com
Phone: 9011192319

Fig 7.2 Hostels



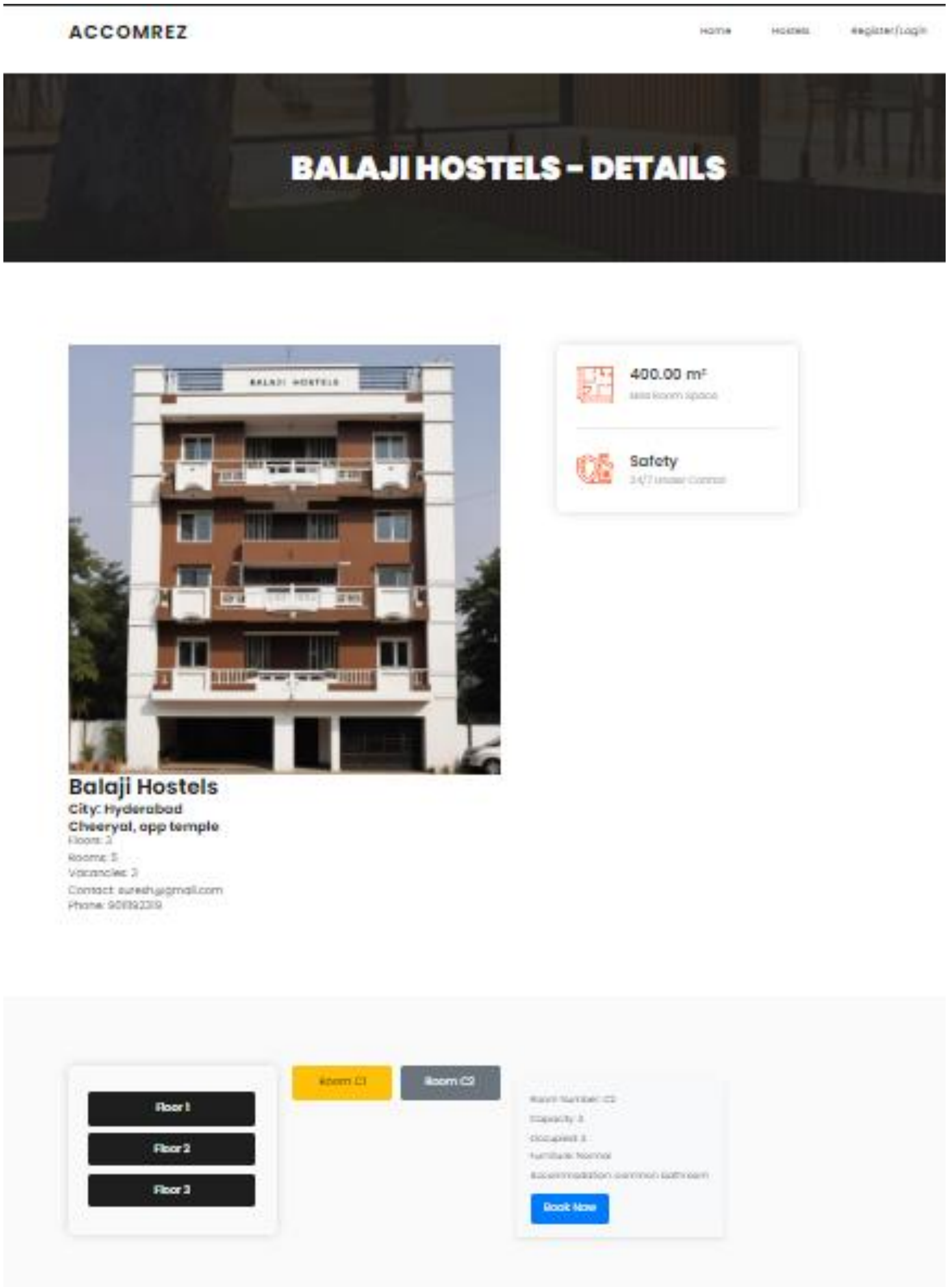
BALAJI HOSTELS - DETAILS

400.00 m²
Max Room Space

Safety
24/7 Under Control

Balaji Hostels
City: Hyderabad
Cheeryal, opp temple
Floors: 3
Rooms: 5
Vacancies: 3
Contact: suresh@gmail.com
Phone: 9011192319

