1. PROJECT DESCRIPTION

1.1 INTRODUCTION

The Hostel Booking System simplifies the booking and management of hostel operations for both administrators and users. This system replaces outdated manual methods with a user-friendly computerized solution, saving considerable time and effort. Administrators have access to a range of functions, including managing staff and users effortlessly. They can easily monitor financial transactions, edit bookings as necessary, and access important documents such as photos and ID cards. Additionally, administrators can handle leave requests, respond to user queries promptly, and maintain records of past users. For users, the system offers convenient features to enhance their hostel experience. They can register and update their information with ease, ensuring accuracy. Users can also make rent payments conveniently, select preferred rooms based on availability, and upload essential documents. Moreover, they can access reports about their stay, request leave, and communicate with administrators for assistance. By leveraging modern technology, the Hostel Management System streamlines hostel operations and improves efficiency. It provides a seamless experience for both administrators and users, reducing manual workload and enhancing communication. Overall, the system offers a practical and effective solution for managing hostels in today's digital age.

1.2 EXISTING SYSTEM

The existing system is manual based and need lot of efforts and consume enough time. In the existing system the user can apply for the hostels online but the allotment processes are done manually. It may lead to corruptions in the allocation process as well as hostel fee calculation. The existing system does not deal with mess calculation, leave form applications and compliant registration.

DISADVANTAGES

- More human power and low security.
- Repetition of same procedure.
- Difficulty to handle, update data, and keeping records.
- Backup data cannot be easily generated.

1.3 PROPOSED SYSTEM

In the proposed system, using this application, a modern approach to streamline hostel operations, overcoming the limitations of the existing manual-based system. Through intuitive modules, users can securely register and apply for hostels online, while automated allotment processes ensure fairness and prevent corruption. The system also incorporates features for mess calculation, leave form applications, and complaint registration, enhancing overall functionality and user experience. With intuitive interfaces and centralized database management, the system facilitates easy updates, record-keeping, and seamless backup data generation for enhanced protection and efficiency.

ADVANTAGES

- Reduced reliance on human power
- Elimination of repetitive procedures through automation.
- Enhanced security measures to protect user data.
- It would less the manual process.
- Strength and strain of manual labor can be reduced.
- Easy to handle.

1.3.1 HARDWARE SPECIFICATION

- Processor : Intel Core i5 or equivalent
- RAM : 4GB
- Hard Disk Drive : 256GB
- Keyboard : Standard wired or wireless keyboard

1.3.2 SOFTWARE SPECIFICATION

- Operating system : Windows 7 or above
- Front End : HTML, CSS, JS
- Back end : PHP, MySQL
- Tool : WampServer
- MySQL Version : MySQL 8.0 or later.

1.3.3 SOFTWARE DESCRIPTION:

1.3.3.1 FRONT END – HTML:

HTML, known as HyperText Markup Language, serves as the fundamental markup language for crafting documents intended for web browser display. It governs the content and layout structure of web content, often complemented by technologies like Cascading Style Sheets (CSS) and scripting languages such as JavaScript. When web browsers receive HTML documents from a web server or local storage, they render these documents into interactive web pages. Originally, HTML provided cues for appearance alongside its structural description.

HTML elements function as the cornerstone of HTML pages, allowing the embedding of images, interactive forms, and other objects within the rendered page. These elements enable the creation of structured documents by defining semantic structures for text elements like headings, paragraphs, lists, links, quotes, and more. Enclosed within angle brackets, HTML tags such as <img> and <input> directly introduce content onto the page, while tags like and encapsulate and offer information about document text, potentially incorporating sub-element tags. Though browsers do not display HTML tags, they interpret them to render page content.

HTML can integrate programs scripted in languages such as JavaScript, influencing web page behavior and content. CSS inclusion dictates content appearance and layout. The World Wide Web Consortium (W3C), former steward of HTML and current guardian of CSS standards, has advocated for CSS usage over explicit presentational HTML since 1997. HTML5, a form of HTML, is employed for displaying video and audio content, primarily utilizing the <canvas> element in tandem with JavaScript.

1.3.3.2 FRONT END – CSS:

Cascading Style Sheets (CSS) serves as a style sheet language utilized to define the presentation and aesthetics of documents composed in markup languages like HTML or XML (including XML-based dialects like SVG, MathML, or XHTML). It stands as a fundamental technology alongside HTML and JavaScript in shaping the World Wide Web.

CSS is engineered to facilitate the segregation of content and presentation, encompassing aspects such as layout, colors, and typography. This segregation enhances content accessibility, offers greater flexibility and precision in specifying presentation attributes, allows multiple web pages to share formatting by referencing a separate .css file, thereby reducing redundancy and complexity in structural content. Moreover, caching the .css file enhances page load speed across pages sharing the same formatting. By separating formatting from content, it becomes feasible to display the same markup page in varied styles for diverse rendering methods, including on-screen, in print, via voice (using speech-based browsers or screen readers), and on tactile Braille-based devices. CSS also incorporates rules for alternate formatting when accessed on mobile devices.

The term "cascading" in CSS denotes a priority scheme employed to determine which declaration takes precedence if multiple declarations of a property match a particular element. This cascading priority scheme adheres to a predictable order. The CSS specifications are upheld by the World Wide Web Consortium (W3C), and the Internet media type (MIME type) text/css is officially registered for CSS usage by RFC 2318 (March 1998). W3C provides a complimentary CSS validation service for CSS documents. Beyond HTML, CSS finds utility in various markup languages, including XHTML, plain XML, SVG, and XUL. It is also integrated into the GTK widget toolkit.

1.3.3.3 FRONT END – JAVASCRIPT:

JavaScript stands as a dynamic and versatile programming language pivotal to contemporary web development. Its primary aim is to augment web pages with interactivity and responsiveness, complementing the structural layout governed by HTML and the styling defined by CSS. Operating chiefly on the client side, JavaScript interacts with the Document Object Model (DOM), empowering developers to dynamically manipulate and update web page content, appearance, and behavior in real-time. By reacting to user-triggered events such as clicks, keystrokes, or mouse movements, JavaScript facilitates the creation of interactive elements like forms, sliders, and dynamic menus, thus delivering a compelling and immersive user experience. An integral feature of JavaScript lies in its support for asynchronous programming, facilitated through mechanisms such as callbacks, promises, and async/await.

This enables developers to execute non-blocking code, crucial for tasks like fetching data from servers, performing animations, or managing user input without causing interruptions to the browsing experience. Moreover, JavaScript's versatility extends beyond client-side scripting, as it finds utility in server-side development with platforms like Node.js. This fosters the development of full-stack web applications, where JavaScript operates on both client and server sides, promoting code reuse and bolstering developer efficiency.

The influence of JavaScript transcends conventional web development, permeating fields such as mobile app development, game development, and even desktop applications through frameworks like Electron. Supported by a rich ecosystem of libraries and frameworks including jQuery, React, Angular, and Vue.js, developers gain access to potent tools and resources that streamline development processes and facilitate the creation of sophisticated applications. As an indispensable component of the web development stack, JavaScript continually evolves to meet the evolving demands of modern web development, reaffirming its status as one of the most crucial languages for crafting dynamic and engaging web experiences.

1.3.3.4 BACK END – PHP:

PHP, an abbreviation for Hypertext Preprocessor, stands as a server-side scripting language that has played an integral role in web development since its inception. Its primary function revolves around facilitating the creation of dynamic web pages and applications. Unlike HTML, which is dedicated to content structuring, and CSS, which manages presentation, PHP focuses on the logic and functionality underpinning web pages. This implies that PHP empowers developers to engage with databases, handle form submissions, manage user sessions, and execute various other server-side operations essential for constructing dynamic and engaging websites.

One of PHP's notable strengths lies in its seamless integration with HTML. PHP code seamlessly embeds within HTML files, enabling developers to blend dynamic content generation and logic directly within the markup. Such integration fosters the development of dynamic web pages where content dynamically adjusts based on user interactions, database queries, or other conditional factors. Moreover, PHP showcases remarkable versatility, catering to a diverse array of web development needs, ranging from straightforward contact forms to intricate content management systems (CMS) and e-commerce platforms.

Furthermore, PHP boasts an extensive ecosystem comprising libraries, frameworks, and tools that augment its capabilities and streamline development workflows. Leading PHP frameworks such as Laravel, Symfony, and CodeIgniter equip developers with robust architectures, integrated security features, and efficient development methodologies, facilitating the creation of scalable and maintainable web applications. Additionally, PHP benefits from an active and thriving developer community, contributing to its continuous advancement, providing assistance, and sharing insights and best practices. This vibrant community ensures PHP's relevance and effectiveness as a potent tool in web development, amidst the perpetual evolution of the internet landscape.

1.3.3.5 BACK END – SQL:

Structured Query Language (SQL) is a universal language used for managing relational databases. It provides developers and database administrators with a set of commands called queries to perform various tasks like creating, retrieving, updating, and deleting data. These queries act as a bridge between users and the database management system (DBMS), allowing them to interact with databases easily. SQL offers a straightforward and effective way to work with databases, offering features like data manipulation, transaction control, and user access management. It uses a declarative syntax, which means developers can specify what they want to achieve without worrying about the technical details of how it's done.

MySQL, on the other hand, is an open-source relational database management system (RDBMS) that uses SQL as its primary language. It's developed by Oracle Corporation and widely used in web development. MySQL is known for its scalability and reliability, making it a popular choice for storing and managing data. It supports various storage engines tailored to different needs and seamlessly integrates with other web development technologies like PHP, Python, and Node.js. This makes MySQL ideal for building database-driven web applications, e-commerce platforms, and content management systems.

In summary, SQL and MySQL are essential tools for managing relational databases, allowing developers to create resilient and scalable applications. They are widely used in web development due to their accessibility and extensive documentation, making them suitable for developers of all skill levels.

1.3.3.6 TOOL – WAMPSERVER:

WampServer is a software stack that facilitates the deployment of web applications locally on a Windows operating system. The name "WampServer" is derived from its components: Windows, Apache, MySQL, and PHP (or sometimes Perl or Python). These components work together to create a local web server environment, allowing developers to build and test websites and web applications before deploying them to a live server.

The main components of WampServer include:

1. Windows: The operating system on which WampServer runs.

2. Apache: An open-source web server software that serves as the backbone of the WampServer environment. Apache interprets HTTP requests from web browsers and serves the corresponding web pages.

3. MySQL: A relational database management system (RDBMS) that stores and manages data. MySQL is used to create and manage databases for web applications.

4. PHP: A server-side scripting language used for web development. PHP code can be embedded within HTML documents to create dynamic web pages.

WampServer provides a convenient and user-friendly way for developers to set up a local development environment. Once installed, developers can create and manage projects directly within the www directory of the WampServer installation. Each project typically consists of HTML, CSS, JavaScript, PHP, and other files and folders that make up the web application.

The primary use of WampServer is for local development and testing of web applications. Developers can write and debug code, test database queries, and preview changes in real-time without the need for an internet connection or a remote server. Once the development process is complete, the web application can be easily deployed to a live server for public access.

In addition to its use in web development, WampServer is also commonly used for educational purposes, allowing students to learn web development techniques in a controlled environment. It provides a safe and isolated space for experimenting with different technologies and frameworks without affecting live websites or databases. Overall, WampServer serves as an invaluable tool for developers and learners alike, simplifying the process of building and testing web applications locally.

1.3.3.7 TOOL – VISUAL STUDIO CODE:

Visual Studio Code (VS Code) is a versatile coding environment crafted by Microsoft, tailored to cater to the diverse needs of programmers across various platforms. Its adaptability extends to different operating systems like Windows, macOS, and Linux, ensuring accessibility for a wide range of users. What sets it apart is its extensive language support and the ability to enhance functionality through customizable extensions, empowering users to tailor their coding experience to their specific requirements.

One of the standout features of VS Code is its intelligent code understanding capabilities, which aid developers in writing code more efficiently. It provides syntax highlighting for better code readability, autocomplete suggestions for faster coding, and built-in error detection to streamline the debugging process. Additionally, it offers version control integration and built-in terminal access, simplifying tasks like code management and execution.

From a usability standpoint, VS Code excels with its intuitive interface, allowing users to easily navigate and customize their coding environment. Its simplicity doesn't compromise functionality, as it offers powerful features like integrated debugging and task automation.

2. LOGICAL DEVELOPMENT

2.1 ARCHITECTURAL DESIGN

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages (ADLs).

Various organizations define systems architecture in different ways, including:

An allocated arrangement of physical elements which provides the design solution for a consumer product or life-cycle process intended to satisfy the requirements of the functional architecture and the requirements baseline. Architecture comprises the most important, pervasive, top-level, strategic inventions, decisions, and their associated rationales about the overall structure (i.e., essential elements and their relationships) and associated characteristics and behavior. If documented, it may include information such as a detailed inventory of current hardware, software and networking capabilities; a description of long-range plans and priorities for future purchases, and a plan for upgrading and/or replacing dated equipment and software The composite of the design architectures for products and their life-cycle processes.

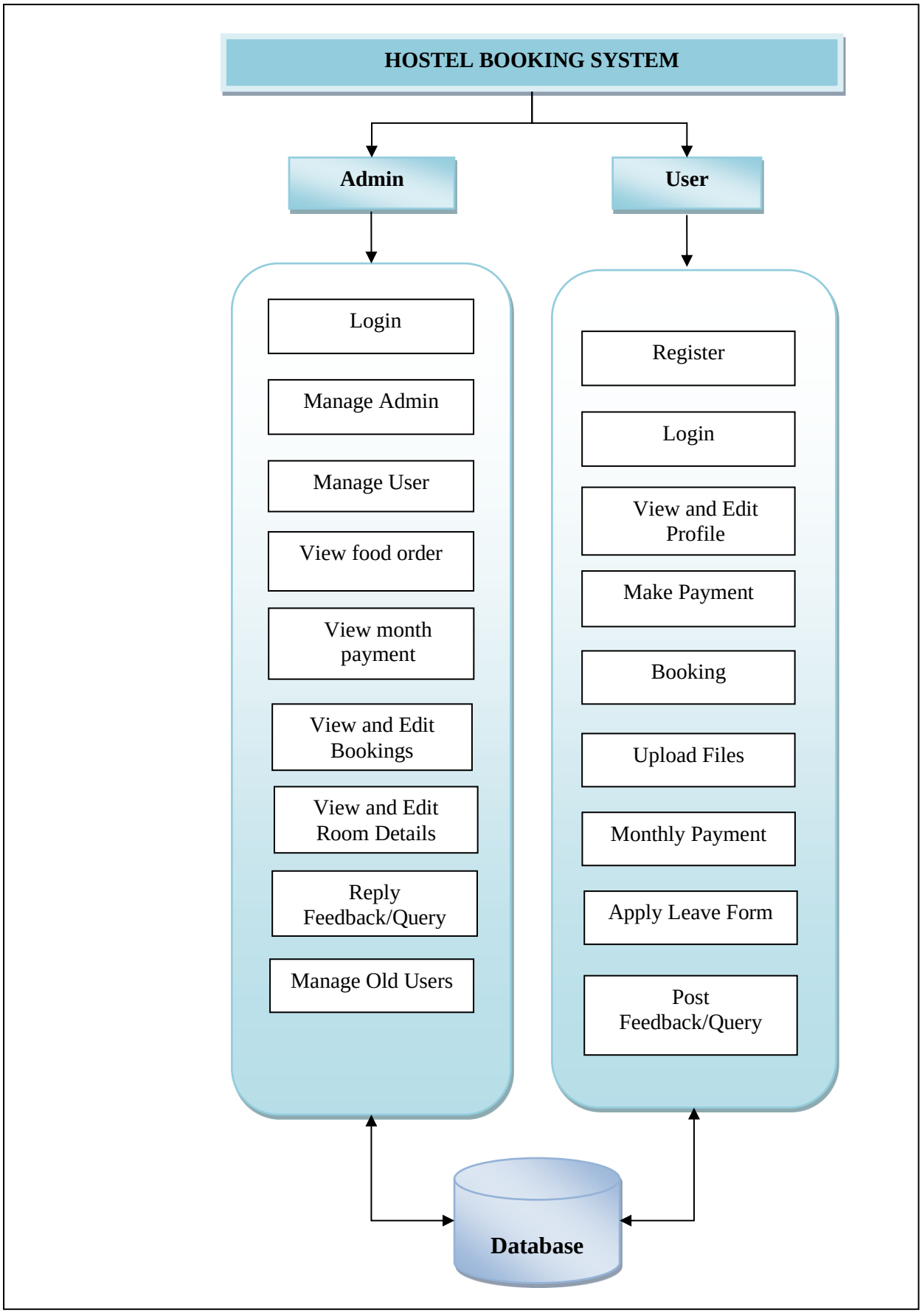


Figure 2.1 Architectural Design

2.2 DATAFLOW DIAGRAM

A two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

2.2.1 LEVEL 0

The Level 0 DFD shows how the system is divided into 'sub-systems' (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.

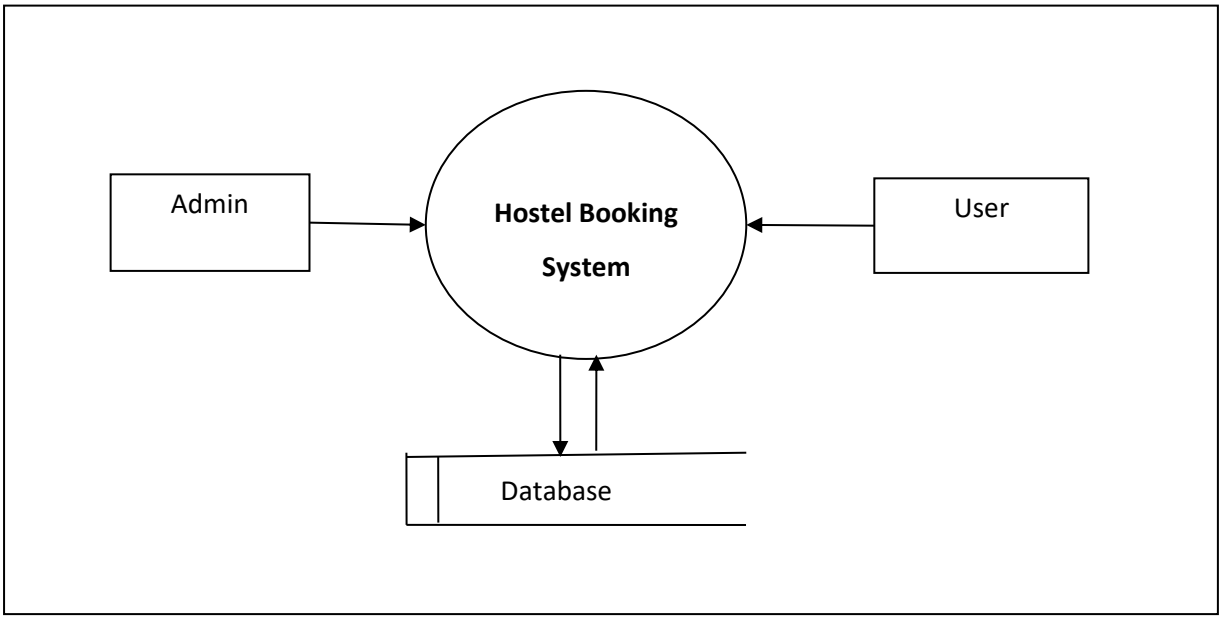


Figure 2.2.1 level 0

2.2.2 LEVEL 1

The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, to describe the system was using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.

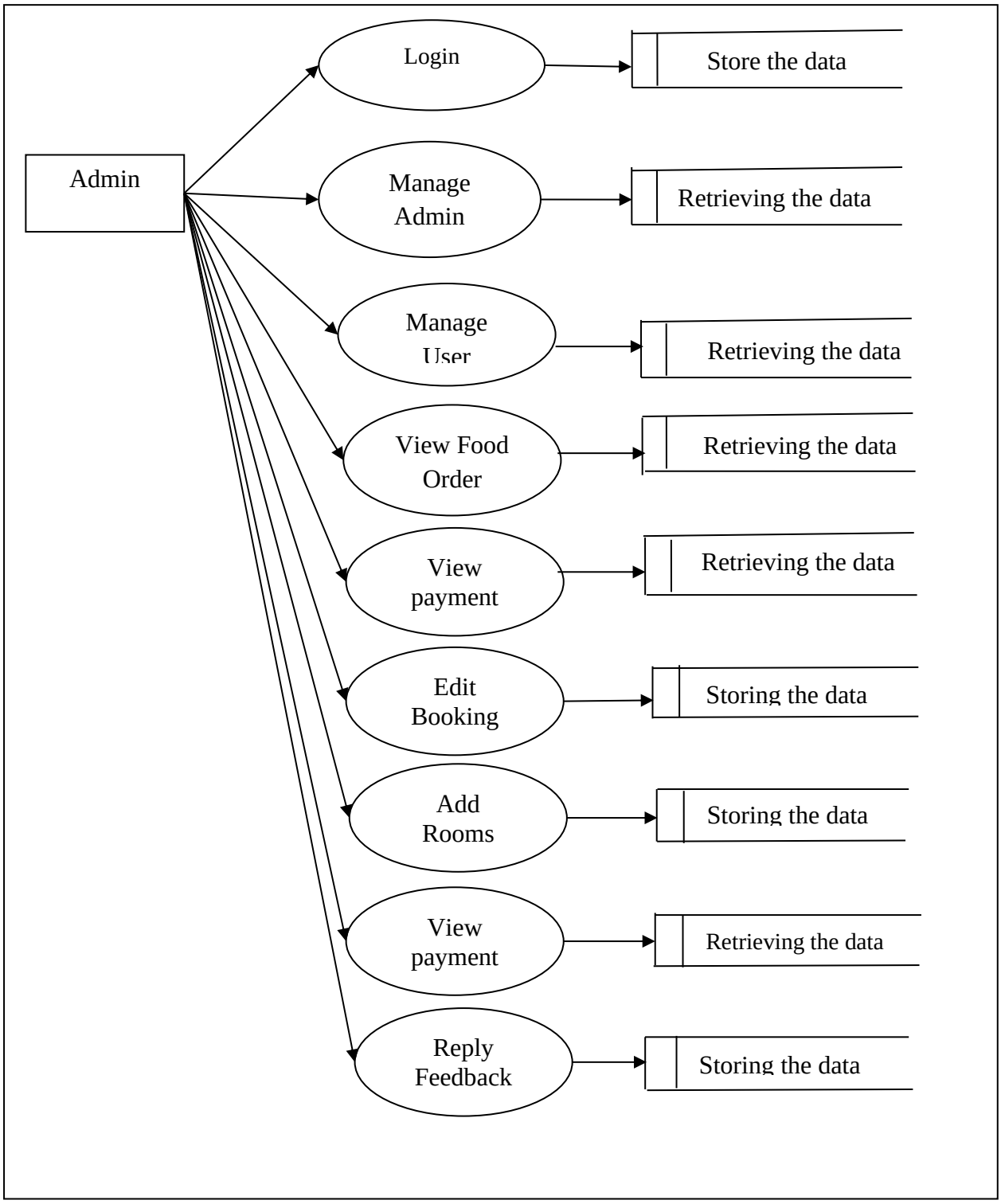


Figure 2.2.2 level 1

2.2.3 LEVEL 2

DFD goes one step deeper into parts of 1-level DFD. It can be used to plan or record the specific/necessary detail about the system's functioning. A DFD defines each domain boundary and illustrates the logical movement and transformation of data within the defined boundary. Higher level DFDs allow you to dive deeper into each step and flow of information.