Fig 7.3 Hostel details



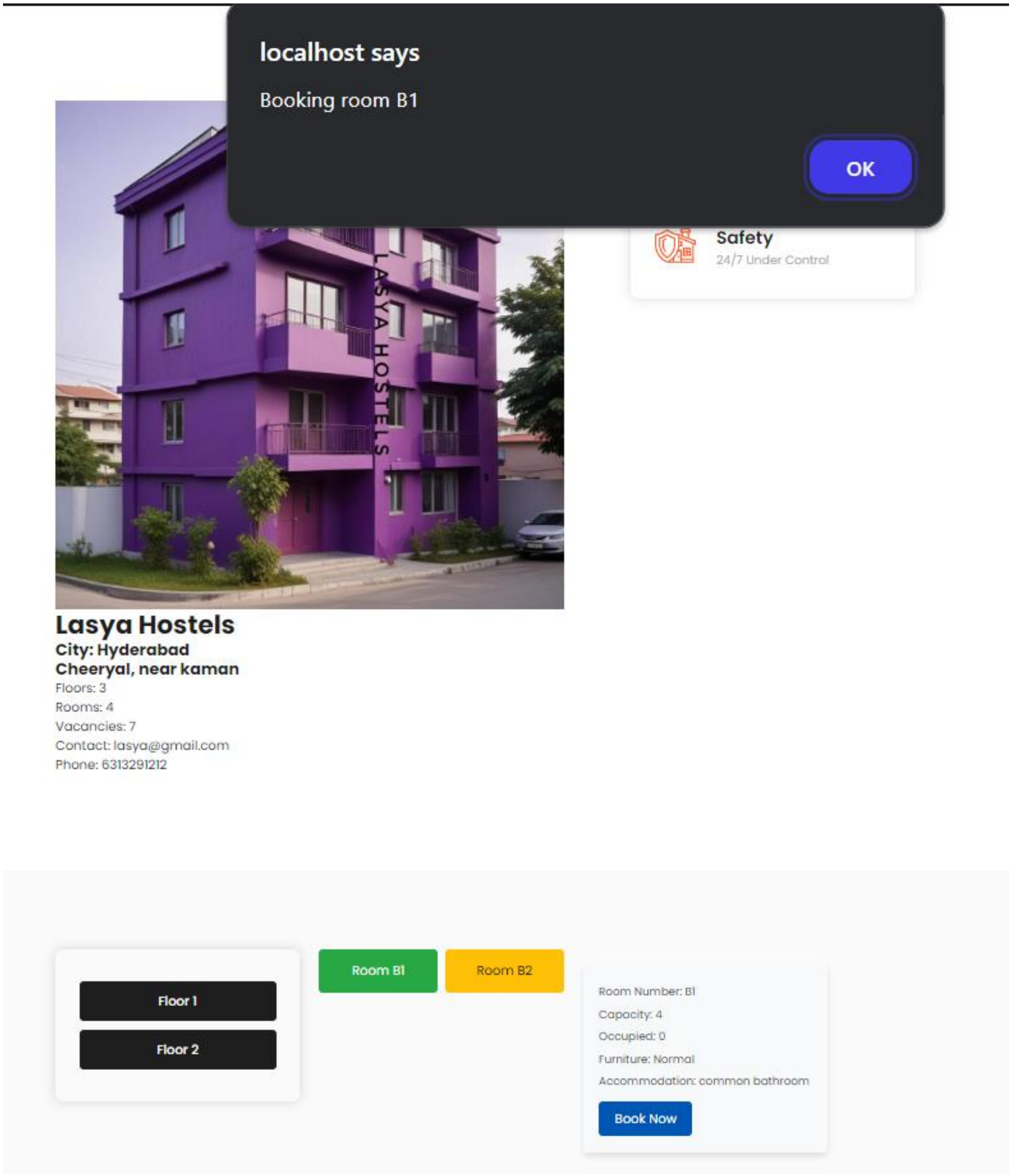
400.00 m²
Max Room Space

Safety
24/7 Under Control

Balaji Hostels
City: Hyderabad
Cheeryal, opp temple
Floors: 3
Rooms: 5
Vacancies: 3
Contact: suresh@gmail.com
Phone: 90392319