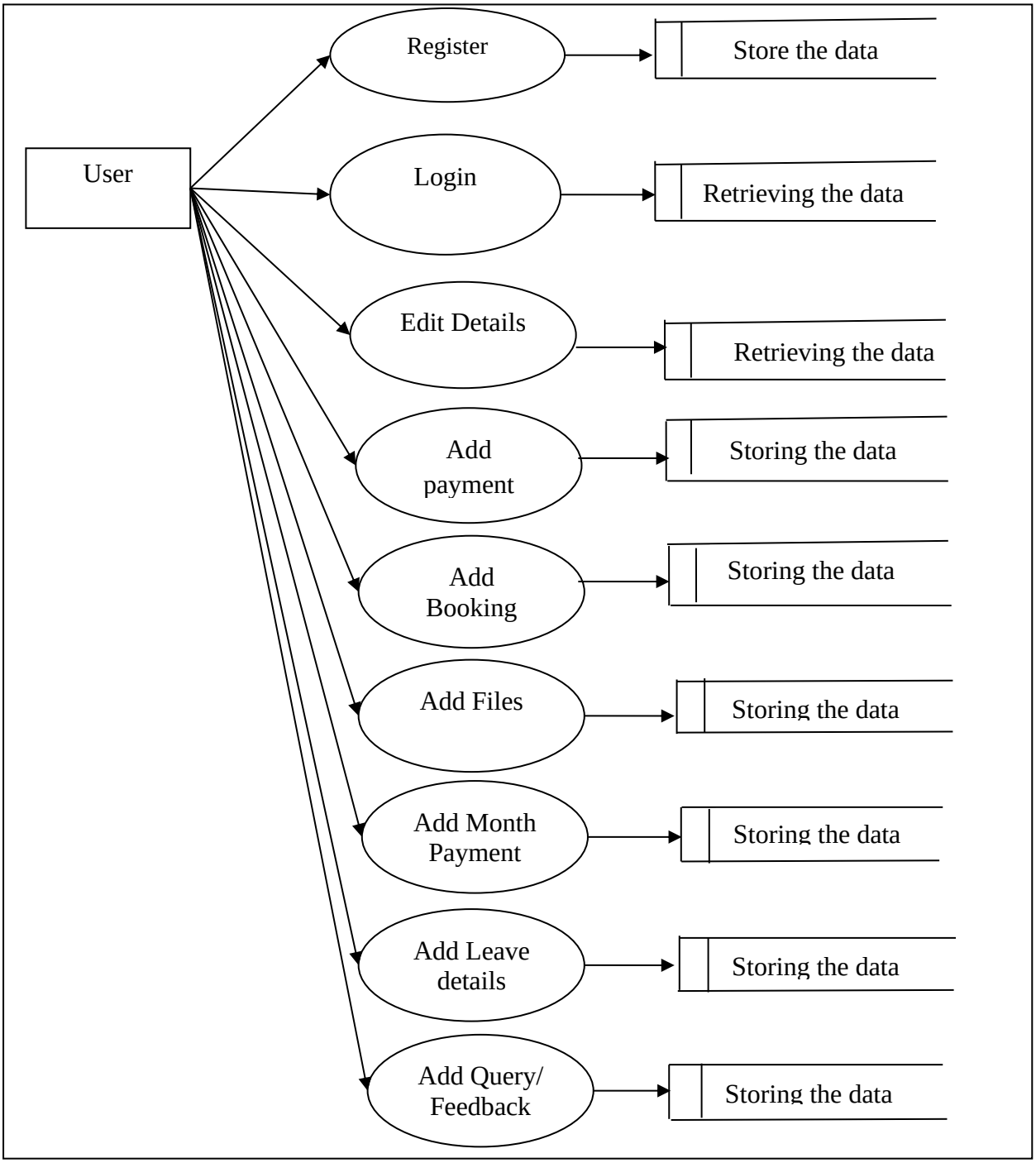


Figure 2.2.3 level 2

3. DATABASE DESIGN

3.1 TABLES DESIGN

A table is a data structure that organizes information into rows and columns. It can be used to both store and display data in a structured format. For example, databases store data in tables so that information can be quickly accessed from specific rows. Websites often use tables to display multiple rows of data on page. Spreadsheets combine both purposes of a table by storing and displaying data in a structured format.

Databases often contain multiple tables, with each one designed for a specific purpose. For example, a company database may contain separate tables for employees, clients, and suppliers. Each table may include its own set of fields, based on what data the table needs to store. In database tables, each field is considered a column, while each entry (or record), is considered a row. A specific value can be accessed from the table by requesting data from an individual column and row.

3.1.1 Table Name: admins

Column	Type	Constraints
id	int(25)	PRIMARY KEY
name	varchar(35)	NOT NULL
email	varchar(35)	NOT NULL
password	varchar(25)	NOT NULL

3.1.2 Table Name: archived_users

Column	Type	Constraints
id	int(25)	PRIMARY KEY
name	varchar(35)	NOT NULL
father_name	varchar(35)	NOT NULL
dob	date	NOT NULL
address	varchar(75)	NOT NULL
phone_number	varchar(15)	NOT NULL
email	varchar(25)	NOT NULL
password	varchar(25)	NOT NULL
booking_id	int(25)	NULL
start_date	date	NULL
duration	int(25)	NULL
food_status	int(2)	NULL
guardian_name	varchar(45)	NULL
relation	varchar(45)	NULL
guardian_contact	varchar(12)	NULL
emergency_contact	varchar(12)	NULL
total_fees	int(25)	NULL
selected_room	varchar(20)	NULL
created_at	timestamp	NULL
transaction_id	varchar(50)	NULL
payment_date	timestamp	NULL
photo_filename	varchar(35)	NULL
id_proof_filename	varchar(35)	NULL

3.1.3 Table Name: bookings

Column	Type	Constraints
booking_id	int(25)	PRIMARY KEY
user_id	int(25)	FOREIGN KEY
start_date	date	NOT NULL
duration	int(25)	NOT NULL
food_status	int(2)	NOT NULL
guardian_name	varchar(45)	NOT NULL
relation	varchar(45)	NOT NULL
guardian_contact	varchar(12)	NOT NULL
emergency_contact	varchar(12)	NOT NULL
total_fees	int(25)	NOT NULL
selected_room	varchar(20)	NOT NULL
created_at	timestamp	NOT NULL

3.1.4 Table Name: foods

Column	Type	Constraints
id	int(25)	PRIMARY KEY
foodname	varchar(25)	NOT NULL
price	int(5)	NOT NULL
photo_name	varchar(35)	NOT NULL

3.1.5 Table Name: food_order

Column	Type	Constraints
id	int(25)	PRIMARY KEY
user_id	int(25)	FOREIGN KEY
foodname	varchar(25)	NOT NULL
foodcount	int(5)	NOT NULL
price	int(5)	NOT NULL
bf_lun_din	varchar(15)	NOT NULL
created_at	timestamp	NOT NULL

3.1.6 Table Name: leave_form

Column	Type	Constraints
id	int(25)	PRIMARY KEY
user_id	int(25)	FOREIGN KEY
name	varchar(35)	NOT NULL
room	varchar(20)	NOT NULL
start_date	datetime	NOT NULL
return_date	datetime	NOT NULL
place	varchar(35)	NOT NULL
purpose	varchar(50)	NOT NULL
parents_number	varchar(20)	NOT NULL
message	varchar(100)	NOT NULL
created_at	timestamp	NOT NULL

3.1.7 Table Name: payment

Column	Type	Constraints
payment_id	int(25)	PRIMARY KEY
user_id	int(25)	FOREIGN KEY
transaction_id	varchar(50)	NOT NULL
payment_date	timestamp	NOT NULL

3.1.8 Table Name: month_payment

Column	Type	Constraints
id	int(25)	PRIMARY KEY
user_id	int(25)	FOREIGN KEY
name	varchar(35)	NOT NULL
monthlyfees	int(10)	NOT NULL
total_mess_food_price	int(10)	NOT NULL
total_amount	int(10)	NOT NULL
transaction_id	varchar(50)	NOT NULL
created_at	timestamp	NOT NULL

3.1.9 Table Name: room

Column	Type	Constraints
room_number	varchar(50)	PRIMARY KEY
total_seats	int(5)	NOT NULL
occupied_seats	int(5)	NOT NULL
available_seats	int(5)	NOT NULL
image_path	varchar(50)	NOT NULL

3.1.10 Table Name: users

Column	Type	Constraints
id	int(25)	PRIMARY KEY
name	varchar(35)	NOT NULL
father_name	varchar(35)	NOT NULL
dob	date	NOT NULL
address	varchar(75)	NOT NULL
phone_number	varchar(15)	NOT NULL
email	varchar(25)	NOT NULL
password	varchar(25)	NOT NULL

3.1.11 Table Name: user_files

Column	Type	Constraints
id	int(25)	PRIMARY KEY
user_id	int(25)	FOREIGN KEY
photo_filename	varchar(35)	NOT NULL
id_proof_filename	varchar(35)	NOT NULL

3.1.12 Table Name: user_query

Column	Type	Constraints
id	int(25)	PRIMARY KEY
user_id	int(25)	FOREIGN KEY
query_text	varchar(100)	NOT NULL
admin_reply	varchar(100)	NOT NULL

3.2 DATA DICTIONARY

FIELD NAME	TYPE	DESCRIPTION	SAMPLE VALUES
email	varchar(25)	Specify the email id	ramki@gmail.com
password	varchar(25)	Specify the password	**********
id	int(25)	Specify the id	2
name	varchar(35)	Specify the name	Ramakrishnan
father_name	varchar(35)	Specify the father name	Murugadasan
dob	date	Specify the date of birth	17/05/2002
address	varchar(75)	Specify the address	Kumbakonam
phone_number	varchar(15)	Specify the phone number	9685985697
start_date	date	Specify the start date	14/02/2024
duration	int(25)	Specify the duration	12
food_status	int(2)	Specify the food status	1
guardian_name	varchar(45)	Specify the guardian name	Ravi
relation	varchar(45)	Specify the relation	Uncle
guardian_contact	varchar(12)	Specify the guardian number	9865784598
emergency_contact	varchar(12)	Specify the parents number	9652451258
selected_room	varchar(20)	Specify the room selected	room2
transaction_id	varchar(50)	Specify the transaction id	Pnb_UThi2341uy
photo_filename	varchar(35)	Specify the photo name	personal.jpg
id_proof_filename	varchar(35)	Specify the id proof file name	idcard.pdf

3.3 RELATIONSHIP DIAGRAM

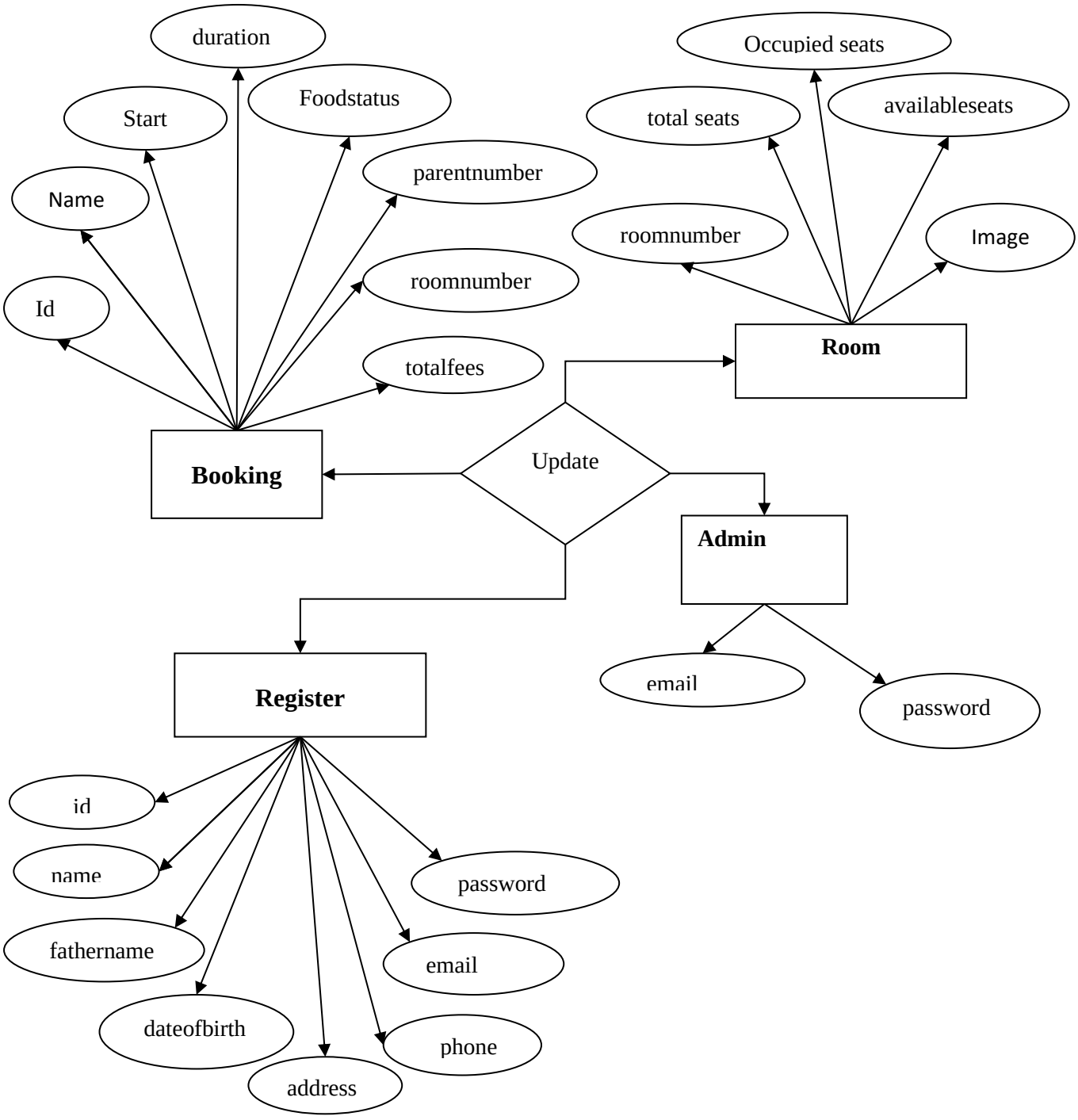


Figure 3.3 relationship diagram

4. PROGRAM DESIGN

The project “Hostel Booking System” is designed to facilitate efficient management and interaction within a system to both administrators and users. Admins have the ability to oversee user accounts, handle bookings, manage payments, monitor room details, and respond to queries and feedback. Additionally, they can handle administrative tasks such as managing other admin accounts and dealing with old user accounts. On the other hand, users can register, log in, and access functionalities tailored to their needs, such as viewing profiles, making bookings, managing payments, uploading files, generating reports, exploring food options, submitting leave requests, and providing feedback.

4.1 MODULES

The project “Hostel Booking System” is divided into Two modules: Admin and User. Admin have the ability to manage user and admin, edit booking, view and manage rooms, payment and reply user query/feedback. User have the ability to book a room, payment, upload id proof, apply leave form and send query/feedback to the admin

4.1.1 Admin

- Login
- Manage Admin
- Manage User
- View Food Order Details
- View Monthly Payment Details
- Edit Booking
- View User Files
- View and Manage Room Details
- View User Reports
- View Leave Form Details
- View and Reply Query/Feedback
- Manage Old Users

4.1.2 User

- Register
- Login
- View Profile
- Payment
- Booking
- Upload Files
- Report
- Food Details
- Monthly Payment
- Leave Form
- Query/Feedback

4.2 MODULE DESCRIPTION

4.2.1 ADMIN MODULE

4.2.1.1 Login:

This module enables secure login functionality for administrators, ensuring access control and system security. It validates user credentials, providing access to respective accounts and functionalities based on roles. If a matching record is found in the database for the provided email, the script retrieves the corresponding password hash from the database. It then compares the entered password with the stored password hash. If they match, the login is considered successful; otherwise, it indicates invalid credentials.

If the login credentials are valid, the script starts a session and sets session variables to indicate that the admin is logged in. These session variables can be used to authenticate the admin's access to protected pages in subsequent requests. Upon successful login, the script redirects the admin to the admin dashboard page using the header function. This prevents unauthorized access to the admin area and ensures a smooth user experience.

4.2.1.2 Manage Admin:

Administrators can efficiently handle administrative accounts within the system, including adding, editing, or removing admin privileges. This module empowers administrators to maintain an updated and organized administrative structure, ensuring smooth operation of hostel management tasks. At the beginning of the script, there's a check to ensure that the admin is authenticated. If the session variable is not set, indicating that the admin is not logged in, the script redirects the user to the admin login page. This prevents unauthorized access to the admin dashboard and other admin-related functionalities. After retrieving admin data from the database, the script displays it in a tabular format on the admin dashboard.

It lists admin IDs, names, emails, passwords, and provides options to edit or remove admin accounts. This presentation allows admins to manage other admin accounts efficiently. The script includes client-side search functionality implemented using JavaScript. Admins can search for specific admin accounts by ID or name, and the table dynamically updates to display only matching results. This feature enhances usability and facilitates efficient navigation of admin data.

4.2.1.3 Manage User:

Administrators can manage user accounts, including registration details and access permissions, ensuring accurate user information and system security. This module enables administrators to oversee user interactions, track user activities, and address any issues or concerns promptly. The script begins by checking if the session variable is set. If it's not set, it means the admin is not logged in. In such cases, the script redirects the user to the admin login page using the header function. This mechanism ensures that only authenticated admins can access the user management section. The script executes a database query to fetch user data from the users table. This query retrieves information about users stored in the database, which will be displayed in the user management interface.

After retrieving user data from the database, the script displays it in a tabular format within the user management interface. Each row of the table represents a user, displaying various user attributes such as ID, name, father's name, date of birth, address, phone number, email, and password. Admins can view, edit, or remove user accounts as needed. The script includes client-side search functionality implemented using JavaScript. Admins can search for specific

users by name or ID, and the table dynamically updates to display only matching results. This feature enhances usability and facilitates efficient navigation of user data.

4.2.1.4 View Food Order Details:

Administrators can easily access and review detailed information about food orders placed by users, facilitating efficient management of food services. This module provides insights into food preferences, order history. It supports effective planning and optimization of food service operations, ensuring timely and satisfactory delivery of meals to hostel residents. The first check verifies if the admin is logged in. If not, it redirects to the admin login page. This suggests that authentication is based on a session variable, which is likely set upon successful login. The script executes a SQL query to fetch data from the users and bookings tables, joining them based on the user_id. This query retrieves information about users and their selected rooms. After retrieving the user's name based on the provided user ID, it displays the user's name along with their ID to provide context for the displayed food orders. It fetches detailed food order information based on the provided user ID and displays it in a table format. Each row in the table includes an option to remove the corresponding food order.

4.2.1.5 View Monthly Payment Details:

Administrators can monitor and track monthly payments made by users for hostel accommodations, ensuring financial transparency and accountability. This module provides administrators with detailed insights into payment statuses, and payment histories, facilitating effective financial management. It supports timely reconciliation of payments, minimizes discrepancies, and ensures uninterrupted hostel services for users. it checks if the variable is set. If not, it redirects the user to the adminlogin page, ensuring that only authenticated admins can access this page. It executes an SQL query to select all records from the month_payment table, presumably containing monthly payment details for users. It displays the retrieved data in an HTML table format. However, there's no explicit data validation or sanitization done before displaying the data

4.2.1.6 Edit Booking:

Administrators have the flexibility to modify existing room bookings based on user requests or changes in availability, ensuring accommodation preferences are met. This module enables administrators to efficiently handle booking amendments, including date changes, room upgrades, or cancellations, while maintaining accurate records. It supports personalized service

delivery, enhances user satisfaction, and contributes to a seamless booking experience for hostel residents. The script checks if the variable is set. If not, it redirects the user to the adminlogin page, ensuring that only authenticated admins can access this page. It displays the booking details in a table format. The script provides a search functionality based on the user ID. It allows the admin to search for bookings associated with a specific user ID. The script provides options to edit and remove bookings. These options include links to edit_bookings and delete_bookings passing the user ID as a parameter. The script includes a confirmation dialog before executing the remove action, enhancing user experience and preventing accidental removal.

4.2.1.7 View User Files:

Administrators can access and review user-uploaded files such as photos and ID proofs for verification purposes, ensuring compliance with hostel policies and regulations. This module facilitates efficient verification processes, enhances security measures, and ensures the authenticity of user information. It supports smooth onboarding procedures, fosters trust and transparency, and contributes to a safe and secure hostel environment. The script first checks if the variable is set. If not, it redirects the user to the adminlogin page, ensuring that only authenticated admins can access this page. It queries the database to fetch user files information, including the user's name, ID, photo filename, and proof filename. The script displays the user files information in a table format. It includes the user's name, ID, and links to view the photo and proof files. The search functionality directly uses the input provided by the admin.

4.2.1.8 View and Manage Room Details:

Administrators can view comprehensive information about available rooms, including occupancy status, amenities, and pricing details, facilitating efficient room management. This module enables administrators to add, edit, or remove room listings as needed, ensuring accurate and up-to-date room. It supports effective room allocation, optimization of occupancy rates, and enhances user satisfaction by providing relevant and transparent room information. Similar to previous scripts, the script first checks if the variable is set. If not, it redirects the user to the adminlogin page, ensuring that only authenticated admins can access this page. It retrieves the selected room number from the form submission using.

The selected room number is then assigned to the selected room variable. The script populates the dropdown menu with room numbers fetched from the database. It loops through the result set obtained from the database query for room numbers and dynamically generates option elements for each room number.

If a room is selected (either previously selected or selected after form submission), it marks that option as selected. If booking details are available for the selected room, it displays the booking details in a table format. It iterates through the result set of booking details and generates table rows for each booking record, displaying user ID, user name, start date, duration, and selected room number. If there are no booking details available for the selected room, it displays a message indicating that there are no bookings for that room.

4.2.1.9 View User Reports:

Administrators can generate and view detailed reports regarding user activities, user information and transactions, enabling data-driven decision-making. The script begins by checking if the variable is set, ensuring that only authenticated admins can access this page. If not set, the script redirects the user to the adminlogin page. The script constructs an SQL query to fetch user report details, including user information, booking details, payment details, and user files.

The query joins multiple tables (users, bookings, user_files, and payment) to retrieve the required information. For each row in the result set, it dynamically generates HTML elements to display the user report details in a tabular format. It also includes a button for downloading the report in PDF format for each user report entry. The script includes JavaScript functions for toggling the sidebar, downloading the user report in PDF format, and searching/filtering user reports based on the user's input. The downloadReportPDF(id) function uses the html2pdf library to generate a PDF from the HTML content of the user report. The searchUser() function filters the displayed user reports based on the search input provided by the user. The showAll() function resets the display of user reports to show all entries.

4.2.1.10 View Leave Form Details:

Administrators can efficiently review leave requests submitted by users. This module provides administrators with detailed information about leave requests, including dates, reasons, and user preferences, facilitating effective leave management. It checks if the variable is set, ensuring that only authenticated admins can access this page. If not set, the script redirects the

user to the adminlogin page. It fetches data from the leave_form table to display leave form details. The script provides options to filter leave form entries by date and time. It retrieves the filter date and time values from the form submission 'filterDate' and 'filterTime'. It uses PHP DateTime class to compare the start and return dates of leave form entries with the filter date and time. If a filter date and time are provided, the script displays only those leave form entries that fall within the specified date and time range.

4.2.1.11 View and Reply Query/Feedback:

Administrators can access queries and feedback submitted by users, providing timely responses or resolutions to address user concerns. This module enables administrators to engage with users, gather valuable feedback, and maintain open communication channels. It supports continuous improvement efforts, fosters positive user relationships, and enhances overall user satisfaction. It checks if the variable is set, ensuring that only authenticated admins can access this page. If not set, it redirects the user to the adminlogin page. The script checks if the request method is POST, indicating that form data has been submitted.

It specifically checks for the presence of 'saveAdminReply', assuming it's an array containing admin replies indexed by query IDs. This suggests that the form is used to save admin replies to user queries. The script iterates over the submitted admin replies and updates them in the database. It constructs and executes SQL queries to update the admin reply field in the user query table based on the provided admin replies and query IDs. The script provides search functionality to filter user queries based on the user's name or ID. It includes JavaScript functions searchUsers and showAllUsers to toggle the visibility of table rows based on the search input.

4.2.1.12 Manage Old Users:

Administrators can efficiently manage past user accounts, including archiving or deleting inactive accounts, ensuring system cleanliness and efficiency. This module facilitates effective user account management, streamlines database operations, and optimizes system performance. It supports data hygiene practices, enhances system security, and ensures compliance with data protection regulations. It checks if the variable is set, ensuring that only authenticated admins can access this page. If not set, it redirects the user to the adminlogin page. It retrieves the admin ID from the session and fetches admin details from the database using a SQL query.

The script then executes a SQL query to select data from the archived_users table.The script provides search functionality to filter archived users based on their names or IDs. It includes JavaScript functions searchUsers and showAllUsers to toggle the visibility of table rows based on the search input. It dynamically generates HTML table rows to display the fetched archived user data. The script populates table cells with the corresponding data retrieved from the database.

4.2.2 USER MODULE:

4.2.2.1 Register:

Users can easily create accounts by providing necessary personal information, streamlining the registration process within the system. Before registering a new user, the module checks if the provided email already exists in the database. If a user with the same email is found, an alert is displayed, informing the user that the email has already been taken. After ensuring the email is unique, the module compares the entered password with the re-entered password to ensure they match. If the passwords do not match, an alert is displayed, prompting the user to re-enter the password correctly.

The module validates the phone number entered by the user using a regular expression (regex). If the phone number does not match the specified format (e.g., (123) 456-7890), an alert is displayed, indicating that the user should enter a valid phone number. Similarly, the module validates the email address using a regex pattern. If the email address is invalid (i.e., does not match the expected format), an alert is displayed, prompting the user to enter a valid email address. The module enforces password strength requirements using a regex pattern. If the entered password does not meet the specified criteria (e.g., at least 8 characters long, containing at least one uppercase letter, one lowercase letter, one number, and one special character), an alert is displayed, guiding the user to choose a stronger password.

4.2.2.2 Login:

Secure login functionality is provided for users to access their accounts and avail of hostel services, ensuring access control and system security. This module verifies if a user is who they claim to be by checking their email and password against stored data. If the credentials match, the user is considered authenticated and logged in. Before processing the login attempt, this module ensures that the email and password fields are filled out by the user. This prevents

submission of empty fields and prompts the user to provide necessary information. This module further validates the user's input by checking if the provided email exists in the database and if the corresponding password is correct. This prevents unauthorized access and ensures data integrity. If the submitted email doesn't exist in the database or the password is incorrect, error messages are displayed to inform the user. This guides users in correcting their input and improves the overall user experience.

4.2.2.3 View Profile:

Users can view and update their profile information, ensuring accuracy and completeness of personal details within the system. This module empowers users to manage their profiles effectively, enhancing control over their account information and preferences. This module checks if the session variable is set. If it's not set, it redirects the user to the login page. This ensures that only authenticated users can access the user dashboard. However, the module could be improved by also checking the validity of the session. For example, verifying if the user's session is still active and not expired.

4.2.2.4 Payment:

Users can make payments for their booked accommodations securely within the system, ensuring timely and hassle-free transactions. This module facilitates payment methods, providing flexibility and convenience for users to complete their payments. The module checks if the session variable is set. If it's not set, the user is redirected to the login page. This ensures that only authenticated users can access the payment page. This helps prevent unauthorized access to sensitive payment functionality. Authentication occurs when a user logs in and their credentials are verified against the database. After successful authentication, a session is created to keep the user logged in across different pages. The module retrieves the user's ID from the session to associate the payment with the logged-in user. The user's name is also fetched from the database based on their ID and displayed on the page to provide a personalized experience. After submitting a payment, the module should prevent the user from submitting the same transaction ID again to avoid duplicate payments. Currently, the module displays an alert message if the payment is already recorded. However, it should also disable the input field for the transaction ID or provide a message indicating that the payment has already been made, thus preventing the user from submitting the same transaction ID again. This prevents accidental or intentional duplicate payments and ensures data integrity in the system.

4.2.2.5 Booking:

Users have access to detailed information about available rooms, amenities, and pricing, facilitating informed decision-making for accommodation bookings. This module enables users to conveniently book hostel accommodations based on their preferences and availability, ensuring a seamless booking experience. It supports personalized booking options, enhances user satisfaction, and contributes to a positive hostel experience. The module begins with session validation to check if the user is logged in. If the variable is not set, the user is redirected to the login page. This ensures that only authenticated users can access the booking page. After validating the user, the module checks if the user has made any transactions. This ensures that only users with completed transactions can proceed with booking a room.

The module checks if the user has already made a booking. If a booking already exists, it prevents the user from making duplicate bookings. Before inserting the booking details into the database, the module checks if the selected room has available seats. If the selected room has no available seats, the user is alerted, and the booking process is halted. The module calculates the total fees based on the selected duration and food status. It updates the total fees dynamically on the page using JavaScript.

4.2.2.6 Upload Files:

Users can upload essential documents such as photos and ID proofs for verification purposes, ensuring compliance with hostel policies and regulations. This module facilitates seamless document submission processes, enhancing security measures and ensuring the authenticity of user information. The module begins with session validation. It checks if the session variable is set. If not, it redirects the user to the login page.

This ensures that only authenticated users can access the upload files page. The module performs input validation to ensure that both the photo and ID proof files are selected before proceeding with the upload.It checks if the connection to the database is successful and if the user exists in the database. If the user already has uploaded files (checked through the user_files table), it prevents duplicate uploads by displaying an alert message and exits the script.When the user selects a photo or ID proof file, JavaScript functions (previewImage and previewPdf) are triggered to preview the selected files before uploading.

These functions validate the file types and sizes. For example, the photo file should be an image with a size between 50KB and 150KB, while the ID proof file should be a PDF with a size between 50KB and 2MB. If the selected file does not meet these criteria, an alert message is

displayed, and the file preview is hidden. If the user attempts to upload files again after already uploading them once, the module detects this scenario and prevents duplicate uploads by displaying an alert message and terminating further processing.

4.2.2.7 Report:

Users can access reports about their bookings, payments, and other activities within the system, providing valuable insights into their interactions and transactions. there's a check to ensure that a session is active. If the variable is not set, the user is redirected to the login page. This ensures that only authenticated users can access the authentication page. The user is required to enter both user ID and password. These fields are checked to ensure that they are not empty before processing the authentication request upon successful authentication, the script retrieves and displays user information from the database, including details such as name, father's name, date of birth, phone number, booking details, payment details, and transaction details. The user's information is displayed in an HTML table format. Additionally, there's functionality to download the displayed user information as a PDF. When the "Download PDF" button is clicked, the content of the displayed user information is converted into a PDF using the html2pdf library, and the PDF file is saved.

4.2.2.8 Food Details:

Users can view detailed information about food options available in the hostel, facilitating meal planning and dietary preferences. This module provides insights into menu offerings, pricing, and meal schedules, enabling users to make informed choices and enjoy satisfying dining experiences. The script starts by checking if a user is logged in by verifying the presence of the variable. This prevents unauthorized access to the food ordering page. Once the user is authenticated, the script retrieves the user's name and ID from the session and uses them to query the database for additional user information, such as their previous food orders and available food items. Before processing each food order, the script validates the submitted food count to ensure it is greater than zero. If the count is not greater than zero, an error message is displayed. For each valid food order, the script retrieves the details of the selected food item from the database, calculates the total price, and inserts the order into the database using SQL INSERT queries. After processing the form submission, if the order is successfully placed, an alert message is displayed to the user indicating that the order was placed successfully.

4.2.2.9 Monthly Payment:

Users can conveniently make monthly payments for hostel accommodations within the system and ensuring timely. This module facilitates recurring payments, providing users with a convenient and predictable payment schedule for their accommodation expenses. The script begins by checking if the user is authenticated by verifying the presence of the variable. This ensures that only authenticated users can access the payment page. After authentication, the script fetches the user's name and ID from the session and uses them to query the database for additional user information, such as their food booking status and previous month's food expenses. This data retrieval is essential for calculating the total payment amount. Before inserting a new payment record into the database, the script checks if a payment record already exists for the current month and user.

If a payment record is found, the script alerts the user that they have already paid for the current month, preventing duplicate payments. If no existing payment record is found, the script inserts the payment details into the month_payment table in the database. This includes the user's ID, name, monthly fees, total mess food price, total amount, and transaction ID. Successful insertion indicates that the payment was successfully submitted. Finally, the script fetches the user's payment records from the database and displays them in a table. This allows the user to view their payment history, providing transparency and accountability.

4.2.2.10 Leave Form:

Users can submit leave requests for periods of absence, providing administrators with detailed information for efficient leave management. This module facilitates leave request submission, ensuring timely processing and approval, and minimizing disruptions to hostel operations. The script begins by checking if the user is authenticated by verifying the presence of the variable. This ensures that only authenticated users can access the leave form page. User data, such as the user's name and room number, is fetched from the database using the user's ID. This data retrieval ensures that the form fields are pre-populated with relevant information. When the form is submitted, the script retrieves the submitted data and performs basic validation on it. For example, it ensures that required fields like start date, return date, place, and purpose are not empty. Additionally, it formats the parents' phone number to ensure it starts with '+91' and contains only numeric characters. This helps prevent incomplete or invalid data from being processed. After successfully inserting the leave form data into the database, the script integrates with the Sinch SMS service to send a notification message to the provided parents' phone number. This notification includes information about the user's leave request. Finally,

the script fetches the user's leave form history from the database and displays it in a table. This allows the user to view their past leave requests, providing transparency and accountability.

4.2.2.11 Query/Feedback:

Users can submit queries or provide feedback regarding hostel services, fostering effective communication and continuous improvement. This module enables users to voice their concerns, suggestions, or compliments, empowering them to contribute to the enhancement of hostel operations. The script starts by checking if the user is authenticated by verifying the presence of the variable. This ensures that only authenticated users can access the feedback page. User data, such as the user's name, is fetched from the database using the user's ID. This data retrieval ensures that the user's name is displayed on the page. If the query text passes validation, the script inserts the query details into the user_query table in the database. This includes the user's ID, query text, and an initial admin reply. If the insertion is successful, the query is submitted and stored in the database. After submitting the query, the script retrieves the user's previous queries from the database and displays them in a table. This allows the user to view their past queries and any admin replies.

5. TESTING

Testing is a set of various examinations with the main goal of thoroughly taxing the computer system. Even though every test has a distinct objective, all of them should confirm that every system component has been correctly integrated and is carrying out its assigned duty. Testing is the process of determining if the developed system satisfies the real requirements and system objectives. The goal of testing is to identify mistakes. A test that has a high chance of uncovering an inaccuracy that has gone unnoticed is good. A test that finds the hidden fault is considered successful. For this reason, test cases are created. A test case is a collection of data that will be input into the system.

TYPES OF TESTING

5.1 SYSTEM TESTING

When a system has been confirmed, it must be extensively tested to make sure that each component is working as intended and meeting all requirements, even in cases when the incorrect functions are requested or the incorrect make sure that every system component is functioning as it should and meeting the criteria, even in cases when incorrect functions are requested or incorrect data is presented.

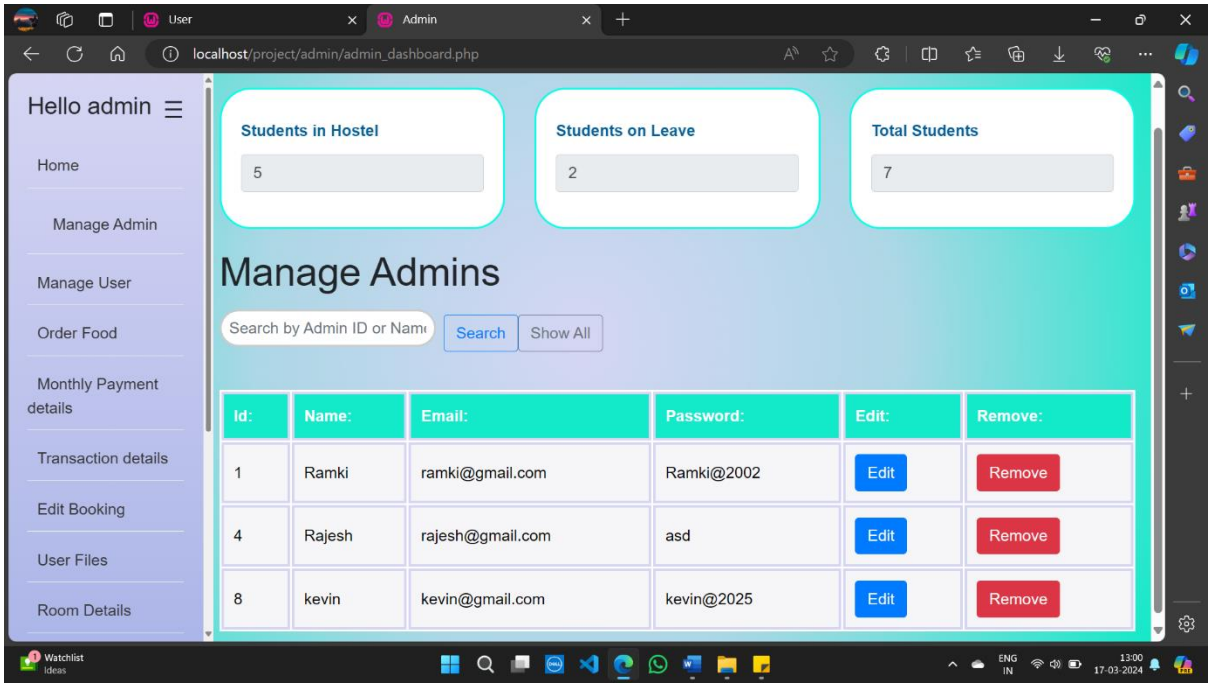


Figure 5.1 System Testing

5.2 UNIT TESTING

The unit test is the initial test conducted during the development process. Typically, the source code is separated into modules, which are further subdivided into units, which are smaller units. These units behave in a particular way. Unit testing is the process of testing these code units. The language used to design the project determines which unit test is used. Unit tests make assurance that every project route operates precisely according to the stated specifications and has inputs and outputs that are well defined. Testing for functionality and dependability in an engineering setting. Creating tests to verify that a product behaves correctly before system integration, starting with tests to verify how a product's components (nodes and vertices) behave.

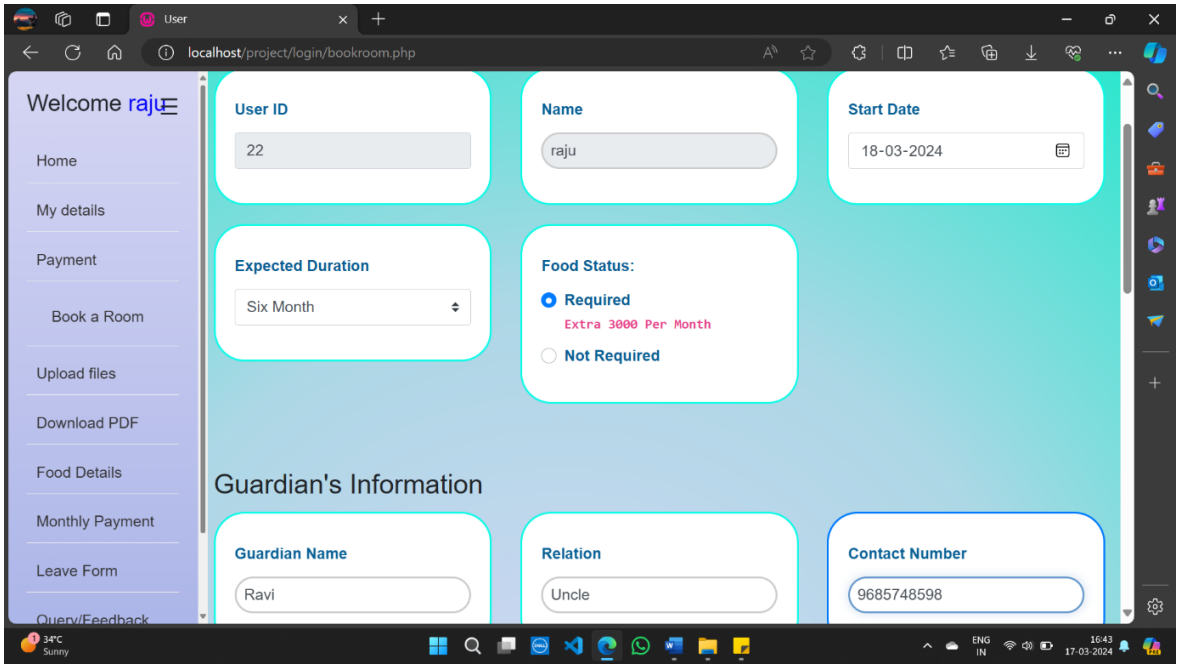


Figure 5.2.1 Unit Testing

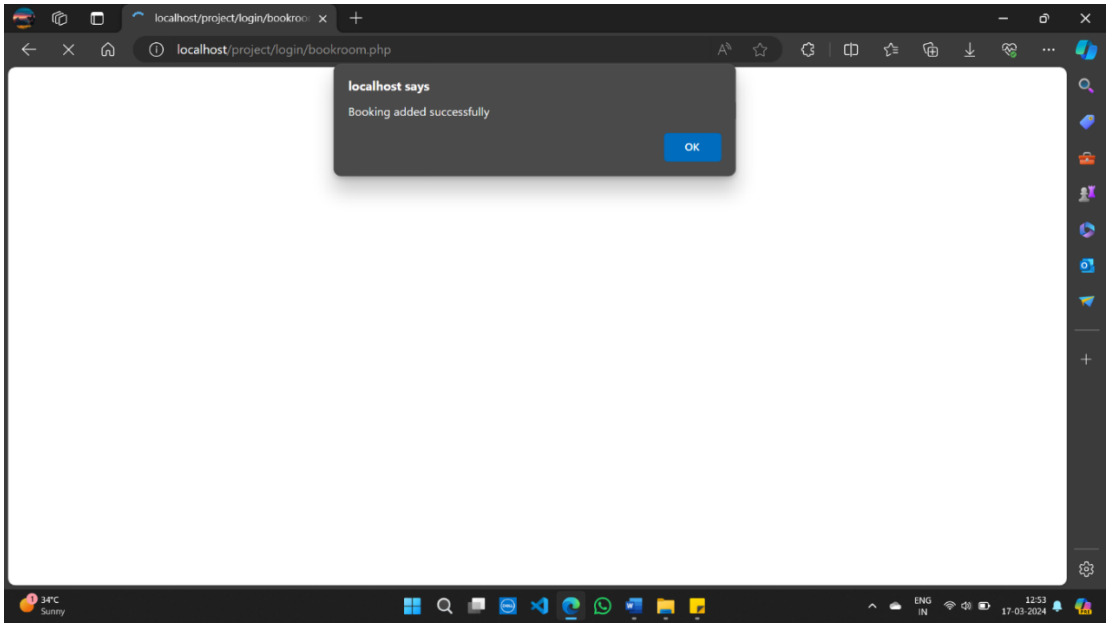


Figure 5.2.2 Unit Testing

5.3 INTEGRATION TESTING

Integration testing involves combining modules and testing them as a group, where modules could be code modules, individual applications, or components within a network. This testing phase occurs after unit testing and before system testing, aiming to validate the interactions between integrated components. Integration testing takes place once the product is code complete, often followed by beta testing, where beta versions may be distributed widely, including to the public, with the aim of gauging interest and generating sales for the final product release.