Room Number: C2
Capacity: 3
Occupied: 3
Furniture: Normal
Accommodation: common bathroom

Fig 7.4 Visualization of room occupancy



localhost says

Booking room B1

Lasya Hostels

City: Hyderabad

Cheeryal, near kaman

Floors: 3

Rooms: 4

Vacancies: 7

Contact: lasya@gmail.com

Phone: 6313291212

Room Number: B1

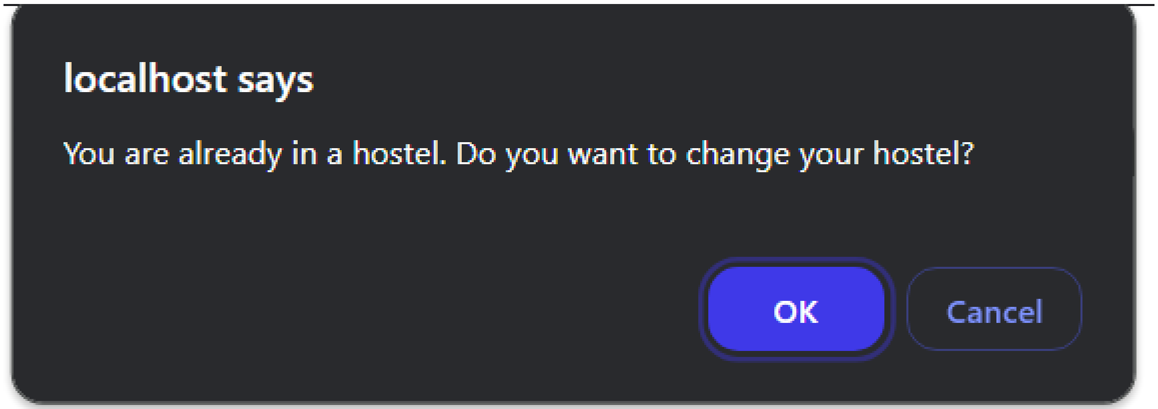
Capacity: 4

Occupied: 0

Furniture: Normal

Accommodation: common bathroom

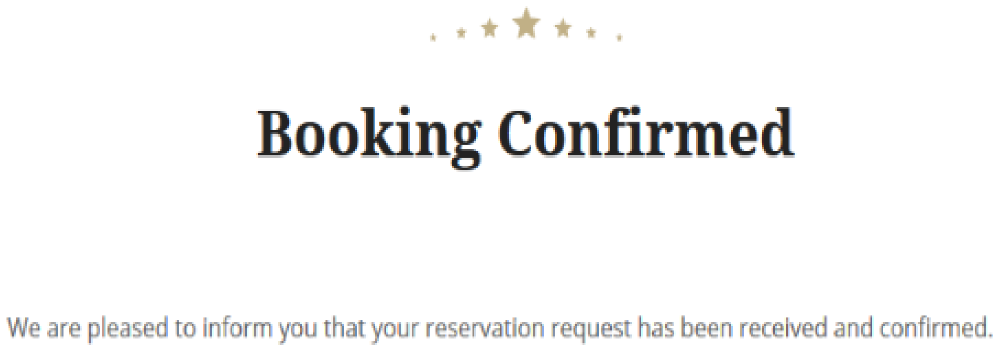
Fig 7.5 Conformation box



localhost says

You are already in a hostel. Do you want to change your hostel?

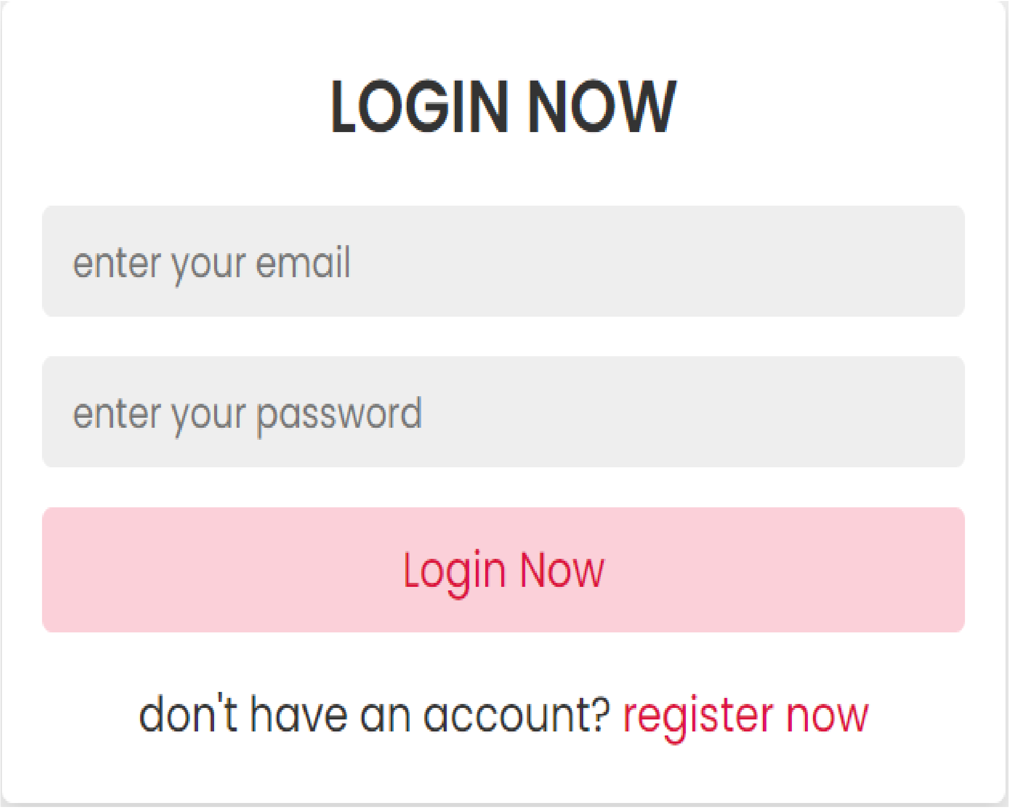
Fig 7.6 Hostel change conformation box



Booking Confirmed

We are pleased to inform you that your reservation request has been received and confirmed.