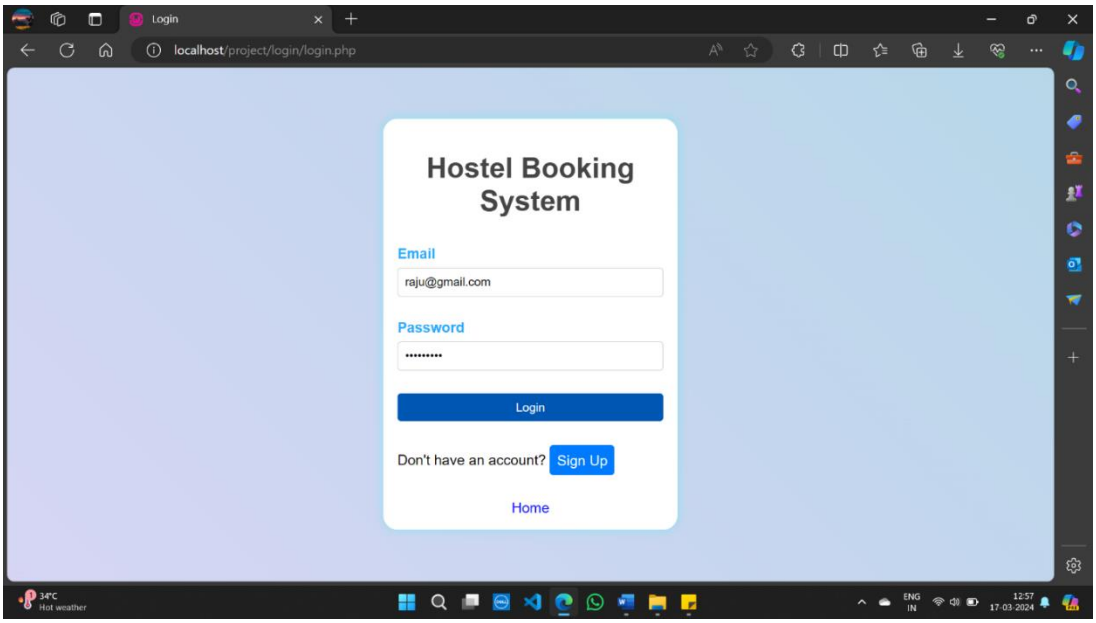


Figure 5.3.1 Integration Testing

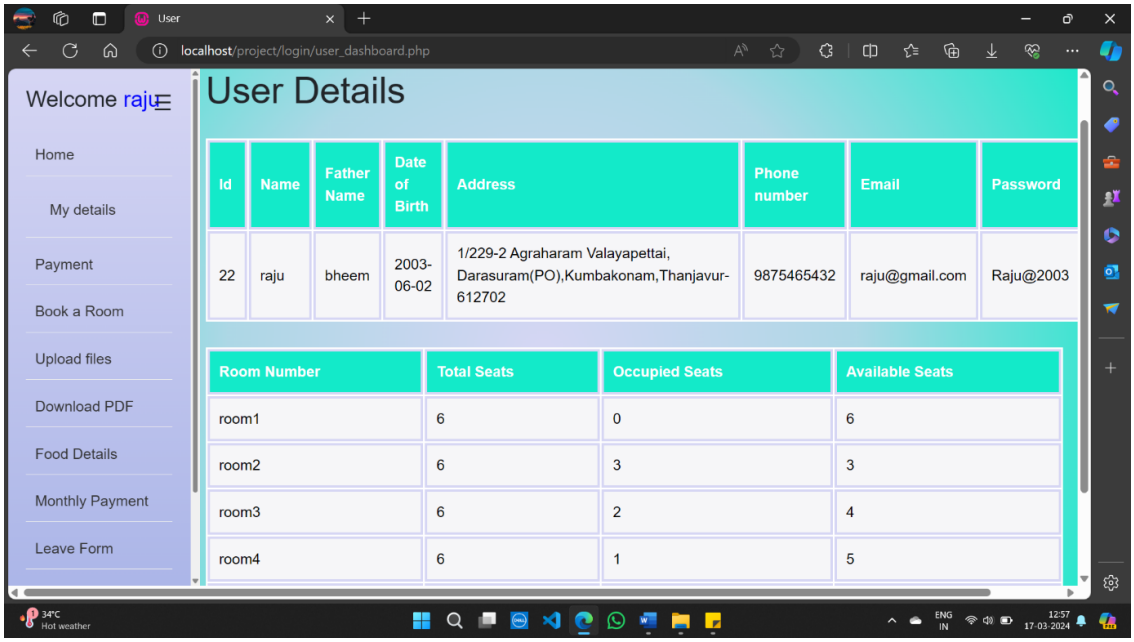


Figure 5.3.2 Integration Testing

5.4 VALIDATION TESTING

Valid and invalid data should be generated, and the program must process this data to detect errors. When users attempt to access a page through the login page using their user ID and password, the system verifies their credentials. If the user provides an incorrect format of password, a message is displayed, indicating, "Please enter a strong password." In this process, user inputs are validated to ensure accuracy and security.

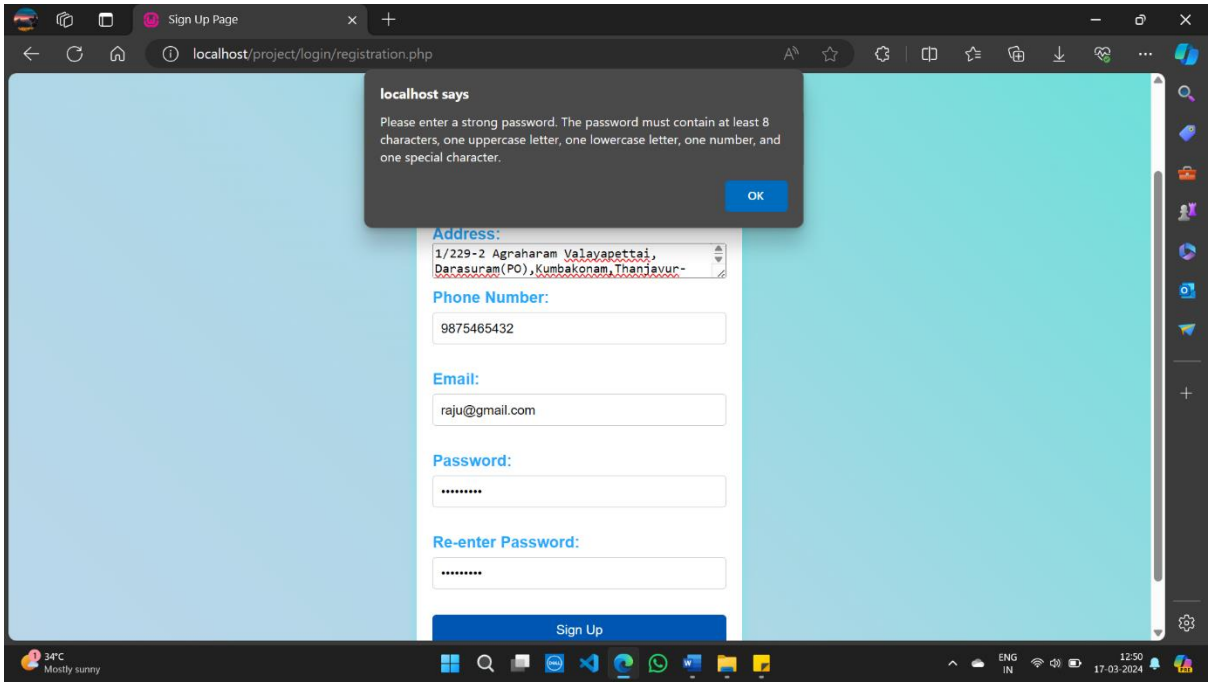


Figure 5.4 Validation Testing

6. CONCLUSION

The project entitled as **"Hostel Booking System"** has been developed to satisfy all the proposed requirements. This project marks a significant leap forward in hostel operations management. By replacing manual methods with an intuitive computerized solution, it streamlines tasks for administrators and users alike. With its user-friendly interface and comprehensive features, it simplifies processes such as registration, booking, and payment. This system not only boosts efficiency by automating tasks but also ensures transparency through detailed reporting. For users, it offers convenience and accessibility, making their experience hassle-free. Overall, the Hostel Management System sets a new standard in hostel management, promising a smoother journey for all involved parties. The system reduces the possibility of errors to a great extent and maintains the data in an efficient manner. User friendliness is the unique feature of this system. The system generates the reports as and when required. The system is highly interactive and flexible for further enhancement. The coding is done in a simplified and easy to understandable manner so that other team trying to enhance the project can do so without facing much difficulty. The documentation will also assist in the process as it has also been carried out in a simplified and concise way.

7. REFERENCES

BOOK REFERENCES

1. Adrian W. West, “Practical PHP and MySQL Website Databases”, 1st Edition, Apress, 2016.
2. Alan Forbes, “The Joy of PHP”, 6th edition, BeakCheck LLC, 2012.
3. Antonio Lopez, “Learning PHP 7”, PACKT Open Source Publication, 2016.
4. Dennis Popel, “Learning PHP Data Objects”, Packt Publishing, 2009.
5. Felke-Morris, “Web Development & Design Foundations with HTML5”, 10th Edition, Addison-Wesley, 2020.
6. Jon Duckett, “HTML and CSS: Design and Build Websites”,1st Edition, Wiley, 2011.
7. Lynn Beighley, “Headfirst PHP & MySQL”, 1st Edition, O’Reilly Media, 2007.
8. Paul DuBois, “MySQL”, 5th Edition, Addison-Wesley Professional, 2013.
9. Rasmus Lerdorf, “Programming PHP”, 4th Edition, O’Reilly Media, 2013.
10. Robin Nixon, “Learning PHP, MySQL, JavaScript, CSS & HTML5“, 4th Edition, O’Reilly Media, 2014.

WEBSITES REFERENCES

https://www.javatpoint.com/mysql-case-expression

https://www.javatpoint.com/php-download-file

https://www.javatpoint.com/php-file-upload

https://www.tutorialspoint.com/php/php_tutorial.pdf

https://www.phptutorial.net/php-tutorial/php-upload-multiple-files/

https://www.phptutorial.net/php-tutorial/php-time/

https://www.phptutorial.net/php-tutorial/php-date/

https://www.w3schools.com/php/php_sessions.asp

8. APPENDIX

8.1 Source code:

index.html:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Hostel Booking System</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <link rel="stylesheet" href="./Styles/index.css">
  </head>
<body>
  <div class="container">
    <h2>Hostel Booking System</h2>
    <ul class="nav nav-pills">
      <li class="nav-item"><a class="nav-link active" href="login.html">
Home</a></li>
      <li class="nav-item"><a class="nav-link" href="photogallery.html">Image
Gallery</a></li>
      <li class="nav-item"><a class="nav-link" href="contact.html">Contact </a></li>
      <li class="nav-item"><a class="nav-link" href="about.html">About </a></li>
      <li class="nav-item"><a class="nav-link" href="./login/login.php">User Login /
Register</a></li>
      <li class="nav-item"><a class="nav-link" href="./admin/adminlogin.php">Admin
Login / Register</a></li>
    </ul>
  </div>
  <div class="jumbotron">
    <h1>Welcome to Hostel Booking System</h1>
    <p>Explore our website to learn more about our hostel and its facilities. Book your
stay today!</p>
```

```
    <a href="./login/login.php" class="btn btn-primary btn-lg">Book Now</a>
  </div>
  <div class="container">
    <h2>Details and Images of Hostel</h2>
    <div class="row">
      <div class="col-md-4">
        <div class="card">
          <img src="./images/Room1.png" alt="Image1" style="width:100%">
          <div class="container">
            <h4><b>Room Details</b></h4>
            <p>Room size, Room capacity, Amenities</p></div></div></div>
      <div class="col-md-4">
        <div class="card">
          <img src="./images/commonarea.png" alt="Image2" style="width:300px;
height:225px">
          <div class="container">
            <h4><b>Common Area</b></h4>
            <p>Lounge, Dining Hall</p></div></div></div>
      <div class="col-md-4">
        <div class="card">
          <img src="./images/clean.png" alt="Image3" style="width:300px;
height:225px">
          <div class="container">
            <h4><b>Clean and Secure</b></h4>
            <p>Our staff is here to assist you 24/7</p>
          </div></div></div></div></div>
  <footer>
    <a href="index.html">Home</a> |
    <a href="photogallery.html">Image Gallery</a> |
    <a href="contact.html">Contact</a> |
    <a href="about.html">About</a> |
    <br>
    &copy; 2024 Hostel Booking System. All rights reserved.
   </footer></body></html>
```

login.php:

```
<?php
require 'config.php';
if(isset($_POST["submit"])){
   $email=$_POST["email"];
   $password=$_POST["password"];
   $result=mysqli_query($conn,"SELECT * FROM users WHERE email='$email'");
   $row=mysqli_fetch_assoc($result);
   if(mysqli_num_rows($result)>0){
      if($password == $row["password"]){
         session_start();
         $_SESSION["login"]=true;
         $_SESSION["id"]=$row["id"];
         header("Location: user_dashboard.php");}
      else{echo"<script> alert('Wrong Password'); </script>"; }}
   else{echo"<script> alert('User not registered'); </script>";}}?>
```

user_dashboard.php:

```
<?php
require 'config.php';
if (!isset($_SESSION["login"])) {
   header("Location: login.php");
   exit();}
$Id = $_SESSION["id"];
$userNameQuery = "SELECT name FROM users WHERE id = '$Id'";
$userNameResult = mysqli_query($conn, $userNameQuery);
if ($userNameResult) {
   $userNameRow = mysqli_fetch_assoc($userNameResult);
   $userName = $userNameRow['name'];
}else {
$userName = 'Guest';}
$result = mysqli_query($conn, "SELECT * FROM users WHERE id='$Id'");
$admin = mysqli_fetch_assoc($result);
```

```
$sql="Select * from users WHERE id = '$Id'";
$result=$conn->query($sql);
$query = "SELECT * FROM room";
$roomresult = $conn->query($query);?>

<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>User</title>
   <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
   <link rel="stylesheet" href="../Styles/user_dashboard.css"></head>
<body>
<div class="sidebar" id="sidebar" style="display: block; width:18%; height: 100%;
overflow-y: auto; scrollbar-width: thin; scrollbar-color: #888 #f0f0f0; -webkit-scrollbar-
width: thin; -webkit-scrollbar-color: #888 #f0f0f0;">
<h2>Welcome <span style="color:blue;"><?php echo $userName; ?> </span></h2>
      <br>
      <ul>
         <li><a href="home.html"><i class="fas fa-home"></i> Home</a></li>
         <li><a class="nav-link active" href="user_dashboard.php"><i class="fas fa-
home"></i>My details</a></li>
         <li><a href="payment.php"><i class="fas fa-cog"></i> Payment</a></li>
         <li><a href="bookroom.php"><i class="fas fa-envelope"></i> Book a
Room</a></li>
         <li><a href="uploadfiles.php"><i class="fas fa-cog"></i>Upload files</a></li>
         <li><a href="report.php"><i class="fas fa-cog"></i> Download PDF</a></li>
         <li><a href="foodorder.php"><i class="fas fa-cog"></i> Food Details</a></li>
         <li><a href="monthlypayment.php"><i class="fas fa-cog"></i>Monthly
Payment</a></li>
         <li><a href="leaveform.php"><i class="fas fa-cog"></i> Leave Form</a></li>
         <li><a href="feedback.php"><i class="fas fa-cog"></i> Query/Feedback</a></li>
```

```
        <li><a href="logout.php"><i class="fas fa-sign-out-alt"></i> Logout</a></li>
      </ul>
      <div class="sidebar-toggle" onclick="toggleSidebar()">&#9776;</div></div>
   <div id="helloAdminSection">
   <h5 style="display: inline-block; margin-right: 10px;">Hello <?php echo $userName; ?></h5>
   <div class="show-sidebar-btn" onclick="toggleSidebar()"  style="display: inline-block;font-size: 24px; color: blue; cursor: pointer;">&#9776;</div></div>
   <div class="main-content" id="mainContent">
      <h1>User Details</h1>
      <table><tr>
      <th>Id</th><th>Name</th><th>Father Name</th><th>Date of Birth</th>
      <th>Address</th><th>Phone number</th><th>Email</th><th>Password</th>
<th>Option</th></tr>
   <?php
   if ($result->num_rows > 0) {
      $count = 0;
      while ($row = $result->fetch_assoc()) {
         if ($count % 2 == 0) {
            echo "<tr>";}
         echo "<td>" . $row["id"] . "</td>";echo "<td>" . $row["name"] . "</td>";
         echo "<td>" . $row["father_name"] . "</td>";echo "<td>" . $row["dob"] . "</td>";
         echo "<td>" . $row["address"] . "</td>";echo "<td>" . $row["phone_number"] . "</td>";echo "<td>" . $row["email"] . "</td>";echo "<td>" . $row["password"] . "</td>";
         echo "<td><a href='edit-user.php?id={$row['id']}'class='btn btn-primary'>Edit</a></td>";
         if ($count % 2 == 1) {
            echo "</tr>"; }
         $count++;}
      if ($count % 2 == 1) {
         echo "</tr>";
      } } else {
      echo "<tr><td colspan='9'>No results found.</td></tr>";}
   echo "<table>
```

```
        <tr><th>Room Number</th><th>Total Seats</th>
            <th>Occupied Seats</th><th>Available Seats</th> </tr>";
    while ($row = $roomresult->fetch_assoc()) {
    echo "<tr><td>{$row['room_number']}</td><td>{$row['total_seats']}</td>
            <td>{$row['occupied_seats']}</td><td>{$row['available_seats']}</td></tr>";}
echo "</table>";?></table></div>
<script>
        function toggleSidebar() {
        var sidebar = document.getElementById("sidebar");
        var mainContent = document.getElementById("mainContent");
        if (sidebar.style.display === "none" || sidebar.style.display === "") {
            sidebar.style.display = "block";
            mainContent.style.marginLeft = "250px"; // Adjust the width as needed
        } else {
            sidebar.style.display = "none";
            mainContent.style.marginLeft = "0";}}
    </script>
</body></html>
```

payment.php:

```
<?php
require "config.php";
if (!isset($_SESSION['login'])) {
    header("Location: login.php");
    exit();}
$user_id = $_SESSION['id'];
$userNameQuery = "SELECT name FROM users WHERE id = '$user_id'";
$userNameResult = mysqli_query($conn, $userNameQuery);
if ($userNameResult) {
    $userNameRow = mysqli_fetch_assoc($userNameResult);
    $userName = $userNameRow['name'];
}else {
       $userName = 'Guest';}
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
```

```
    $transaction_id = $_POST['transaction_id'];
    if (!$conn) {
        die("Connection failed: " . mysqli_connect_error());
    }
    $checkUserQuery = "SELECT * FROM users WHERE id = '$user_id'";
    $result = mysqli_query($conn, $checkUserQuery);
    if (mysqli_num_rows($result) > 0) {
    $query = "INSERT INTO payment (user_id, transaction_id) VALUES ('$user_id',
'$transaction_id')";
    if (mysqli_query($conn, $query)) {echo "<script>alert('Payment successfully
recorded.')</script>";
    } else {echo "<script>alert('Payment already recorded.')</script>";}}
    else {echo "<script>alert('User not found.')</script>";}mysqli_close($conn);}?>
<!DOCTYPE html>
<html lang="en">
<head>
    </head>
<body>
        <div class="sidebar-toggle" onclick="toggleSidebar()">&#9776;</div>
    </div>
    <div class="main-content" id="mainContent">
        <div class="container">
            <h3>Payment QR Code</h3>
            <h6> Pay ₹500 to confirm your room (Only Once)</h6>
            <img src="../images/payment.jpg" alt="Payment QR Code">
            <form action="payment.php" method="post">
                <label for="user_id">User ID:</label>
                <input type="number" name="user_id" id="user_id" required class="form-control"
value="<?php echo $user_id; ?>" readonly>
                <label for="transaction_id">Transaction ID:</label>
                <input type="text" id="transaction_id" name="transaction_id"
placeholder="Careful" required>
                <button type="submit">Submit Payment</button></form></div></div>
</body></html>
```

bookroom.php:

```
<?php
require 'config.php';
if (!isset($_SESSION['login'])) {
   header("Location: login.php");
   exit();}
$user_id = $_SESSION['id'];
$sql = "SELECT * FROM room";
$result = mysqli_query($conn, $sql);
$userNameQuery = "SELECT name FROM users WHERE id = '$user_id'";
$userNameResult = mysqli_query($conn, $userNameQuery);
if ($userNameResult) {
   $userNameRow = mysqli_fetch_assoc($userNameResult);
   $userName = $userNameRow['name'];
}else {
     $userName = 'Guest'; }
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $start_date = $_POST['startdate']; duration = $_POST['duration'];
   $food_status = $_POST['foodstatus']; $guardian_name = $_POST['gname'];
   $relation = $_POST['grelation']; $guardian_contact = $_POST['gcontact'];
   $emergency_contact = $_POST['econtact']; $selected_room = $_POST['room'];
   if ($conn->connect_error) {
      die("Connection failed: " . $conn->connect_error);}
   $check_user_sql = "SELECT * FROM users WHERE id = '$user_id'";
   $result = $conn->query($check_user_sql);
   if ($result->num_rows > 0) {
      $check_transaction_query = "SELECT * FROM payment WHERE user_id =
'$user_id'";
      $transaction_result = $conn->query($check_transaction_query);
      if ($transaction_result->num_rows > 0) {
         $base_fees = ($food_status == 1) ? 6000 : 3000;
         $total_fees = $base_fees * $duration;
         $check_booking_query = "SELECT * FROM bookings WHERE user_id =
'$user_id'";
```

```
        $booking_result = $conn->query($check_booking_query);
        if ($booking_result->num_rows == 0) {
            $sql = "INSERT INTO bookings (user_id, start_date, duration, food_status,
guardian_name, relation, guardian_contact, emergency_contact, total_fees, selected_room)
                    VALUES ('$user_id', '$start_date', '$duration', '$food_status',
'$guardian_name', '$relation', '$guardian_contact', '$emergency_contact', '$total_fees',
'$selected_room')";
            if ($conn->query($sql) === TRUE) {
                $update_room_query = "UPDATE room SET occupied_seats = occupied_seats +
1, available_seats = available_seats - 1 WHERE room_number = '$selected_room' AND
available_seats > 0";
                if ($conn->query($update_room_query) === TRUE) {
                    echo "<script>alert('Booking added successfully');</script>";}
                else {echo "<script>alert('Error updating room: " . $conn->error . "');</script>";}
            } else {echo "<script>alert('Error: " . $sql . "<br>" . $conn->error . "');</script>";}
        } else {echo "<script>alert('You have already made a booking. Each user can only
book once.');</script>";}}
    else {echo "<script>alert('User has no transaction. Please make a transaction
first.');</script>";}
    }else {echo "<script>alert('Invalid User ID. Please enter a valid User ID.');</script>";}
    $check_available_seats_query = "SELECT available_seats FROM room WHERE
room_number = '$selected_room'";
    $available_seats_result = $conn->query($check_available_seats_query);
    if ($available_seats_result && $available_seats_row = $available_seats_result-
>fetch_assoc()) {$available_seats = $available_seats_row['available_seats'];
        if ($available_seats == 0) {echo "<script>alert('No available seats in
$selected_room');</script>";}
    } else {echo "<script>alert('Error checking available seats: " . $conn->error .
"');</script>";}}
$conn->close();
?>
```

uploadfiles.php:

```
<?php
require 'config.php';
if (!isset($_SESSION['login'])) {
   header("Location: login.php");
   exit();}
$input_user_id= $_SESSION['id'];
$userNameQuery = "SELECT name FROM users WHERE id = '$input_user_id'";
$userNameResult = mysqli_query($conn, $userNameQuery);
if ($userNameResult) {
   $userNameRow = mysqli_fetch_assoc($userNameResult);
   $userName = $userNameRow['name'];
}else {
   // Handle the error if necessary
   $userName = 'Guest'; // Default value if the name is not found}
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   if (empty($_FILES['photo']['name']) || empty($_FILES['id_card']['name'])) {
      echo "<script>alert('Please select both photo and ID proof files.')</script>";
      echo "<script>window.location.href = 'uploadfiles.php';</script>";
      return; // Stop further processing and stay on the same page}
   if ($conn->connect_error) {
      die("Connection failed: " . $conn->connect_error);}
   $check_user_query = "SELECT * FROM users WHERE id = '$input_user_id'";
   $result = $conn->query($check_user_query);
   $check_existing_files_query = "SELECT * FROM user_files WHERE user_id =
'$input_user_id'";
   $result_existing_files = $conn->query($check_existing_files_query);
   if ($result_existing_files->num_rows > 0) {
      echo "<script>alert('Files have already been uploaded for this user.')</script>";
      $conn->close();
      exit();}
   if ($result->num_rows > 0) {
      if (isset($_FILES['photo']) && isset($_FILES['id_card'])) {
```

```
        $photo_filename = $_FILES['photo']['name'];
        $photo_tmp_path = $_FILES['photo']['tmp_name'];
        $id_card_filename = $_FILES['id_card']['name'];
        $id_card_tmp_path = $_FILES['id_card']['tmp_name'];
        $photo_unique_filename = generateUniqueFilename($photo_filename);
        $id_card_unique_filename = generateUniqueFilename($id_card_filename);
        $uploadDir = 'uploads/';
        $photo_destination = $uploadDir . $photo_unique_filename;
        $id_card_destination = $uploadDir . $id_card_unique_filename;
        move_uploaded_file($photo_tmp_path, $photo_destination);
        move_uploaded_file($id_card_tmp_path, $id_card_destination);
        $insert_files_query = "INSERT INTO user_files (user_id, photo_filename,
id_proof_filename) VALUES (?, ?, ?)";
        $stmt_insert_files = $conn->prepare($insert_files_query);
        $stmt_insert_files->bind_param("iss", $input_user_id, $photo_unique_filename,
$id_card_unique_filename);
        if ($stmt_insert_files->execute()) {
            echo "<script>alert('Files uploaded and stored successfully!')</script>";
        } else {echo "Error: " . $stmt_insert_files->error;}
        $stmt_insert_files->close();
      } else {echo "<script>alert('Please select both photo and ID proof files.')</script>";}
   } else {echo "<script>alert('Invalid User ID. Please enter a valid User ID.')</script>";}
   $conn->close(); }
function generateUniqueFilename($filename) {
   $timestamp = time();
   $extension = pathinfo($filename, PATHINFO_EXTENSION);
   return $timestamp . '_' . uniqid() . '.' . $extension;}?>

   <h3>Upload Files</h3>
   <form action="uploadfiles.php" method="POST" enctype="multipart/form-data">
   <h3 class="card-title mt-5"></h3><div class="row">
   <div class="col-sm-12 col-md-6 col-lg-4">
                        <div class="card">
                           <div class="card-body">
```

```
                        <h6 class="card-title">User ID</h6>
                            <div class="form-group">
                                <input type="number" name="userid" id="userid" required
class="form-control" value="<?php echo $input_user_id; ?>" readonly>
                        </div></div></div></div></div>
    <table>
    <th><label for="photo">Photo:</label></th>
    <td><input type="file" name="photo" accept=".jpg,.png" onchange="previewImage(event,
'photo')"></td>
    <th><label for="id_proof">ID Proof:</label> </th>
    <td><input type="file" name="id_card" accept=".pdf" onchange="previewPdf(event,
'id_card')"></td> </table>
    <div id="preview-container"><div id="preview-photo">
        <img id="photo-preview" src="" alt="Passport size photo preview"></div>
      <div id="preview-id-card">
        <embed id="id-card-preview" type="application/pdf" width="100%"
height="600px"></div></div>
    <button type="submit" name="submit">Upload Files</button></form>
    <script>
      function previewImage(event, id) {
        var previewContainer = document.getElementById("preview-container");
        previewContainer.style.display = "block";
        var previewImage = document.getElementById("photo-preview");
        var file = event.target.files[0];
        var reader = new FileReader();
        if (file.type.startsWith("image/")) {
          if (file.size >= 50000 && file.size <= 150000) {
            reader.onload = function (event) {
              previewImage.src = event.target.result;};
            reader.readAsDataURL(file);
          } else {alert("Image file size should be between 50KB and 150KB.");
            previewContainer.style.display = "none";}
        } else {alert("Invalid file type. Please select an image file for the passport size
photo.");
```

```
            previewContainer.style.display = "none";}}
        function previewPdf(event, id) {
            var previewContainer = document.getElementById("preview-container");
            previewContainer.style.display = "block";
            var previewPdf = document.getElementById("id-card-preview");
            var file = event.target.files[0];
            var reader = new FileReader();
            if (file.type === "application/pdf") {
                if (file.size >= 50000 && file.size <= 2000000) {
                    reader.onload = function (event) {
                        previewPdf.src = event.target.result;};
                    reader.readAsDataURL(file);
                } else {
                    alert("PDF file size should be between 50KB and 2MB.");
                    previewContainer.style.display = "none";}
            } else {
                alert("Invalid file type. Please select a PDF file for the college ID card.");
                previewContainer.style.display = "none";}}    </script></body></html>
```

foodorder.php:

```
<?php
require "config.php";
if (!isset($_SESSION['login'])) {
    header("Location: login.php");
    exit();}
$user_id = $_SESSION['id'];
$userNameQuery = "SELECT name FROM users WHERE id = '$user_id'";
$userNameResult = mysqli_query($conn, $userNameQuery);
if ($userNameResult) {
    $userNameRow = mysqli_fetch_assoc($userNameResult);
    $userName = $userNameRow['name'];
} else {
    $userName = 'Guest'; }
if ($conn->connect_error) {
```

```
   die("Connection failed: " . $conn->connect_error);}
$foods_query = "SELECT * FROM foods";
$foods_result = $conn->query($foods_query);
if ($foods_result === FALSE) {echo "Error in query for fetching foods: " . $conn->error;}
$user_orders_query = "SELECT * FROM food_order WHERE user_id = '$user_id'";
$user_orders_result = $conn->query($user_orders_query);
if ($user_orders_result === FALSE) {
   echo "Error in query for fetching user orders: " . $conn->error;}
$orderPlaced = false;
if ($_SERVER["REQUEST_METHOD"] == "POST") {
      foreach ($_POST as $key => $value) {
          if (strpos($key, 'food_item_') === 0) {
          $rowId = substr($key, strlen('food_item_'));
          $food_item_id = $_POST["food_item_$rowId"];
          $foodcount = $_POST["foodcount_$rowId"];
          $meal_type = $_POST["meal_type_$rowId"];
          if ($foodcount > 0) {
             $food_details_query = "SELECT foodname, price, photo_name FROM foods
WHERE id = $food_item_id";
             $food_details_result = $conn->query($food_details_query);
             if ($food_details_result === FALSE) {
                echo "Error in query for fetching food details: " . $conn->error;}
             $food_details_row = $food_details_result->fetch_assoc();
             $foodname = $food_details_row['foodname'];
             $price = $food_details_row['price'];
             $total_price = $foodcount * $price;
             $order_query = "INSERT INTO food_order (user_id, foodname, foodcount, price,
bf_lun_din) VALUES ('$user_id', '$foodname', '$foodcount', '$total_price', '$meal_type')";
             if ($conn->query($order_query) === TRUE) {
                $orderPlaced = true;}
                   else {echo "Error placing order: " . $conn->error;}
          } else {//echo "Food count must be greater than 0.";}}}}
          if ($orderPlaced) {
```

```
    echo "<script>alert('Order placed successfully!'); window.location.href =
'foodorder.php';</script>";}?>
    <h1>Food Details:</h1>
    <img id="topImage" src="../images/foodmenu.jpg" alt="Top Image">
    <form method="post" action="foodorder.php">
    <table class="table table-bordered"><thead><tr>
        <th>Food Name</th><th>Image</th><th>Price</th>
        <th>Food Count</th><th>Meal Type</th><th>Total Price</th></tr></thead><tbody>
    <?php
    $rowId = 1;
    while ($food_row = $foods_result->fetch_assoc()) {
        echo "<tr>";
        echo "<td>{$food_row['foodname']}</td>";
        echo "<td><img src='../admin/uploads/{$food_row['photo_name']}' alt='Food Image'
style='width: 100px; height:100px;'></td>";
        echo "<td>{$food_row['price']}</td>";
        echo "<td><input type='number' id='foodcount_$rowId' name='foodcount_$rowId'
value='0' min='0' onchange='updateTotalPrice($rowId)' required></td>";echo "<td>";
        echo "<select name='meal_type_$rowId'>";
        echo "<option value='breakfast'>Breakfast</option>";
        echo "<option value='lunch'>Lunch</option>";
        echo "<option value='dinner'>Dinner</option>";echo "</select>";echo "</td>";
        echo "<td><input type='text' id='total_price_$rowId' name='total_price_$rowId'
readonly></td>";echo "<td>";
        echo "<input type='hidden' name='food_item_$rowId' value='{$food_row['id']}'>";
        echo "<input type='hidden' id='price_$rowId' value='{$food_row['price']}'>";
        echo "</td>";echo "</tr>";$rowId++;}?></tbody></table>
    <center><button type="submit" class="btn btn-primary">Place Order</button></center>
</form>
<div>
        <h2>User's Food Orders:</h2>
        <table class="table table-bordered">
            <thead>
                <tr><th>Order ID</th><th>Food Name</th>
```

```
            <th>Food Count</th><th>Total Price</th>
            <th>Period</th><th>Ordered DateTime</th>
        </tr></thead><tbody>
        <?php
        while ($order_row = $user_orders_result->fetch_assoc()) {
            echo "<tr>";
            echo "<td>{$order_row['id']}</td>";
            echo "<td>{$order_row['foodname']}</td>";
            echo "<td>{$order_row['foodcount']}</td>";
            echo "<td>{$order_row['price']}</td>";
            echo "<td>{$order_row['bf_lun_din']}</td>";
            echo "<td>{$order_row['created_at']}</td>";echo "</tr>";}?>
    </tbody>       </table></div></div>
<script>
    function updateTotalPrice(rowId) {
        var quantityInput = document.getElementById("foodcount_" + rowId);
        var priceInput = document.getElementById("price_" + rowId);
        var totalInput = document.getElementById("total_price_" + rowId);
        var total_price = quantityInput.value * priceInput.value;
        totalInput.value = total_price;
    }</body></html>
```

report.php:

```
<?php

        require "config.php";
        if (!isset($_SESSION['login'])) {
            header("Location: login.php");
            exit();}
        $input_user_id = $_SESSION['id'];
        $userNameQuery = "SELECT name FROM users WHERE id = '$input_user_id'";
        $userNameResult = mysqli_query($conn, $userNameQuery);
        if ($userNameResult) {$userNameRow = mysqli_fetch_assoc($userNameResult);
            $userName = $userNameRow['name'];
```

```
}else {$userName = 'Guest'; }
if ($conn->connect_error) {die("Connection failed: " . $conn->connect_error);}
if (isset($_POST['user_id']) && isset($_POST['password'])) {
   $user_id = $_POST['user_id']; $password = $_POST['password'];
   $auth_query = "SELECT * FROM users WHERE id = '$user_id' AND password =
'$password'";$auth_result = $conn->query($auth_query);
   if ($auth_result === false) {
      die("Authentication query error: " . $conn->error);
   } elseif ($auth_result->num_rows > 0) {
      // User authenticated, proceed with displaying the table}?>
<div id ="invoice"><div class="table-responsive">
            <table class="table"><tbody><?php
                  $query = "SELECT u.name, u.father_name, u.dob, u.phone_number,
b.start_date, b.duration, b.food_status, b.guardian_name, b.relation,
b.guardian_contact, b.emergency_contact, b.total_fees, b.selected_room,
p.transaction_id,p.payment_date,uf.photo_filename, b.created_at FROM users u
JOIN bookings b ON u.id = b.user_id JOIN user_files uf ON u.id = uf.user_id
JOIN payment p ON u.id = p.user_id WHERE u.id = '$user_id'";
                  $result = $conn->query($query);
                  if ($result === false) {
                     die("Query error: " . $conn->error);}
                  while ($row = $result->fetch_assoc()) {
                     echo "<tr>";echo "<th>Created At:</th>";
                     echo "<td>{$row['created_at']}</td>";echo "</tr>";echo "<tr>";
                     echo "<th>User Photo:</th>";
                     echo "<td><img src='uploads/{$row['photo_filename']}' alt='User
Photo'></td>";
                     echo "</tr>";echo "<tr>";
                     echo "<th>Name:</th>"; echo "<td>{$row['name']}</td>";
                     echo "</tr>";echo "<tr>";
                     echo "<th>Father Name:</th>";
                     echo "<td>{$row['father_name']}</td>";echo "</tr>";echo "<tr>";
                     echo "<th>DOB:</th>";
                     echo "<td>{$row['dob']}</td>";echo "</tr>";echo "<tr>";
```

```
                        echo "<th>Phone Number:</th>";
                        echo "<td>{$row['phone_number']}</td>";echo "</tr>";
                        echo "<tr>";echo "<th>Start Date:</th>";
                        echo "<td>{$row['start_date']}</td>";echo "</tr>";
                        echo "<tr>";echo "<th>Duration (in months):</th>";
                        echo "<td>{$row['duration']}</td>";echo "</tr>";
                        echo "<tr>";echo "<th>Food Status:</th>";
                        echo "<td>{$row['food_status']}</td>";echo "</tr>";echo "<tr>";
                        echo "<th>Guardian Name:</th>";
                        echo "<td>{$row['guardian_name']}</td>";echo "</tr>";
                        echo "<tr>";echo "<th>Relation:</th>";
                        echo "<td>{$row['relation']}</td>";echo "</tr>";
                        echo "<tr>";echo "<th>Guardian Contact:</th>";
                        echo "<td>{$row['guardian_contact']}</td>";echo "</tr>";
                        echo "<tr>";echo "<th>Emergency Contact:</th>";
                        echo "<td>{$row['emergency_contact']}</td>";echo "</tr>";
                        echo "<tr>";echo "<th>Total Fees:</th>";
                        echo "<td>{$row['total_fees']}</td>";
                        echo "</tr>";echo "<tr>";echo "<th>Selected Room:</th>";
                        echo "<td>{$row['selected_room']}</td>";
                        echo "</tr>"; echo "<tr>";echo "<th>Transaction Id:</th>";
                        echo "<td>{$row['transaction_id']}</td>";
                        echo "</tr>";echo "<th>Transaction Date::</th>";
                        echo "<td>{$row['payment_date']}</td>";echo "</tr>";
                    }$result->close();?>
                </tbody></table></div></div>
        <center> <button onclick="downloadReportPDF()" class="btn btn-
primary">Download PDF</button></center></div>
</body><script>
    function downloadReportPDF(){
    const element= document.getElementById("invoice");
    html2pdf()
    .from(element)
    .save();}
```

```
</script></html>
<?php} else {echo "Invalid credentials. Please try again.";}}?>
```

leaveform.php:

```
<?php
require "config.php";
if (!isset($_SESSION['login'])) {
   header("Location: login.php");
   exit();}
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);}
$user_id = $_SESSION['id'];
$userQuery = "SELECT name FROM users WHERE id = '$user_id'";
$userResult = $conn->query($userQuery);
$userNameQuery = "SELECT name FROM users WHERE id = '$user_id'";
$userNameResult = mysqli_query($conn, $userNameQuery);
$userparentsnoQuery ="SELECT emergency_contact FROM bookings WHERE
user_id = '$user_id'";
$userparentsResult =mysqli_query($conn,$userparentsnoQuery);
if ($userNameResult) {$userNameRow = mysqli_fetch_assoc($userNameResult);
   $userName = $userNameRow['name'];
}else {$userName = 'Guest';}
if ($userparentsResult === false) {echo "Error: " . $userparentsnoQuery . "<br>" .
mysqli_error($conn);
} else {if (mysqli_num_rows($userparentsResult) > 0) {
      $row = mysqli_fetch_assoc($userparentsResult);
      $emergencyContactInfo = $row["emergency_contact"];
   } else {$emergencyContactInfo = "Not available";}}
if ($userResult->num_rows > 0) {$userData = $userResult->fetch_assoc();
   $userName = $userData['name'];
} else {$userName = ""; }
$bookingQuery = "SELECT selected_room FROM bookings WHERE user_id =
'$user_id'";
$bookingResult = $conn->query($bookingQuery);
```

```
if ($bookingResult->num_rows > 0) {
   $bookingData = $bookingResult->fetch_assoc();
   $roomNumber = $bookingData['selected_room'];
} else {
   $roomNumber = ""; }
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $room = $_POST['room'];
   $startDate = $_POST['startDate']; $returnDate = $_POST['returnDate'];
   $place = $_POST['place']; $purpose = $_POST['purpose'];
   $parentsNumber = $_POST['parentsNumber']; $parentsNumber =
$_POST['parentsNumber']; $parentsNumber = '+91' . preg_replace('/[^0-9]/', '',
$parentsNumber); $messageContent = "Your son Mr. $userName ($room) is leaving
the hostel at $startDate. He mentioned that he will return to the hostel at $returnDate.
He is going to $place because of $purpose. Thank you. By V-Boys Hostel.";
   $insertQuery = "INSERT INTO leave_form (user_id, name, room, start_date,
return_date, place, purpose, parents_number, message) VALUES ('$user_id',
'$userName', '$room', '$startDate', '$returnDate', '$place', '$purpose',
'$parentsNumber', '$messageContent')";
   if ($conn->query($insertQuery) === TRUE) {
      echo "<script>alert('Message sent successfully and data stored!')</script>";
      $service_plan_id = "aed8e25ec5a64d409c8bc1a142ea44a";
      $bearer_token = "b1af6ddbc970481d8612e9b821fc140";
      $send_from = "+44752066244#";
         if (stristr($parentsNumber, ',')) {
         $recipient_phone_numbers = explode(',', $parentsNumber);
      } else {$recipient_phone_numbers = [$parentsNumber];}
      $content = [
         'to' => array_values($recipient_phone_numbers),
         'from' => $send_from,
         'body' => $messageContent];
      $data = json_encode($content);
      $ch =
curl_init("https://us.sms.api.sinch.com/xms/v1/{$service_plan_id}/batches");
```

```
        curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type:
application/json'));
        curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BEARER);
        curl_setopt($ch, CURLOPT_XOAUTH2_BEARER, $bearer_token);
        curl_setopt($ch, CURLOPT_POST, true);
        curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
        $result = curl_exec($ch);
        if (curl_errno($ch)) {
            //echo 'Curl error: ' . curl_error($ch);
        } else {
            //echo $result;}
        curl_close($ch);
    } else {
        echo "Error: " . $insertQuery . "<br>" . $conn->error;}}?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>User</title>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <link rel="stylesheet" href="../Styles/user_dashboard.css">
    <link rel="stylesheet" href="../Styles/innerlogin.css">
</head>
<body>
        <div class="sidebar-toggle" onclick="toggleSidebar()">&#9776;</div></div>
    <div id="helloAdminSection">
    <h5 style="display: inline-block; margin-right: 10px;">Hello <?php echo
$userName; ?></h5>
    <div class="show-sidebar-btn" onclick="toggleSidebar()"  style="display: inline-
block;font-size: 24px; color: blue; cursor: pointer;">&#9776;</div>
```

```
    </div>
    <div class="main-content" id="mainContent">
        <h1>Leave Form:</h1>
        <div class="container">
    <form id="travelForm" action="leaveform.php" method="post">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name" value="<?php echo $userName;
?>" readonly><br>
        <label for="room">Room:</label>
        <input type="text" id="room" name="room" value="<?php echo $roomNumber;
?>" readonly><br>
        <label for="startDate">From Date and Time:</label>
        <input type="datetime-local" id="startDate" name="startDate" required><br>
        <label for="returnDate">Return Date and Time:</label>
        <input type="datetime-local" id="returnDate" name="returnDate" required><br>
        <label for="place">Place:</label>
        <input type="text" id="place" name="place" required><br>
        <label for="purpose">Purpose:</label>
        <input type="text" id="purpose" name="purpose" required><br>
        <label for="parentsNumber">Parents' Number:</label>
        <input type="tel" id="parentsNumber" name="parentsNumber" value="<?php
echo $emergencyContactInfo; ?>" readonly><br>
        <button type="submit">Submit</button>
</form>
</div>
<h1>Your Leave Form History:</h1>
<table border="1"><tr>
        <th>ID</th><th>Name</th><th>Room</th>
        <th>Start Date</th><th>Return Date</th><th>Place</th>
        <th>Purpose</th><th>Parents' Number</th><th>Applied Date</th>
        <th>No. of days</th><th>Action</th></tr>
    <?php
    $leaveFormQuery = "SELECT * FROM leave_form WHERE user_id = '$user_id'
ORDER BY created_at DESC";
```

```
    $leaveFormResult = $conn->query($leaveFormQuery);
    if ($leaveFormResult->num_rows > 0)
        while ($row = $leaveFormResult->fetch_assoc()) {
            echo "<tr>";
            echo "<td>" . $row["id"] . "</td>";echo "<td>" . $row["name"] . "</td>";
            echo "<td>" . $row["room"] . "</td>";echo "<td>" . $row["start_date"] .
"</td>";echo "<td>" . $row["return_date"] . "</td>";
            echo "<td>" . $row["place"] . "</td>";echo "<td>" . $row["purpose"] .
"</td>"; echo "<td>" . $row["parents_number"] . "</td>";
            echo "<td>" . $row["created_at"] . "</td>";$startDate = new
DateTime($row["start_date"]);$returnDate = new DateTime($row["return_date"]);
            $interval = $startDate->diff($returnDate); $numberOfDays = $interval->days;
            echo "<td>" . ($numberOfDays == 1 ? $numberOfDays . " day" :
$numberOfDays . " days") . "</td>";echo "<td>";
            echo "<form method='post' action='delete_leave.php'>";
            echo "<input type='hidden' name='delete_id' value='" . $row["id"] . "'>";
            echo "<button type='submit' name='remove' onclick='return confirm(\"Are
you sure you want to remove this entry?\")'>Remove</button>";
            echo "</form>";echo "</td>"; echo "</tr>";}
    } else {echo "<tr><td colspan='9'>No leave form history available</td></tr>";}?>
</table>    </div>
    <script>
        function toggleSidebar() {
        var sidebar = document.getElementById("sidebar");
        var mainContent = document.getElementById("mainContent");
        if (sidebar.style.display === "none" || sidebar.style.display === "") {
            sidebar.style.display = "block";
            mainContent.style.marginLeft = "250px";
        } else {sidebar.style.display = "none";mainContent.style.marginLeft = "0";}}
    </script></body></html>
```

monthlypayment.php:

```
<?php
require "config.php";
if (!isset($_SESSION['login'])) {
```

```
    header("Location: login.php");
    exit();}
$user_id = $_SESSION['id'];
$userNameQuery = "SELECT name FROM users WHERE id = '$user_id'";
$userNameResult = mysqli_query($conn, $userNameQuery);
if ($userNameResult) {
    $userNameRow = mysqli_fetch_assoc($userNameResult);
    $userName = $userNameRow['name'];
} else {$userName = 'Guest'; }
$bookingQuery = "SELECT food_status FROM bookings WHERE user_id =
'$user_id'";
$bookingResult = mysqli_query($conn, $bookingQuery);
if ($bookingResult) {
    $bookingRow = mysqli_fetch_assoc($bookingResult);
    $foodStatus = $bookingRow['food_status'];
    if ($foodStatus == 1) {$monthlyFees = 6000;
    } else {$monthlyFees = 3000;}
} else {$monthlyFees = 3000;}
$previousMonth = date("Y-m-d H:i:s", strtotime("last month"));
$totalFoodPriceQuery = "SELECT COALESCE(SUM(price), 0) AS total_price
FROM food_order WHERE user_id = '$user_id' AND MONTH(created_at) =
MONTH('$previousMonth') AND YEAR(created_at) = YEAR('$previousMonth')";
$totalFoodPriceResult = mysqli_query($conn, $totalFoodPriceQuery);
if ($totalFoodPriceResult) {
    $totalFoodPriceRow = mysqli_fetch_assoc($totalFoodPriceResult);
    $totalMessFoodPrice = $totalFoodPriceRow['total_price'];
} else {
    echo "Error: " . mysqli_error($conn);
    $totalMessFoodPrice = 0; }
$totalAmount = $monthlyFees + $totalMessFoodPrice;
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $user_id = $_POST['user_id'];
    $name = $_POST['name'];
    $monthly_fees = $_POST['monthly_fees'];
```

```
    $total_mess_food_price = $_POST['total_mess_food_price'];
    $total_amount = $_POST['total_amount'];
    $transaction_id = $_POST['transaction_id'];
    $currentDateTime = date("Y-m-d H:i:s");
    $currentMonth = date("Y-m");
    $existingPaymentQuery = "SELECT id FROM month_payment WHERE user_id =
'$user_id' AND created_at >= '$currentMonth-01 00:00:00' AND created_at <=
'$currentMonth-31 23:59:59'";
    $existingPaymentResult = mysqli_query($conn, $existingPaymentQuery);
    if (mysqli_num_rows($existingPaymentResult) > 0) {
        echo "<script>alert('You have already paid for the current month. Payment for
the next month can be made after the current month ends.')</script>";
    } else {
        $insertPaymentQuery = "INSERT INTO month_payment (user_id, name,
monthlyfees, total_mess_food_price, total_amount, transaction_id) VALUES
('$user_id', '$name', '$monthly_fees', '$total_mess_food_price', '$total_amount',
'$transaction_id')";
        if (mysqli_query($conn, $insertPaymentQuery)) {
            echo "<script>alert('Payment successfully submitted!')</script>";
        } else {
            echo "Error: " . mysqli_error($conn);}}}
$paymentRecordsQuery = "SELECT * FROM month_payment WHERE user_id =
'$user_id'";
$paymentRecordsResult = mysqli_query($conn, $paymentRecordsQuery);
$paymentTable = "<h3>Payment History</h3><table
class='table'><thead><tr><th>ID</th><th>User
ID</th><th>Name</th><th>Monthly Fees</th><th>Total Mess Food
Price</th><th>Total Amount</th><th>Transaction ID</th><th>Created
At</th></tr></thead><tbody>";
if ($paymentRecordsResult) {
    while ($row = mysqli_fetch_assoc($paymentRecordsResult)) {
        $paymentTable .= "<tr>";
        $paymentTable .= "<td>{$row['id']}</td>";
        $paymentTable .= "<td>{$row['user_id']}</td>";
```

```
            $paymentTable .= "<td>{$row['name']}</td>";
            $paymentTable .= "<td>{$row['monthlyfees']}</td>";
            $paymentTable .= "<td>{$row['total_mess_food_price']}</td>";
            $paymentTable .= "<td>{$row['total_amount']}</td>";
            $paymentTable .= "<td>{$row['transaction_id']}</td>";
            $paymentTable .= "<td>{$row['created_at']}</td>";$paymentTable .= "</tr>";}
        } else {
           $paymentTable .= "<tr><td colspan='8'>No payment records found.</td></tr>";}
        $paymentTable .= "</tbody></table>"; echo $paymentTable; ?>
```

feedback.php:

```
<?php
require "config.php";
if (!isset($_SESSION['login'])) {
   header("Location: login.php");
   exit();}
$user_id = $_SESSION['id'];
$userNameQuery = "SELECT name FROM users WHERE id = '$user_id'";
$userNameResult = mysqli_query($conn, $userNameQuery);
if ($userNameResult) {
   $userNameRow = mysqli_fetch_assoc($userNameResult);
   $userName = $userNameRow['name'];
}else {$userName = 'Guest'; }
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);}
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $userID = isset($_SESSION["id"]) ? $_SESSION["id"] : null;
   $queryText = isset($_POST['query']) ? $_POST['query'] : null;
   $adminReply = isset($_POST['admin_reply']) ? $_POST['admin_reply'] : 'Admin will
reply soon';
   $validateUserQuery = "SELECT * FROM users WHERE id = '$userID'";
```

```
    $result = $conn->query($validateUserQuery);
    if ($result->num_rows > 0) {
        $insertQuery = "INSERT INTO user_query (user_id, query_text, admin_reply)
VALUES ('$userID', '$queryText', '$adminReply')";
        if ($conn->query($insertQuery) === TRUE) {// Query submitted successfully
        } else {echo "Error: " . $insertQuery . "<br>" . $conn->error;}
    } else {echo "Invalid user ID.";}}?>
        <?php
        $userID = isset($_SESSION['id']) ? $_SESSION['id'] : null;
        $fetchQueriesQuery = "SELECT * FROM user_query WHERE user_id = '$userID'";
        $queriesResult = $conn->query($fetchQueriesQuery);
        if ($queriesResult === FALSE) {
            echo "Error in fetchQueriesQuery: " . $conn->error;
        } elseif ($queriesResult->num_rows > 0) {
            echo "<label><h2>Previous Queries:</h2></label>";
            echo "<table border='1'>";
            echo "<tr><th>Query</th><th>Admin Reply</th><th>Action</th></tr>";
            while ($row = $queriesResult->fetch_assoc()) {
                echo "<tr>";echo "<td>{$row['query_text']}</td>";
                echo "<td>{$row['admin_reply']}</td>";echo "<td>
                    <form action='delete_feedback.php' method='post'>
                        <input type='hidden' name='deleteQueryID' value='{$row['id']}'>
                        <button type='submit' name='deleteQuery'>Remove</button>
                    </form></td>";echo "</tr>";}} else {
            echo "No previous queries found.";}?></div></body></html>
```

logout.php:

```
<?php
require 'config.php';
$_SESSION=[];session_unset();session_destroy();header("Location: ../login.html");?>
```

delete_feedback.php:

```
<?php
require "config.php";
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    if (isset($_POST['deleteQuery'])) {
        if (isset($_POST['deleteQueryID'])) {
            $deleteQueryID = $_POST['deleteQueryID'];
            $deleteQuery = "DELETE FROM user_query WHERE id = '$deleteQueryID'";
            if ($conn->query($deleteQuery) === TRUE) {
                header("Location: feedback.php");
                exit();} else {echo "Error: " . $deleteQuery . "<br>" . $conn->error; }
        } else {echo "Error: deleteQueryID is not set."; } }}?>
```

delete_leave.php:

```
<?php
require "config.php";
if (isset($_POST['remove'])) {
    $delete_id = $_POST['delete_id'];
    $deleteQuery = "DELETE FROM leave_form WHERE id = '$delete_id'";
    if ($conn->query($deleteQuery) === TRUE) {
        header("Location: leaveform.php");
        exit();
    } else {echo "Error: " . $deleteQuery . "<br>" . $conn->error;}
    if ($conn->query($deleteQuery) === TRUE) {echo "<script>alert('Entry removed
successfully!')</script>";}
    else {echo "Error: " . $deleteQuery . "<br>" . $conn->error;}}?>
```

edit-user.php:

```
<?php
require 'config.php';
if (isset($_GET['id'])) {$id = $_GET['id'];
```

```
$result = mysqli_query($conn, "SELECT * FROM users WHERE id=$id");

$user = mysqli_fetch_assoc($result);}

if (isset($_POST['submit'])) {

    $id = $_POST['id']; $name = $_POST['name'];

    $father_name = $_POST['father_name']; $dob = $_POST['dob'];

    $address = $_POST['address']; $phone_number = $_POST['phone_number'];

    $email = $_POST['email']; $password = $_POST['password'];

    mysqli_query($conn, "UPDATE users SET name='$name', father_name='$father_name',
dob='$dob', address='$address', phone_number='$phone_number', email='$email',
password='$password' WHERE id=$id");

    header('Location: user_dashboard.php');}?>
```

update_seats.php:

```
<?php

require 'config.php';

if(isset($_GET['userid']) && isset($_GET['room'])) {

    $room = $_GET['room'];

    $userid = $_GET['userid'];

    $checkPaymentQuery = "SELECT * FROM payment WHERE user_id = '$userid'";

    $paymentResult = mysqli_query($conn, $checkPaymentQuery);

    if ($paymentResult && mysqli_num_rows($paymentResult) > 0) {

    $updateQuery = "UPDATE room SET occupied_seats = occupied_seats + 1,
available_seats = available_seats - 1 WHERE room_number = '$room' AND available_seats
> 0";

    if(mysqli_query($conn, $updateQuery)) {

        $fetchQuery = "SELECT available_seats FROM room WHERE room_number =
'$room'";

        $result = mysqli_query($conn, $fetchQuery);

        if ($result && $row = mysqli_fetch_assoc($result)) {echo $row['available_seats'];

        } else {echo "Error fetching available seats.";}

    } else {echo "Error updating seats.";}} else {echo "User has no payment. Please make a
payment first.";}} else {echo "Invalid request.";}mysqli_close($conn);?>
```

ADMIN:

adminsignup.php:

```
<?php
require 'config.php';
if (isset($_POST["submit"])) {
   $name=$_POST["name"];$email = $_POST["email"];
   $password = $_POST["password"];$code = $_POST["code"];
   if ($code === "admin123") {
      $duplicate = mysqli_query($conn, "SELECT * FROM admins WHERE email='$email'");
      if (mysqli_num_rows($duplicate) > 0) {
         echo "<script> alert('Email is already registered'); </script>";
      } else {
         $query = "INSERT INTO admins (name,email, password) VALUES ('$name','$email', '$password')";
         mysqli_query($conn, $query);
         echo "<script> alert('Admin registration successful'); window.location.href = 'adminlogin.php'; </script>";}
   } else {
      echo "<script> alert('Incorrect code. You need a valid code to register as an admin.'); </script>";}}?>
```

adminlogin.php:

```
<?php
require 'config.php';
if (isset($_POST["submit"])) {
   $email = $_POST["email"];
   $password = $_POST["password"];
   $result = mysqli_query($conn, "SELECT * FROM admins WHERE email='$email'");
   $row = mysqli_fetch_assoc($result);
   if (mysqli_num_rows($result) > 0 && $password==$row["password"]) {
```

```
        session_start();
        $_SESSION["admin_login"] = true;
        $_SESSION["admin_id"] = $row["id"];
        header("Location: admin_dashboard.php");
    } else echo "<script> alert('Invalid credentials'); </script>";}}?>
```

admin_dashboard.php:

```
<?php
require 'config.php';
if (!isset($_SESSION["admin_login"])) {
    header("Location: adminlogin.php");
    exit();}
$adminId = $_SESSION["admin_id"];
$result = mysqli_query($conn, "SELECT * FROM admins WHERE id='$adminId'");
$admin = mysqli_fetch_assoc($result);
$sql="SELECT * FROM admins";
$result=$conn->query($sql);
$currentDateTime = date("Y-m-d H:i:s");
$filterCondition = "";
if (isset($_POST['filterDate']) && !empty($_POST['filterDate']) &&
isset($_POST['filterTime']) && !empty($_POST['filterTime'])) {
    $filterDate = $_POST['filterDate']; $filterTime = $_POST['filterTime'];
    $filterDateTime = "$filterDate $filterTime";
$filterCondition = "AND '$filterDateTime' BETWEEN start_date AND return_date";}
$leaveFormQuery = "SELECT * FROM leave_form WHERE start_date <=
'$currentDateTime' AND return_date >= '$currentDateTime'";
$leaveFormResult = mysqli_query($conn, $leaveFormQuery);
if (!$leaveFormResult) {
    die('Error in query: ' . mysqli_error($conn)); }
$studentsOnLeave = 0;
```

```
while ($leaveFormRow = mysqli_fetch_assoc($leaveFormResult)) {
   $startDateTime = strtotime($leaveFormRow['start_date']);
   $endDateTime = strtotime($leaveFormRow['return_date']);
   if ($startDateTime <= strtotime($currentDateTime) && $endDateTime >=
strtotime($currentDateTime)) {
      $studentsOnLeave++; }}
$studentsInHostelQuery = "SELECT COUNT(*) as count FROM bookings";
$studentsInHostelResult = mysqli_query($conn, $studentsInHostelQuery);
if (!$studentsInHostelResult) {
   die('Error in query: ' . mysqli_error($conn)); }
$studentsInHostelRow = mysqli_fetch_assoc($studentsInHostelResult);
$studentsInHostel = $studentsInHostelRow['count'];
$studentsInHostel = $studentsInHostel - $studentsOnLeave;?>
<script>
   function searchAdmin() {
         const searchValue = document.getElementById('searchInput').value.toLowerCase();
         const tableRows = document.querySelectorAll("#mainContent table tr");
         tableRows.forEach(row => {
            if (row.getElementsByTagName('td').length > 0) {
               const name = row.getElementsByTagName('td')[1].innerText.toLowerCase();
               const id = row.getElementsByTagName('td')[0].innerText.toLowerCase();
               row.style.display = (name.includes(searchValue) || id.includes(searchValue)) ?
'table-row' : 'none'; }});}
      function showAll() {
         const tableRows = document.querySelectorAll("#mainContent table tr");
         tableRows.forEach(row => {
            row.style.display = 'table-row';});}
</script></body></html>
```

admin_manageuser.php:

```
<?php
require 'config.php';
if (!isset($_SESSION["admin_login"])) {
   header("Location: adminlogin.php");
   exit();}
$adminId = $_SESSION["admin_id"];
$result = mysqli_query($conn, "SELECT * FROM admins WHERE id='$adminId'");
$admin = mysqli_fetch_assoc($result);
$sql="Select * from users";
$result=$conn->query($sql);?>
      <h1>Manage Users:</h1>
       <form class="form-inline mb-3">
          <input class="form-control mr-sm-2" type="text" placeholder="Search by Name or
ID" id="searchInput">
          <button class="btn btn-outline-primary my-2 my-sm-0" type="button"
onclick="searchUser()">Search</button>
          <button class="btn btn-outline-secondary my-2 my-sm-0" type="button"
onclick="showAll()">Show All</button>
      </form>
      <table>
         <tr>
            <th>Id: </th><th>Name: </th><th>Father Name: </th>
            <th>Date of Birth: </th><th>Address: </th><th>Phone Number: </th>
            <th>Email: </th><th>Password: </th><th>Edit</th><th>Remove</th></tr>
         <?php
         if($result-> num_rows>0){
            while($row=$result->fetch_assoc()){
               echo
"<tr><td>".$row["id"]."</td><td>".$row["name"]."</td><td>".$row["father_name"]."</td><
td>".$row["dob"]."</td><td>"
```

```
.$row["address"]."</td><td>".$row["phone_number"]."</td><td>".$row["email"]."</td><td
>".$row["password"]."</td>

                <td><a href='edit-user.php?id={$row['id']}'class='btn btn-
primary'>Edit</a></td>

                <td><a href='delete-user.php?id={$row['id']}'class='btn btn-danger'
onclick='return confirm(\"Are you sure?\");'>Remove</a></td></tr>";}

            echo"</table>";}

        else{echo"0 result";}$conn->close();?></table></div><script>

    function searchUser() {

        const searchValue = document.getElementById('searchInput').value.toLowerCase();

        const tableRows = document.querySelectorAll("#mainContent table tr");

        tableRows.forEach(row => {

            if (row.getElementsByTagName('td').length > 0) {

                const name = row.getElementsByTagName('td')[1].innerText.toLowerCase();

                const id = row.getElementsByTagName('td')[0].innerText.toLowerCase();

                row.style.display = (name.includes(searchValue) || id.includes(searchValue)) ?
'table-row' : 'none';} });}

      function showAll() {

        const tableRows = document.querySelectorAll("#mainContent table tr");

        tableRows.forEach(row => {row.style.display = 'table-row'; });}

</script></body></html>
```

vieworder.php:

```
<?php

require "config.php";

if (!isset($_SESSION["admin_login"])) {

   header("Location: adminlogin.php");

   exit();}

if ($conn->connect_error) {die("Connection failed: " . $conn->connect_error);}?>

      <button class="btn btn-success" onclick="location.href='add_food.php'">Add
Food</button>

      <h1>User Food Orders:</h1>
```

```
<table border="1"><thead><tr><th>User ID</th><th>Name</th>
    <th>Room Number</th><th>Actions</th></tr></thead><tbody>
<?php
$query = "SELECT users.id, users.name, bookings.selected_room FROM users
    LEFT JOIN bookings ON users.id = bookings.user_id";
$result = $conn->query($query);
if ($result === FALSE) {echo "Error in query: " . $conn->error;}
while ($row = $result->fetch_assoc()) {echo "<tr>";
    echo "<td>{$row['id']}</td>";echo "<td>{$row['name']}</td>";
    echo "<td>{$row['selected_room']}</td>";
    echo "<td><a href='vieworder.php?user_id={$row['id']}' class='btn btn-
info'>View Orders</a></td>";echo "</tr>";}?>
</tbody></table></div></body></html>
```

monthpayment.php:

```
<?php
require 'config.php';
if (!isset($_SESSION["admin_login"])) {
   header("Location: adminlogin.php");
   exit();}
$adminId = $_SESSION["admin_id"];
$result = mysqli_query($conn, "SELECT * FROM admins WHERE id='$adminId'");
$admin = mysqli_fetch_assoc($result);
$sql = "SELECT id, user_id, name, monthlyfees, total_mess_food_price, total_amount,
transaction_id, created_at FROM month_payment";
$result = $conn->query($sql);?>
   <div class="main-content" id="mainContent">
      <h1>Monthly Payment Details: </h1>
      <form class="form-inline mb-3">
         <input class="form-control mr-sm-2" type="search" placeholder="Search by UserID"
id="searchInput">
```

```
        <button class="btn btn-outline-primary my-2 my-sm-0" type="button" onclick="searchPayment()">Search</button>

        <button class="btn btn-outline-secondary my-2 my-sm-0" type="button" onclick="showAll()">Show All</button>

    </form>

    <table>

        <tr>

            <th>ID</th><th>User ID</th><th>Name</th><th>Monthly Fees</th>

            <th>Total Mess Food Price</th><th>Total Amount</th><th>Transaction ID</th>

            <th>Created At</th></tr>

        <?php

        if ($result->num_rows > 0) {

            while ($row = $result->fetch_assoc()) {

                echo "<tr><td>" . $row["id"] . "</td><td>" . $row["user_id"] . "</td><td>" . $row["name"] . "</td><td>" . $row["monthlyfees"] . "</td><td>"

                    . $row["total_mess_food_price"] . "</td><td>" . $row["total_amount"] . "</td><td>" . $row["transaction_id"] . "</td><td>" . $row["created_at"] . "</td><!-- Change this line in monthpayment.php -->

                    </tr>";}

            echo "</table>";} else {echo "0 result";}$conn->close();?></table></div>

<script>

    function searchPayment() {

        const searchValue = document.getElementById('searchInput').value;

        const tableRows = document.querySelectorAll("#mainContent table tr");

        tableRows.forEach(row => {

            if (row.getElementsByTagName('td').length > 0) {

                const userId = row.getElementsByTagName('td')[1].innerText;

                row.style.display = (userId.includes(searchValue)) ? 'table-row' : 'none';

            }});}

    function showAll() {

        const tableRows = document.querySelectorAll("#mainContent table tr");
```

```
        tableRows.forEach(row => {
            row.style.display = 'table-row'; });}    </script>
```

transaction.php:

```
  <h1>User Details With Payments: </h1>
   <form class="form-inline mb-3">
        <input class="form-control mr-sm-2" type="text" placeholder="Search by Name or ID" id="searchInput">
        <button class="btn btn-outline-primary my-2 my-sm-0" type="button" onclick="searchUser()">Search</button>
        <button class="btn btn-outline-secondary my-2 my-sm-0" type="button" onclick="showAll()">Show All</button>
   </form>
   <?php
      require 'config.php';
      if (!isset($_SESSION["admin_login"])) {
         header("Location: adminlogin.php");
         exit();}
      $userQuery = "SELECT id, name FROM users";
      $archivedUserQuery = "SELECT id, name FROM archived_users";
      $combinedQuery = "($userQuery) UNION ($archivedUserQuery)";
      $combinedResult = mysqli_query($conn, $combinedQuery);
      $paymentQuery = "SELECT user_id, payment_id, transaction_id, payment_date FROM payment";
      $paymentResult = mysqli_query($conn, $paymentQuery);
      if ($combinedResult && $paymentResult) {
         $users = mysqli_fetch_all($combinedResult, MYSQLI_ASSOC);
         $payments = mysqli_fetch_all($paymentResult, MYSQLI_ASSOC);
         echo '<table border="1" id="userTable">';
         echo '<tr><th>User ID</th><th>Name</th><th>Payment ID</th><th>Transaction ID</th><th>Payment Date</th></tr>';
```