Fig 7.7 Confirmation Message



LOGIN NOW

enter your email

enter your password

Login Now

don't have an account? register now

Fig 7.8 Login Page

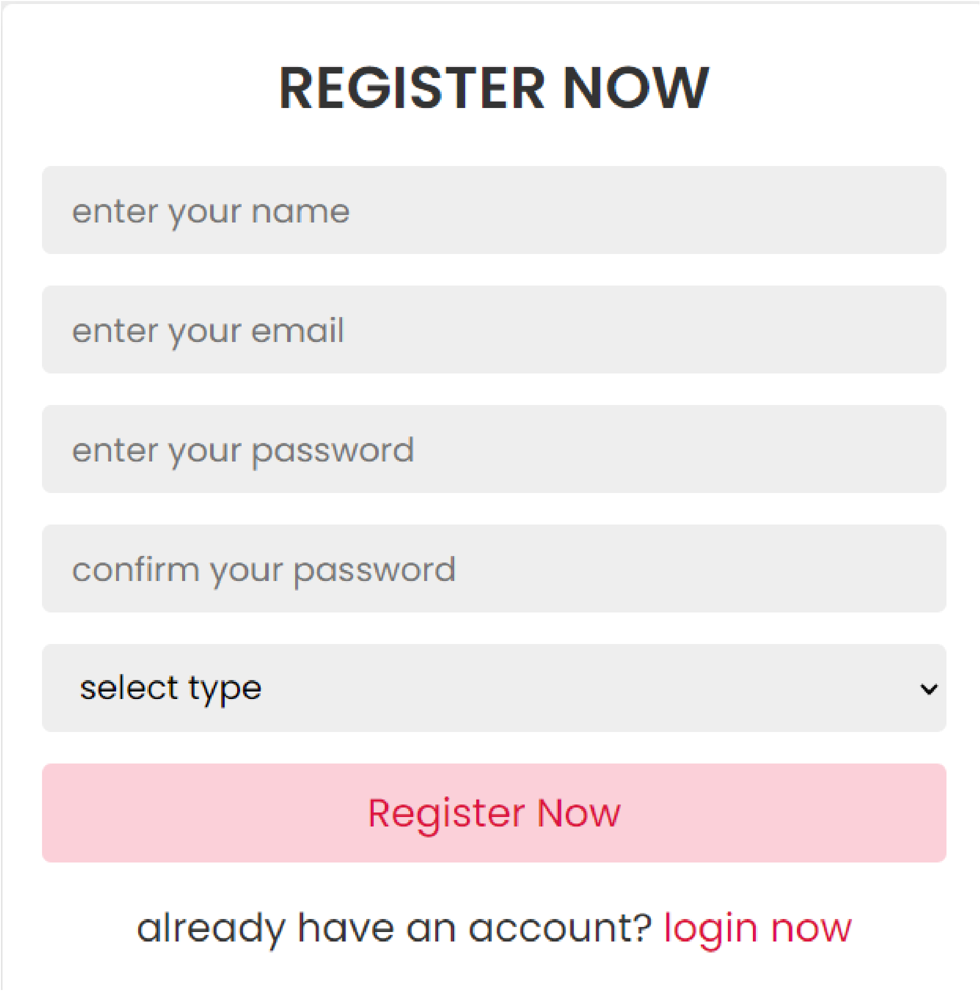


Fig 7.9 Register Page

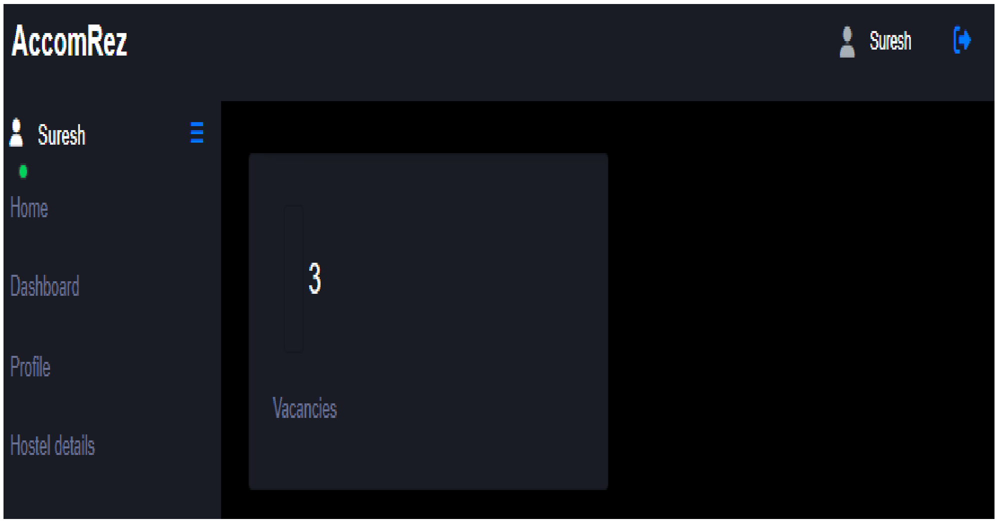


Fig 7.10 Admin Dashboard

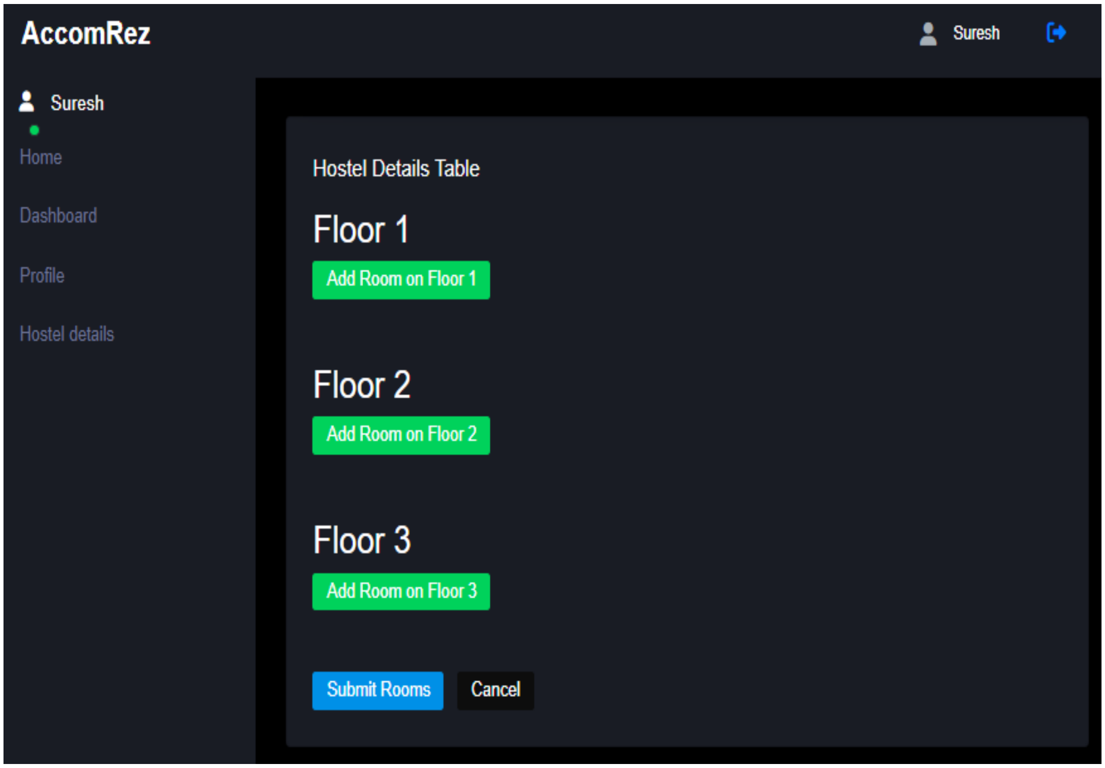
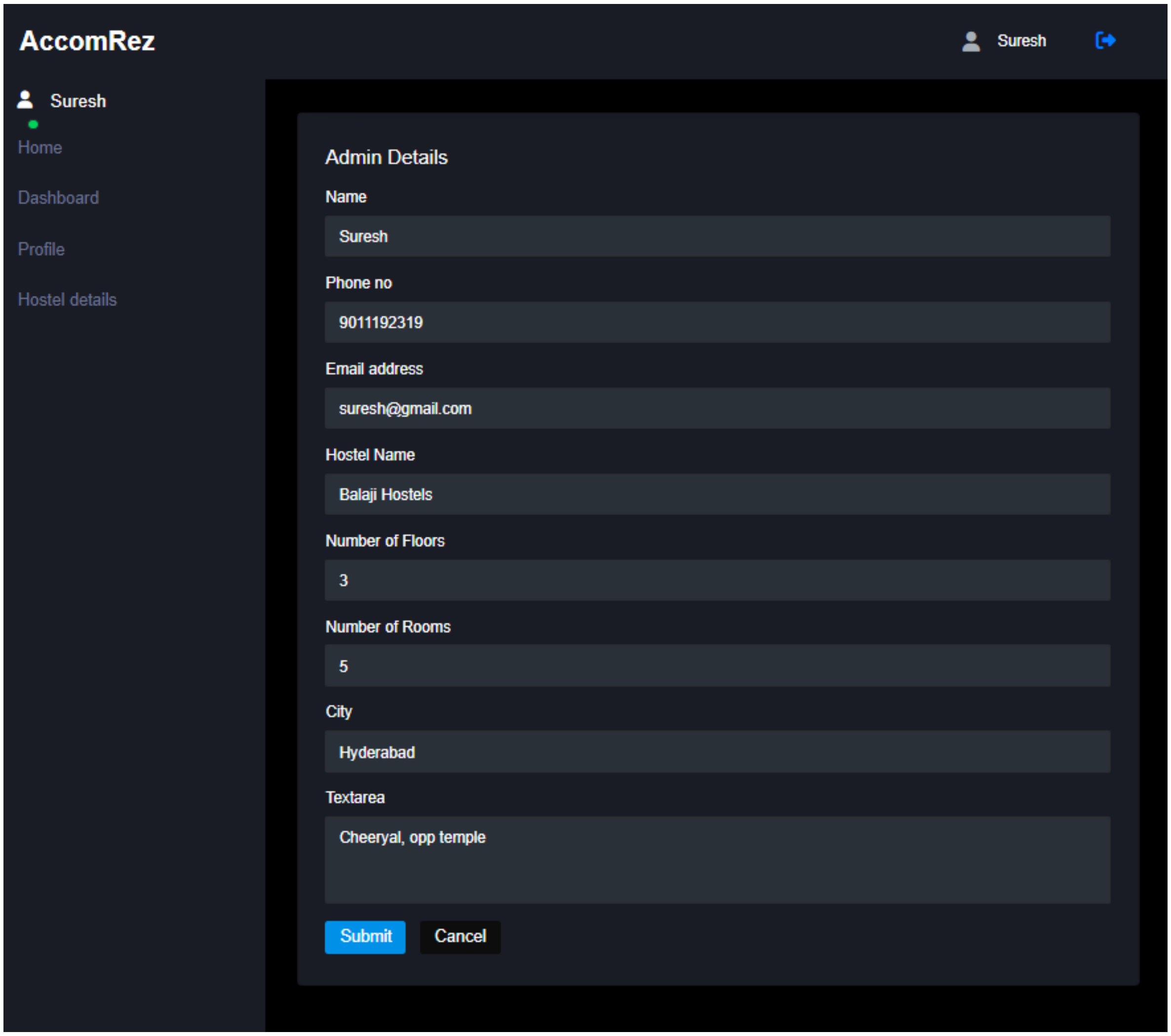


Fig 7.11 Add Hostel Details



Admin Details

Name

Suresh

Phone no

9011192319

Email address

suresh@gmail.com

Hostel Name

Balaji Hostels

Number of Floors

3

Number of Rooms

5

City

Hyderabad

Textarea

Cheeryal, opp temple

Fig 7.12 Admin Profile

8. CONCLUSION

8.1 Conclusion

The Smart Room Accommodation and Tracking System has proven to be a comprehensive and highly effective solution for addressing the common challenges faced by educational institutions in managing hostel accommodations for students, faculty, and staff. By utilizing modern web technologies such as HTML, CSS, JavaScript, PHP, and MySQL, the system offers a robust, user-friendly interface that significantly improves the efficiency of booking rooms, monitoring occupancy, and managing room availability in real time. Through automation, the system reduces the risk of manual errors, such as double bookings or mismanagement of room assignments, while simultaneously increasing transparency across the entire process. This ensures that both residents and administrators have easy access to accurate, up-to-date information regarding room availability and occupancy status, fostering smoother communication and decision-making within the institution.

Moreover, the system's design emphasizes ease of use for all stakeholders, from students looking to book rooms to administrators managing the overall hostel operations. With features like real-time updates, secure login mechanisms for different user roles, and intuitive navigation, the platform offers a streamlined and organized approach to handling hostel accommodations. By enhancing resource utilization and ensuring better management of available living spaces, the Smart Room Accommodation and Tracking System contributes to an optimized accommodation process, creating a more comfortable and well-organized living environment for the academic community. This system not only alleviates the administrative burden associated with traditional, manual accommodation processes but also establishes a scalable framework that can be further expanded or customized to meet the evolving needs of educational institutions in the future.

8.2 Future Scope

In future enhancements of the Smart Room Accommodation and Tracking System, several features could be introduced to further improve its functionality and user experience. One potential addition is the implementation of advanced analytics to monitor room usage patterns, enabling administrators to optimize room allocation and predict future demand more accurately. Another feature could be the integration of a notification system that alerts users about upcoming room vacancies, booking confirmations, or payment reminders. This would enhance user engagement and ensure timely communication between administrators and residents.

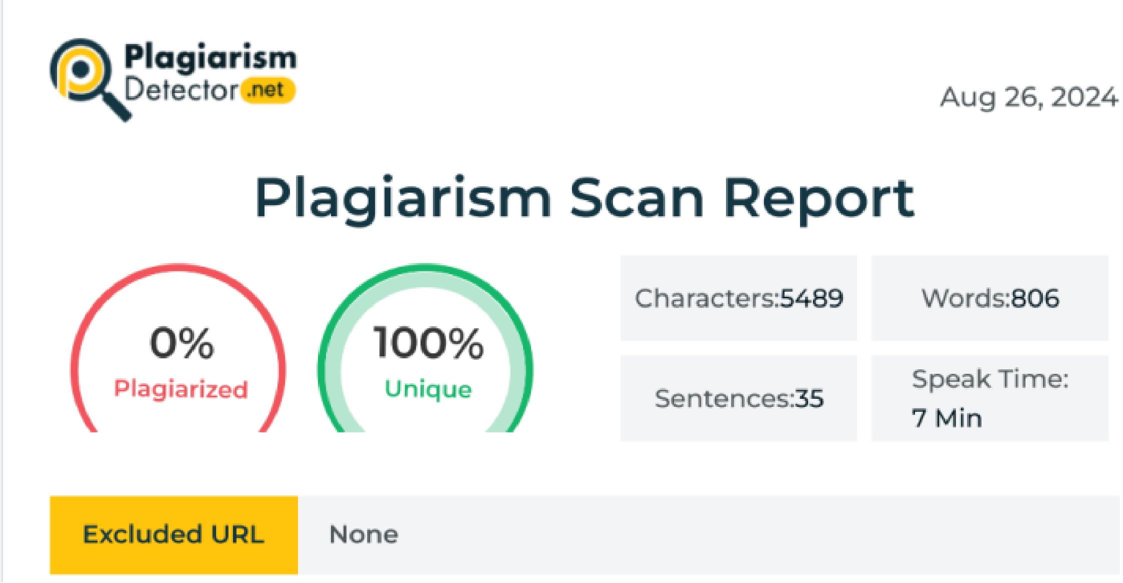
Furthermore, the system could be expanded to include mobile application support, allowing users to manage their accommodations on-the-go. Integration with other campus management systems, such as payment gateways or student portals, could provide a more seamless experience for users. Additionally, incorporating machine learning algorithms to predict and suggest optimal room allocations based on user preferences and historical data could significantly enhance the system's efficiency. These improvements would transform the platform into a comprehensive tool for managing all aspects of hostel accommodations, providing even greater value to the institution and its members.

In addition to these enhancements, incorporating features like **s**elf-service kiosks or QR code-based check-ins could further streamline the accommodation process, reducing administrative workload and making it more convenient for residents to access their rooms.

9. BIBLIOGRAPHY

1. **Visual Studio Code:** A code editor used for project development.
   URL: https://code.visualstudio.com/
2. **PHP**: A popular general-purpose scripting language that is especially suited to web development.
   URL: https://www.php.net/
3. **Website:**
   - Hostelworld website
     URL: https://www.hostelworld.com
   - CloudBed website
     URL: https://www.cloudbeds.com/
   - ResDiary website
     URL: https://resdiary.com/
4. **IEEE document:**
   - Hostel Management
     URL: https://ieeexplore.ieee.org/document/10040481
   - A Study on Shift towards Digitization of Hostel Room Allotment for a University
     URL: https://ieeexplore.ieee.org/document/9077688

10.PLAGIARISM REPORT



Plagiarism Detector.net

Aug 26, 2024

Plagiarism Scan Report

0% Plagiarized

100% Unique

Characters:5489	Words:806
Sentences:35	Speak Time: 7 Min

Excluded URL	None

Fig. Plagiarism report