```
        foreach ($users as $user) {
            $userPayment = array_filter($payments, function ($payment) use ($user) {
                return $payment['user_id'] == $user['id'];});
            if (!empty($userPayment)) {
                $userPayment = current($userPayment);
                echo '<tr>'; echo '<td>' . $user['id'] . '</td>'; echo '<td>' . $user['name'] . '</td>';
                echo '<td>' . $userPayment['payment_id'] . '</td>';
                echo '<td>' . $userPayment['transaction_id'] . '</td>';
                echo '<td>' . $userPayment['payment_date'] . '</td>'; echo '</tr>';
            } else {echo '<tr>'; echo '<td>' . $user['id'] . '</td>';
                echo '<td>' . $user['name'] . '</td>';
                echo '<td colspan="3">No payment details found</td>'; echo '</tr>';}}
        echo '</table>';
    } else {echo 'Error fetching data: ' . mysqli_error($conn);}mysqli_close($conn); ?>
</div></body></html>
```

bookings.php:

```
<?php
require 'config.php';
if (!isset($_SESSION["admin_login"])) {
   header("Location: adminlogin.php");
   exit();}
$adminId = $_SESSION["admin_id"];
$result = mysqli_query($conn, "SELECT * FROM admins WHERE id='$adminId'");
$admin = mysqli_fetch_assoc($result);
$sql="Select
booking_id,user_id,start_date,food_status,guardian_name,relation,guardian_contact,emergency_contact,selected_room from bookings";
$result=$conn->query($sql);
?>
```

```
    <h1>Booking Details: </h1>

    <form class="form-inline mb-3">

        <input class="form-control mr-sm-2" type="search" placeholder="Search by UserID" id="searchInput">

        <button class="btn btn-outline-primary my-2 my-sm-0" type="button" onclick="searchBooking()">Search</button>

        <button class="btn btn-outline-secondary my-2 my-sm-0" type="button" onclick="showAll()">Show All</button>

      </form>

        <?php

        if($result-> num_rows>0){

            while($row=$result->fetch_assoc()){

                echo "<tr><td>".$row["booking_id"]."</td><td>".$row["user_id"]."</td><td>".$row["start_date"]."</td><td>".$row["food_status"]."</td><td>".$row["guardian_name"]."</td><td>".$row["relation"]."</td><td>".$row["guardian_contact"]."</td><td>".$row["emergency_contact"]."</td><td>".$row["selected_room"]."</td><!-- Change this line in bookings.php -->

                <td><a href='edit_bookings.php?user_id={$row['user_id']}'class='btn btn-primary'>Edit</a></td>

                <td><a href='delete_bookings.php?user_id={$row['user_id']}'class='btn btn-danger' onclick='return confirm(\"Are you sure?\");'>Remove</a></td></tr>";

            }echo"</table>";}

        else{ echo"0 result";}$conn->close();?></table></div></body></html>
```

userfiles.php:

```
<?php

require 'config.php';

if (!isset($_SESSION["admin_login"])) {

    header("Location: adminlogin.php");

    exit();}

$adminId = $_SESSION["admin_id"];

$result = mysqli_query($conn, "SELECT * FROM admins WHERE id='$adminId'");

$admin = mysqli_fetch_assoc($result);
```

```
$sql = "SELECT uf.photo_filename AS photo, uf.id_proof_filename AS proof, u.name,u.id
        FROM user_files uf
        JOIN users u ON uf.user_id = u.id";
$result = $conn->query($sql);
?>
    <h1>User Files:</h1>
    <form class="form-inline mb-3">
            <input class="form-control mr-sm-2" type="search" placeholder="Search UserName" id="searchInput">
            <button class="btn btn-outline-primary my-2 my-sm-0" type="button" onclick="searchUser()">Search</button>
            <button class="btn btn-outline-secondary my-2 my-sm-0" type="button" onclick="showAll()">Show All</button>
        </form>
        <table><tr><th>User Name:</th><th>Id: </th><th>Photo:</th><th>Proof:</th></tr>
            <?php
            if ($result->num_rows > 0) {
                while ($row = $result->fetch_assoc()) {
                    echo "<tr><td>" . $row["name"] . "</td><td>".$row["id"]."</td>
                        <td><img src='../login/uploads/" . $row["photo"] . "' alt='User Photo' width='100'></td>
                        <td><a href='../login/uploads/" . $row["proof"] . "' target='_blank'>View Proof</a></td></tr>";}echo "</table>";
            } else {echo "0 result";}$conn->close();?></table></div><script></script></body></html>
```

roomdetails.php:

```
<?php
require 'config.php';
if (!isset($_SESSION["admin_login"])) {
    header("Location: adminlogin.php");
    exit();}
$roomQuery = "SELECT room_number FROM room";
```

```
$roomResult = mysqli_query($conn, $roomQuery);
$selectedRoom = isset($_POST['room_number']) ? $_POST['room_number'] : null;
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $selectedRoom = $_POST['room_number'];
    $sql = "SELECT b.user_id, u.name, b.start_date, b.duration, b.selected_room
            FROM bookings b
            JOIN users u ON b.user_id = u.id
            WHERE b.selected_room = '$selectedRoom'";
    $result = $conn->query($sql);
    if (!$result) {die("Error: " . $conn->error . "<br>SQL: " . $sql);}}?>
    <h1>Room Details</h1>
        <form method="post" id="roomDetailsForm">
            <label for="room_number">Select Room Number:</label>
            <select id="room_number" name="room_number" required
onchange="document.getElementById('roomDetailsForm').submit()">
                <?php
                while ($roomRow = mysqli_fetch_assoc($roomResult)) {
                    $roomValue = $roomRow['room_number'];
                    $isSelected = (isset($selectedRoom) && $selectedRoom === $roomValue) ?
'selected' : '';
                    echo "<option value='$roomValue' $isSelected> $roomValue</option>";}?>
            </select><br><br></form>
        <?php
        if (isset($result) && $result->num_rows > 0) {
            echo "<table><tr> <th>User ID</th><th>User Name</th><th>Start Date</th>
                    <th>Duration</th><th>Room</th></tr>";
            while ($row = $result->fetch_assoc()) {
                echo "<tr><td>" . $row['user_id'] . "</td><td>" . $row['name'] . "</td>
                    <td>" . $row['start_date'] . "</td><td>" . $row['duration'] . "</td>
                    <td>" . $row['selected_room'] . "</td></tr>";}echo "</table>";
```

```
    } else {echo "No persons in ($selectedRoom).";}?> </div>
  <script></body></html>
```

addrooms.php:

```
<?php
require "config.php";
if (!isset($_SESSION["admin_login"])) {
  header("Location: adminlogin.php");
  exit();}
$sql = "SELECT * FROM room";
$result = mysqli_query($conn, $sql);
?>
 <h2>Room Details</h2>
      <div>
      <a href="newroom.php" class="btn btn-primary">Add Rooms</a>
      </div>      <br>
      <table class="table"><thead><tr>
            <th>Room Number</th><th>Available Seats</th><th>Occupied Seats</th>
            <th>Total Seats</th><th>Image: </th><th>Edit</th><th>Remove</th></tr>
        </thead><tbody>
          <?php
          while ($row = mysqli_fetch_assoc($result)) {
            echo "<tr>";echo "<td>{$row['room_number']}</td>";
            echo "<td>{$row['available_seats']}</td>";
            echo "<td>{$row['occupied_seats']}</td>";
            echo "<td>{$row['total_seats']}</td>";
            echo "<td><img src='{$row['image_path']}' alt='Room Image' style='width:
200px; height: 100px;'></td>";
            echo "<td><a href='edit_room.php?id={$row['room_number']}' class='btn btn-
primary'>Edit</a></td>";
            echo "<td><a href='delete_room.php?id={$row['room_number']}' class='btn btn-
danger' onclick='return confirm(\"Are you sure?\");'>Remove</a></td>";
```

```
            echo "</tr>"; }?></tbody></table></div>
    <script></body></html>
```

userreport.php:

```
<?php
require "config.php";
if (!isset($_SESSION['admin_login'])) {
    header("Location: adminlogin.php");
    exit();}
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);}
$query = "SELECT u.name,u.id, u.father_name, u.dob, u.phone_number, b.start_date,
b.duration, b.food_status, b.guardian_name, b.relation, b.guardian_contact,
b.emergency_contact, b.total_fees, b.selected_room,
p.transaction_id,p.payment_date,uf.photo_filename, b.created_at FROM users u
JOIN bookings b ON u.id = b.user_id JOIN user_files uf ON u.id = uf.user_id
JOIN payment p ON u.id = p.user_id";
$result = $conn->query($query);
if ($result === false) {die("Query error: " . $conn->error);}?>
<form class="form-inline mb-3">
        <input class="form-control mr-sm-2" type="search" placeholder="Search UserName"
id="searchInput">
        <button class="btn btn-outline-primary my-2 my-sm-0" type="button"
onclick="searchUser()">Search</button>
        <button class="btn btn-outline-secondary my-2 my-sm-0" type="button"
onclick="showAll()">Show All</button></form>
    <?php
while ($row = $result->fetch_assoc()) {
    echo "<div class='invoice' id='invoice{$row['id']}'>";
    echo "<div class='table-responsive'>";echo "<table class='table'>";echo "<tr>";
    echo "<th>Created At: </th>";echo "<td>{$row['created_at']}</td>";echo "</tr>";
    echo "<tr>";echo "<th>User Id:</th>";echo "<td>{$row['id']}</td>";echo "</tr>";
    echo "<tr>";echo "<th>User Photo: </th>";
```

```
    echo "<td><img src='../login/uploads/{$row['photo_filename']}' alt='User Photo'
style='width: 80px; height: 80px'></td>";echo "</tr>";echo "<tr>";

    echo "<th>Name: </th>";echo "<td>{$row['name']}</td>";echo "</tr>";

    echo "<tr>";echo "<th>Father Name: </th>";echo "<td>{$row['father_name']}</td>";

    echo "</tr>";echo "<tr>";echo "<th>DOB: </th>";echo "<td>{$row['dob']}</td>";

    echo "</tr>";echo "<tr>";echo "<th>Phone Number: </th>";

    echo "<td>{$row['phone_number']}</td>";echo "</tr>";echo "<tr>";

    echo "<th>Start Date: </th>";echo "<td>{$row['start_date']}</td>";echo "</tr>";

    echo "<tr>";echo "<th>Duration: </th>";echo "<td>{$row['duration']}</td>";echo "</tr>";

    echo "<tr>";echo "<th>Food Status: </th>";echo "<td>{$row['food_status']}</td>";

    echo "</tr>";echo "<tr>";echo "<th>Guardian Name: </th>";

    echo "<td>{$row['guardian_name']}</td>";echo "</tr>";

    echo "<tr>";echo "<th>Relation: </th>";echo "<td>{$row['relation']}</td>";echo "</tr>";

    echo "<tr>";echo "<th>Guardian Contact: </th>";

    echo "<td>{$row['guardian_contact']}</td>";echo "</tr>";echo "<tr>";

    echo "<th>Emergency Contact: </th>";echo "<td>{$row['emergency_contact']}</td>";

    echo "</tr>";echo "<tr>";echo "<th>Total Fees: </th>";

    echo "<td>{$row['total_fees']}</td>"; echo "</tr>";echo "<tr>";

    echo "<th>Selected Room: </th>";echo "<td>{$row['selected_room']}</td>";

    echo "</tr>";echo "<tr>";echo "<th>Transaction Id: </th>";

    echo "<td>{$row['transaction_id']}</td>";echo "</tr>";

    echo "<tr>"; echo "<th>Transaction Date: </th>";

    echo "<td>{$row['payment_date']}</td>";echo "</tr>";echo "</table>";echo "</div>";

    echo "<center><button onclick='downloadReportPDF({$row['id']})' class='btn btn-
primary'>Download PDF</button></center>";echo "</div>";}?></div>

    <script>

        function downloadReportPDF(id) {

            const element = document.getElementById(`invoice${id}`);

            html2pdf().from(element).save();}

         function searchUser() {
```

```
        const searchValue = document.getElementById('searchInput').value.toLowerCase();

        <?php

        $result->data_seek(0); // Reset result set pointer

        while ($row = $result->fetch_assoc()) {

            echo "const userRow{$row['id']} =
document.getElementById('invoice{$row['id']}');\n";

            echo "userRow{$row['id']}.style.display =
(userRow{$row['id']}.innerText.toLowerCase().includes(searchValue)) ? 'table' :
'none';\n";}?>}

    function showAll() {

        <?php

        $result->data_seek(0); // Reset result set pointer

        while ($row = $result->fetch_assoc()) {

            echo "const userRow{$row['id']} =
document.getElementById('invoice{$row['id']}');\n";

            echo "userRow{$row['id']}.style.display = 'table';\n";}?>}

  </script></body></html>
```

leaveform.php:

```
<?php

require "config.php";

date_default_timezone_set('Asia/Kolkata');

$leaveFormQuery = "SELECT * FROM leave_form";

$leaveFormResult = $conn->query($leaveFormQuery);?>

 <h1>Leave Form Details:</h1>

    <form method="post">

        <label for="filterDate">Filter by Date:</label>

        <input type="date" id="filterDate" name="filterDate">

        <label for="filterTime">Filter by Time:</label>

        <input type="time" id="filterTime" name="filterTime" value="<?php echo date('H:i');
?>">
```

```
        <button type="submit" class="btn btn-outline-primary my-2 my-sm-
0">Filter</button>

        <button type="button" class="btn btn-outline-primary my-2 my-sm-0"
onclick="showAll()">Show All</button></form>

      <table class="table table-bordered"><thead><tr>

            <th>ID</th><th>User ID</th><th>Name</th><th>Room</th>

            <th>Start Date</th><th>Return Date</th><th>Place</th>

            <th>Purpose</th><th>Parents' Number</th></tr></thead><tbody>

          <?php

          if ($leaveFormResult->num_rows > 0) {

            while ($row = $leaveFormResult->fetch_assoc()) {

              if (isset($_POST['filterDate']) && !empty($_POST['filterDate']) &&
isset($_POST['filterTime']) && !empty($_POST['filterTime'])) {

                $filterDate = $_POST['filterDate']; $filterTime = $_POST['filterTime'];

                $startDateTime = new DateTime($row['start_date']);

                $endDateTime = new DateTime($row['return_date']);

                $filterDateTime = new DateTime($filterDate . ' ' . $filterTime);

                if ($startDateTime <= $filterDateTime && $endDateTime >=
$filterDateTime) {

                  echo "<tr>";echo "<td>" . $row["id"] . "</td>";

                  echo "<td>" . $row["user_id"] . "</td>";

                  echo "<td>" . $row["name"] . "</td>";

                  echo "<td>" . $row["room"] . "</td>";

                  echo "<td>" . $row["start_date"] . "</td>";

                  echo "<td>" . $row["return_date"] . "</td>";

                  echo "<td>" . $row["place"] . "</td>";

                  echo "<td>" . $row["purpose"] . "</td>";

                  echo "<td>" . $row["parents_number"] . "</td>";echo "</tr>";}

              } else {echo "<tr>";echo "<td>" . $row["id"] . "</td>";

                echo "<td>" . $row["user_id"] . "</td>";

                echo "<td>" . $row["name"] . "</td>";
```

```
                    echo "<td>" . $row["room"] . "</td>";
                    echo "<td>" . $row["start_date"] . "</td>";
                    echo "<td>" . $row["return_date"] . "</td>";
                    echo "<td>" . $row["place"] . "</td>";
                    echo "<td>" . $row["purpose"] . "</td>";
                    echo "<td>" . $row["parents_number"] . "</td>";echo "</tr>";}}
            } else {echo "<tr><td colspan='10'>No leave form entries found.</td></tr>";}
            ?></tbody></table></div> </div>
        <script>function showAll() {document.getElementById("filterDate").value = "";
            document.forms[0].submit();}
        </script></body></html>
```

feedbackreply.php:

```
<?php
require "config.php";
if (!isset($_SESSION["admin_login"])) {
    header("Location: adminlogin.php");
    exit();}
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);}
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    if (isset($_POST['saveAdminReply'])) {
        foreach ($_POST['adminReply'] as $queryId => $adminReply) {
            $updateQuery = "UPDATE user_query SET admin_reply = '$adminReply' WHERE
id = '$queryId'";
            if ($conn->query($updateQuery) === TRUE) { } else {echo "Error updating admin
reply for query ID $queryId: " . $conn->error . "<br>";}}}}
$query = "SELECT user_query.id, user_query.user_id, user_query.query_text,
user_query.admin_reply, bookings.selected_room,users.name FROM user_query
LEFT JOIN bookings ON user_query.user_id = bookings.user_id LEFT JOIN users ON
user_query.user_id = users.id WHERE user_query.user_id IN (SELECT id FROM users)";
```

```
$result = $conn->query($query);

if ($result === FALSE) {echo "Error in query: " . $conn->error;}?>
<label><h1>Feedback Reply:</h1></label>

   <form class="form-inline mb-3">

      <input class="form-control mr-sm-2" type="search" placeholder="Search by Name or
ID" id="searchInput">

      <button class="btn btn-outline-primary my-2 my-sm-0" type="button"
onclick="searchUsers()">Search</button>

      <button class="btn btn-outline-secondary my-2 my-sm-0" type="button"
onclick="showAllUsers()">Show All</button>

   </form><form method="POST"><table border="1"><thead><tr>

            <th>User ID</th><th>Name</th><th>Query Text</th>

            <th>Room Number</th><th>Admin Reply</th><th>Action</th></tr>

         </thead><tbody>

            <?php

            while ($row = $result->fetch_assoc()) {

               echo "<tr>";echo "<td>{$row['user_id']}</td>";

               echo "<td>{$row['name']}</td>";echo "<td>{$row['query_text']}</td>";

               echo "<td>{$row['selected_room']}</td>";

               echo "<td><input type='textarea' name='adminReply[{$row['id']}]'
value='{$row['admin_reply']}'></td>";

               echo "<td><button type='submit' name='saveAdminReply[{$row['id']}]'
value='{$row['admin_reply']}'>Save</button></td>";echo "</tr>";}?></tbody>

      </table></form> </div>

</body></html>
```

oldusers.php:

```
<?php

require 'config.php';

if (!isset($_SESSION["admin_login"])) { header("Location: adminlogin.php");exit();}

$adminId = $_SESSION["admin_id"];

$result = mysqli_query($conn, "SELECT * FROM admins WHERE id='$adminId'");
```

```
$admin = mysqli_fetch_assoc($result);

$sql="Select * from archived_users";

$result=$conn->query($sql);?>

   <form class="form-inline mb-3">

      <input class="form-control mr-sm-2" type="search" placeholder="Search by Name or ID" id="searchInput">

      <button class="btn btn-outline-primary my-2 my-sm-0" type="button" onclick="searchUsers()">Search</button>

      <button class="btn btn-outline-secondary my-2 my-sm-0" type="button" onclick="showAllUsers()">Show All</button> </form>

            <?php

            if ($result->num_rows > 0) {

               while ($row = $result->fetch_assoc()) {

                  echo "<tr>";echo "<td>" . $row['id'] . "</td>";

                  echo "<td>" . $row['name'] . "</td>";echo "<td>" . $row['email'] . "</td>";

                  echo "<td>" . $row['father_name'] . "</td>";

                  echo "<td>" . $row['dob'] . "</td>";echo "<td>" . $row['address'] . "</td>";

                  echo "<td>" . $row['phone_number'] . "</td>";

                  echo "<td>" . $row['password'] . "</td>";

                  echo "<td>" . $row['booking_id'] . "</td>";

                  echo "<td>" . $row['start_date'] . "</td>";

                  echo "<td>" . $row['duration'] . "</td>";

                  echo "<td>" . $row['food_status'] . "</td>";

                  echo "<td>" . $row['guardian_name'] . "</td>";

                  echo "<td>" . $row['relation'] . "</td>";

                  echo "<td>" . $row['guardian_contact'] . "</td>";

                  echo "<td>" . $row['emergency_contact'] . "</td>";

                  echo "<td>" . $row['total_fees'] . "</td>";

                  echo "<td>" . $row['selected_room'] . "</td>";

                  echo "<td>" . $row['created_at'] . "</td>";

                  echo "<td>" . $row['transaction_id'] . "</td>";
```

```
                echo "<td>" . $row['payment_date'] . "</td>";

                echo "<td><img src='../login/uploads/" . $row["photo_filename"] . "' alt='User Photo' width='100'></td>";

        echo "<td><a href='../login/uploads/" . $row["id_proof_filename"] . "' target='_blank'>View Proof</a></td>";echo "</tr>";}

            } else {echo "<tr><td colspan='24'>No archived users found</td></tr>";}?>

        </tbody></table></div>

   <script></body></html>
```

add_food.php:

```
<?php

require "config.php";

if ($_SERVER["REQUEST_METHOD"] === "POST") {

   $name = mysqli_real_escape_string($conn, $_POST['name']);

   $price = mysqli_real_escape_string($conn, $_POST['price']);

   $targetDirectory = "uploads/";

   $uploadedFileName = uniqid() . '_' . basename($_FILES['image']['name']);

   $targetFilePath = $targetDirectory . $uploadedFileName;

   $fileType = pathinfo($targetFilePath, PATHINFO_EXTENSION);

   $uploadOk = getimagesize($_FILES["image"]["tmp_name"]) !== false;

   if (file_exists($targetFilePath)) {echo "Error: File already exists."; $uploadOk = 0;}

   if ($_FILES["image"]["size"] > 500000) {echo "Error: File is too large."; $uploadOk = 0;}

   $allowedFormats = array("jpg", "jpeg", "png", "gif");

   if (!in_array($fileType, $allowedFormats)) {

      echo "Error: Only JPG, JPEG, PNG, and GIF files are allowed."; $uploadOk = 0;}

   if ($uploadOk == 0) {echo "Error: Your file was not uploaded.";

   } else {

      if (move_uploaded_file($_FILES["image"]["tmp_name"], $targetFilePath)) {

         $insertQuery = "INSERT INTO foods (name, price, photo_name) VALUES ('$name', '$price', '$uploadedFileName')";

         $result = $conn->query($insertQuery);

         if ($result === FALSE) {echo "Error adding food: " . $conn->error;
```

```
        } else {header("Location: foodorder.php");exit();}
    } else {echo "Error: There was an error uploading your file."; }}}?>
    <h1>Add Food:</h1>
    <form method="post" action="" enctype="multipart/form-data">
        <label for="name">Food Name:</label>
        <input type="text" name="name" required>
        <label for="price">Price:</label>
        <input type="number" name="price" step="0.01" required>
        <label for="image">Select Image:</label>
        <input type="file" name="image" accept="image/*" required>
        <button type="submit" class="btn btn-primary" name="submit">Add Food</button>
    </form></div></body></html>
```

delete_bookings.php:

```
<?php
require 'config.php';
if (isset($_GET['id'])) {
    $userId = $_GET['id'];
    $deleteQuery = "DELETE FROM bookings WHERE user_id = '$userId'";
    if ($conn->query($deleteQuery) === TRUE) {
        echo "<script>alert('Booking record deleted successfully');</script>";
    } else {echo "<script>alert('Error deleting record: " . $conn->error . "');</script>";}}
$conn->close();header("Location: bookings.php");exit();
?>
```

delete_order.php:

```
<?php
require "config.php";
if ($_SERVER["REQUEST_METHOD"] == "GET" && isset($_GET["order_id"])) {
```

```
    $order_id = $_GET["order_id"];
    $delete_query = "DELETE FROM food_order WHERE id = '$order_id'";
    if ($conn->query($delete_query) === TRUE) {
        echo "<script>alert('Order removed successfully!')</script>";
        header("Location: foodorder.php");exit();
    } else {echo "Error removing order: " . $conn->error;}
} else {    echo "Invalid request.";}?>
```

delete_room.php:

```
<?php
require "config.php";
if (isset($_GET['id'])) {
    $roomNumber = $_GET['id'];
    $selectQuery = "SELECT * FROM room WHERE room_number = '$roomNumber'";
    $result = mysqli_query($conn, $selectQuery);
    if ($result && $row = mysqli_fetch_assoc($result)) {
        $roomNumber = $row['room_number'];
        $deleteQuery = "DELETE FROM room WHERE room_number = '$roomNumber'";
        $deleteResult = mysqli_query($conn, $deleteQuery);
        if ($deleteResult) {echo "Room removed successfully.";
        } else {echo "Error removing room: " . mysqli_error($conn);}
    } else {echo "Room not found.";}
} else {echo "Room number not provided.";}?>
```

delete-admin.php:

```
<?php
require 'config.php';
if (isset($_GET['id'])) {
    $id = $_GET['id'];
    mysqli_query($conn, "DELETE FROM admins WHERE id=$id");}
```

```
header('Location: admin_dashboard.php');?>
```

delete-user.php:

```
<?php
require 'config.php';
if (isset($_GET['id'])) {
    $userId = $_GET['id'];
    $checkUserInUsersQuery = "SELECT * FROM users WHERE id = '$userId'";
    $checkUserInUsersResult = mysqli_query($conn, $checkUserInUsersQuery);
    if ($checkUserInUsersResult && mysqli_num_rows($checkUserInUsersResult) > 0) {
        $checkUserInBookingsQuery = "SELECT * FROM bookings WHERE user_id =
'$userId'";
        $checkUserInBookingsResult = mysqli_query($conn, $checkUserInBookingsQuery);
        if ($checkUserInBookingsResult && mysqli_num_rows($checkUserInBookingsResult)
> 0) {                                                    $getSelectedRoomQuery = "SELECT
selected_room FROM bookings WHERE user_id = '$userId'";
            $selectedRoomResult = mysqli_query($conn, $getSelectedRoomQuery);
            if ($selectedRoomResult && $selectedRoomRow =
mysqli_fetch_assoc($selectedRoomResult)) {
                $selectedRoom = $selectedRoomRow['selected_room'];
                $getUserDetailsQuery = "SELECT * FROM users WHERE id = '$userId'";
                $userDetailsResult = mysqli_query($conn, $getUserDetailsQuery);
                $userDetails = mysqli_fetch_assoc($userDetailsResult);
                $getBookingDetailsQuery = "SELECT * FROM bookings WHERE user_id =
'$userId'";
                $bookingDetailsResult = mysqli_query($conn, $getBookingDetailsQuery);
                $bookingDetails = mysqli_fetch_assoc($bookingDetailsResult);
                $getPaymentDetailsQuery = "SELECT * FROM payment WHERE user_id =
'$userId'";
                $paymentDetailsResult = mysqli_query($conn, $getPaymentDetailsQuery);
                $paymentDetails = mysqli_fetch_assoc($paymentDetailsResult);
```

```
$getUserFilesDetailsQuery = "SELECT * FROM user_files WHERE user_id =
'$userId'";

$userFilesDetailsResult = mysqli_query($conn, $getUserFilesDetailsQuery);

$userFilesDetails = mysqli_fetch_assoc($userFilesDetailsResult);

$userFilesFound = ($userFilesDetails !== null);

if ($userFilesFound) {

$insertArchivedUserQuery = "INSERT INTO archived_users VALUES
('$userDetails[id]','$userDetails[name]','$userDetails[father_name]','$userDetails[dob]','$user
Details[address]',$userDetails[phone_number]','$userDetails[email]','$userDetails[password]',
'$bookingDetails[booking_id]','$bookingDetails[start_date]','$bookingDetails[duration]',
'$bookingDetails[food_status]','$bookingDetails[guardian_name]','$bookingDetails[relation]',
'$bookingDetails[guardian_contact]','$bookingDetails[emergency_contact]','$bookingDetails[
total_fees]','$bookingDetails[selected_room]','$bookingDetails[created_at]','$paymentDetails
[transaction_id]','$paymentDetails[payment_date]','$userFilesDetails[photo_filename]',

'$userFilesDetails[id_proof_filename]')";

mysqli_query($conn, $insertArchivedUserQuery);

$deleteUserQuery = "DELETE FROM users WHERE id = $userId";

mysqli_query($conn, $deleteUserQuery);

$deleteBookingQuery = "DELETE FROM bookings WHERE user_id =
'$userId'"; mysqli_query($conn, $deleteBookingQuery);

$deletePaymentQuery = "DELETE FROM payment WHERE user_id =
'$userId'";

mysqli_query($conn, $deletePaymentQuery);

$deleteFilesQuery = "DELETE FROM user_files WHERE user_id = '$userId'";

mysqli_query($conn, $deleteFilesQuery);

$updateRoomQuery = "UPDATE room SET occupied_seats = occupied_seats -
1, available_seats = available_seats + 1 WHERE room_number = '$selectedRoom'";

mysqli_query($conn, $updateRoomQuery);

header('Location: admin_manageuser.php');

exit();

} else{ echo "<script>alert('No user_files found for the user.')</script>";

$insertArchivedUserQuery = "INSERT INTO archived_users VALUES
('$userDetails[id]','$userDetails[name]','$userDetails[father_name]','$userDetails[dob]','$user
Details[address]','$userDetails[phone_number]','$userDetails[email]','$userDetails[password]'
```

,'$bookingDetails[booking_id]','$bookingDetails[start_date]','$bookingDetails[duration]','$bookingDetails[food_status]','$bookingDetails[guardian_name]','$bookingDetails[relation]','$bookingDetails[guardian_contact]','$bookingDetails[emergency_contact]','$bookingDetails[total_fees]','$bookingDetails[selected_room]','$bookingDetails[created_at]','$paymentDetails[transaction_id]','$paymentDetails[payment_date]', NULL, NULL)";

```
                mysqli_query($conn, $insertArchivedUserQuery);
                $deleteUserQuery = "DELETE FROM users WHERE id = $userId";
                mysqli_query($conn, $deleteUserQuery);
                $deleteBookingQuery = "DELETE FROM bookings WHERE user_id = '$userId'";
                mysqli_query($conn, $deleteBookingQuery);
                $deletePaymentQuery = "DELETE FROM payment WHERE user_id = '$userId'";
                mysqli_query($conn, $deletePaymentQuery);
                $updateRoomQuery = "UPDATE room SET occupied_seats = occupied_seats - 1, available_seats = available_seats + 1 WHERE room_number = '$selectedRoom'";
                mysqli_query($conn, $updateRoomQuery);
                header('Location: admin_manageuser.php');
                exit();}}
        } else {
            echo "<script>alert('No booking found for the user.'); window.location.href = 'admin_manageuser.php';</script>";
            $getUserDetailsQuery = "SELECT * FROM users WHERE id = '$userId'";
            $userDetailsResult = mysqli_query($conn, $getUserDetailsQuery);
            $userDetails = mysqli_fetch_assoc($userDetailsResult);
            $insertArchivedUserQuery = "INSERT INTO archived_users VALUES ('$userDetails[id]','$userDetails[name]','$userDetails[father_name]','$userDetails[dob]','$userDetails[address]','$userDetails[phone_number]','$userDetails[email]','$userDetails[password]',NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL)";
            mysqli_query($conn, $insertArchivedUserQuery);
```

```
        $deleteUserQuery = "DELETE FROM users WHERE id = $userId";
        mysqli_query($conn, $deleteUserQuery);}
    } else {echo "<script>alert('No user found with the specified ID.')</script>";}
} else {echo "Error: User ID not provided.";}?>
```

edit_bookings.php:

```
<?php
require 'config.php';
if (isset($_GET['user_id'])) {
    $userId = $_GET['user_id'];
    $result = mysqli_query($conn, "SELECT * FROM bookings WHERE user_id=$userId");
    $user = mysqli_fetch_assoc($result);}
if (isset($_POST['submit'])) {
    $userId = $_POST['user_id']; $startDate = $_POST['start_date'];
    $foodStatus = $_POST['food_status']; $guardianName = $_POST['guardian_name'];
    $relation = $_POST['relation']; $guardianContact = $_POST['guardian_contact'];
    $emergencyContact = $_POST['emergency_contact'];
    $selectedRoom = $_POST['selected_room'];
     if ($selectedRoom != $user['selected_room']) {
        $prevRoom = $user['selected_room'];
        mysqli_query($conn, "UPDATE room SET
            available_seats = available_seats + 1,
            occupied_seats = occupied_seats - 1
            WHERE room_number = '$prevRoom'");
            mysqli_query($conn, "UPDATE room SET
            available_seats = available_seats - 1,
            occupied_seats = occupied_seats + 1
            WHERE room_number = '$selectedRoom'");}
    mysqli_query($conn, "UPDATE bookings SET
    start_date = '$startDate',
```

```
food_status = '$foodStatus',  guardian_name = '$guardianName', relation = '$relation',

guardian_contact = '$guardianContact', emergency_contact = '$emergencyContact',

selected_room = '$selectedRoom' WHERE user_id = '$userId'"); header('Location: bookings.php');}?>

<h2>Edit Booking Details</h2>

<input type="hidden" name="user_id" value="<?php echo $user['user_id']; ?>">

<label for="start_date">New Start Date:</label>

<input type="date" id="start_date" name="start_date" required value="<?php echo $user['start_date']; ?>">

<label for="food_status">Food Status:</label>

<select id="food_status" name="food_status" required>

  <option value="1" <?php echo ($user['food_status'] == 1) ? 'selected' : ''; ?>>Required</option>

  <option value="0" <?php echo ($user['food_status'] == 0) ? 'selected' : ''; ?>>Not Required</option>

</select>

<label for="guardian_name">Guardian Name:</label>

<input type="text" id="guardian_name" name="guardian_name" required value="<?php echo $user['guardian_name']; ?>">

<label for="relation">Relation:</label>

<input type="text" id="relation" name="relation" required value="<?php echo $user['relation']; ?>">

<label for="guardian_contact">Guardian Contact:</label>

<input type="text" id="guardian_contact" name="guardian_contact" required value="<?php echo $user['guardian_contact']; ?>">

<label for="emergency_contact">Emergency Contact:</label>

<input type="text" id="emergency_contact" name="emergency_contact" required value="<?php echo $user['emergency_contact']; ?>">

<label for="selected_room">Selected Room:</label>

<input type="text" id="selected_room" name="selected_room" required value="<?php echo $user['selected_room']; ?>">

<button type="submit" name="submit">Update Booking</button>

</form></div>
```

edit_room.php:

```
<?php
require "config.php";
if (!isset($_SESSION["admin_login"])) {
    header("Location: adminlogin.php");
    exit();}
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $roomNumber = $_POST["room_number"];
    $availableSeats = $_POST["available_seats"];
    $occupiedSeats = $_POST["occupied_seats"];
    $totalSeats = $_POST["total_seats"];
    if ($_FILES["image"]["name"]) {
        $targetDirectory = "uploads/";
        $imageName = uniqid() . "_" . basename($_FILES["image"]["name"]);
        $targetFile = $targetDirectory . $imageName;
        move_uploaded_file($_FILES["image"]["tmp_name"], $targetFile);
        $updateImageQuery = "UPDATE room SET image_path = '$targetFile' WHERE
room_number = '$roomNumber'";
        mysqli_query($conn, $updateImageQuery);}
    $updateQuery = "UPDATE room SET
                available_seats = '$availableSeats',
                occupied_seats = '$occupiedSeats',
                total_seats = '$totalSeats'
                WHERE room_number = '$roomNumber'";
    mysqli_query($conn, $updateQuery);
    header("Location: addrooms.php");
    exit();}
if (isset($_GET['id'])) {
    $roomId = $_GET['id'];
    $selectQuery = "SELECT * FROM room WHERE room_number = '$roomId'";
```

```
    $result = mysqli_query($conn, $selectQuery);
    if ($result && $row = mysqli_fetch_assoc($result)) {
        $roomNumber = $row['room_number']; $availableSeats = $row['available_seats'];
        $occupiedSeats = $row['occupied_seats']; $totalSeats = $row['total_seats'];
        $imagePath = $row['image_path'];
    } else {header("Location: addrooms.php");exit();}
} else {header("Location: addrooms.php");exit();}?>
        <form action="" method="post" enctype="multipart/form-data">
            Room Number: <input type="text" name="room_number" value="<?php echo
$roomNumber; ?>" readonly><br>
            Available Seats: <input type="number" name="available_seats" value="<?php echo
$availableSeats; ?>" required><br>
            Occupied Seats: <input type="number" name="occupied_seats" value="<?php echo
$occupiedSeats; ?>" required><br>
            Total Seats: <input type="number" name="total_seats" value="<?php echo
$totalSeats; ?>" required><br>
            Current Image: <img src="<?php echo $imagePath; ?>" alt="Current Image"
style="width: 150px; height: 100px;"><br>
            New Image: <input type="file" name="image" accept="image/*"><br>
            <input type="submit" value="Update Room"></form></div></body></html>
```

edit-admin.php:

```
<?php
require 'config.php';
if (isset($_GET['id'])) {
    $id = $_GET['id'];
    $result = mysqli_query($conn, "SELECT * FROM admins WHERE id=$id");
    $user = mysqli_fetch_assoc($result);}
if (isset($_POST['submit'])) {
    $id = $_POST['id']; $name = $_POST['name']; $email = $_POST['email'];
    $password = $_POST['password'];
```

```
    mysqli_query($conn, "UPDATE admins SET name='$name', email='$email',
password='$password' WHERE id=$id");

    header('Location: admin_dashboard.php');}?>

<link rel="stylesheet" href="../Styles/login.css">

<div class="container">
```

edit-user.php:

```
<?php

require 'config.php';

if (isset($_GET['id'])) {

    $id = $_GET['id'];

    $result = mysqli_query($conn, "SELECT * FROM users WHERE id=$id");

    $user = mysqli_fetch_assoc($result);}

if (isset($_POST['submit'])) {

    $id = $_POST['id']; $name = $_POST['name']; $father_name = $_POST['father_name'];

    $dob = $_POST['dob']; $address = $_POST['address'];

    $phone_number = $_POST['phone_number']; $email = $_POST['email'];

    $password = $_POST['password'];

    mysqli_query($conn, "UPDATE users SET name='$name', father_name='$father_name',
dob='$dob', address='$address', phone_number='$phone_number', email='$email',
password='$password' WHERE id=$id");

    header('Location: admin_manageuser.php');}?>
```

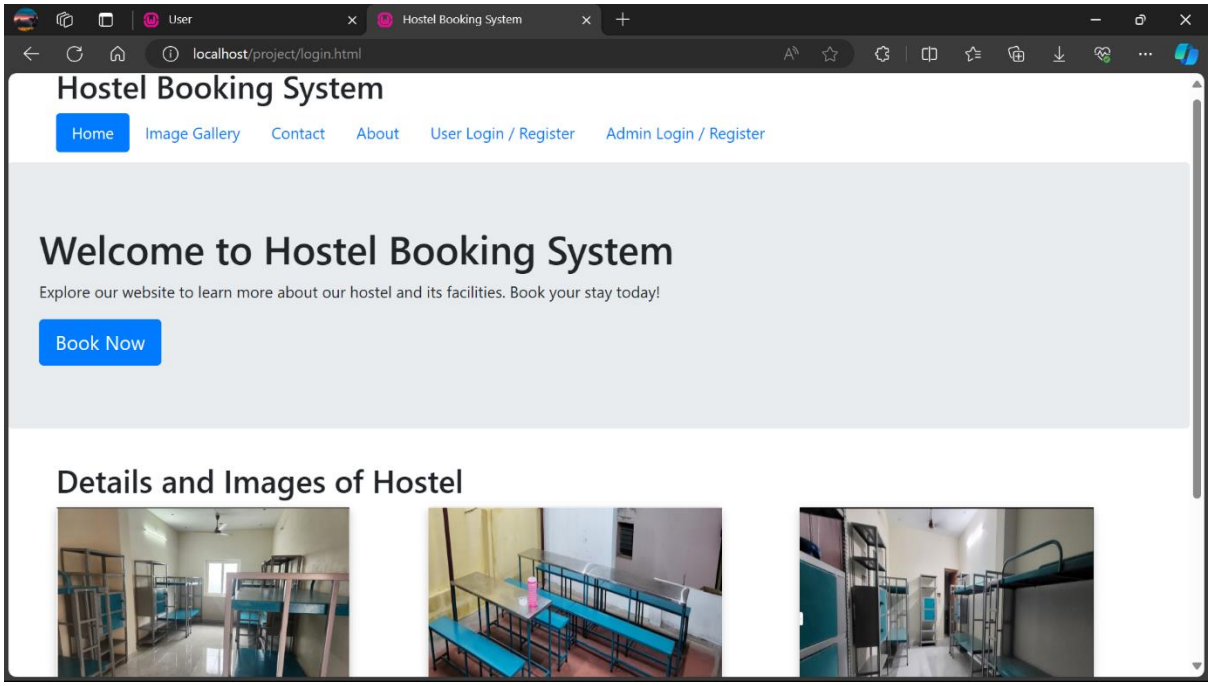
Newroom.php:

```
<?php

require "config.php";

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $roomNumber = $_POST["room_number"];

    $availableSeats = $_POST["available_seats"];

    $occupiedSeats = $_POST["occupied_seats"];

    $totalSeats = $_POST["total_seats"];
```

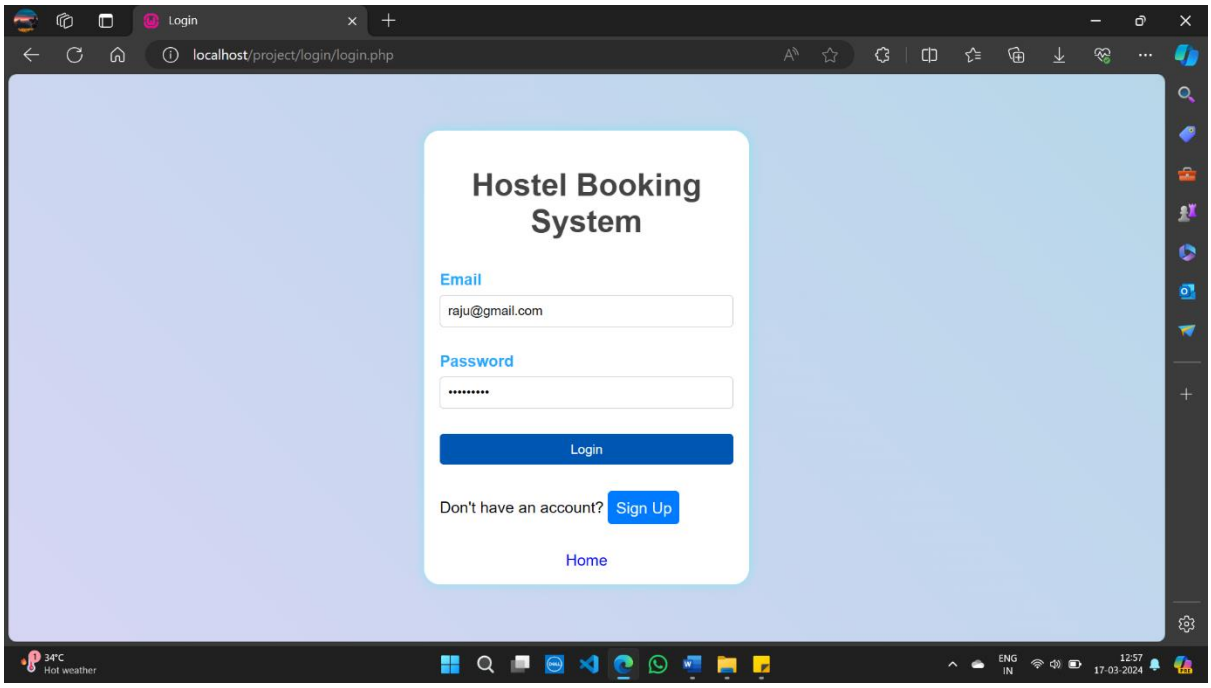
```
    if ($_FILES["image"]["name"]) {
        $targetDirectory = "uploads/";
        $imageName = uniqid() . "_" . basename($_FILES["image"]["name"]);
        $targetFile = $targetDirectory . $imageName;
        move_uploaded_file($_FILES["image"]["tmp_name"], $targetFile);
    } else $targetFile = "uploads/default_image.png";}
    $insertQuery = "INSERT INTO room (room_number, available_seats, occupied_seats,
total_seats, image_path)VALUES ('$roomNumber', '$availableSeats', '$occupiedSeats',
'$totalSeats', '$targetFile')";
    $insertResult = mysqli_query($conn, $insertQuery);
    if ($insertResult) {echo "<script>alert('Room added successfully.'); window.location.href =
'addrooms.php'; </script>";exit();
    } else {echo "Error adding room: " . mysqli_error($conn);}}?>
        <form action="" method="post" enctype="multipart/form-data">
            Room Number: <input type="text" name="room_number" required><br>
            Available Seats: <input type="number" name="available_seats" required><br>
            Occupied Seats: <input type="number" name="occupied_seats" required><br>
            Total Seats: <input type="number" name="total_seats" required><br>
            Image: <input type="file" name="image" accept="image/*"><br>
            <input type="submit" value="Add Room">
        </form></div></body></html>
```

8.2 Output screens:

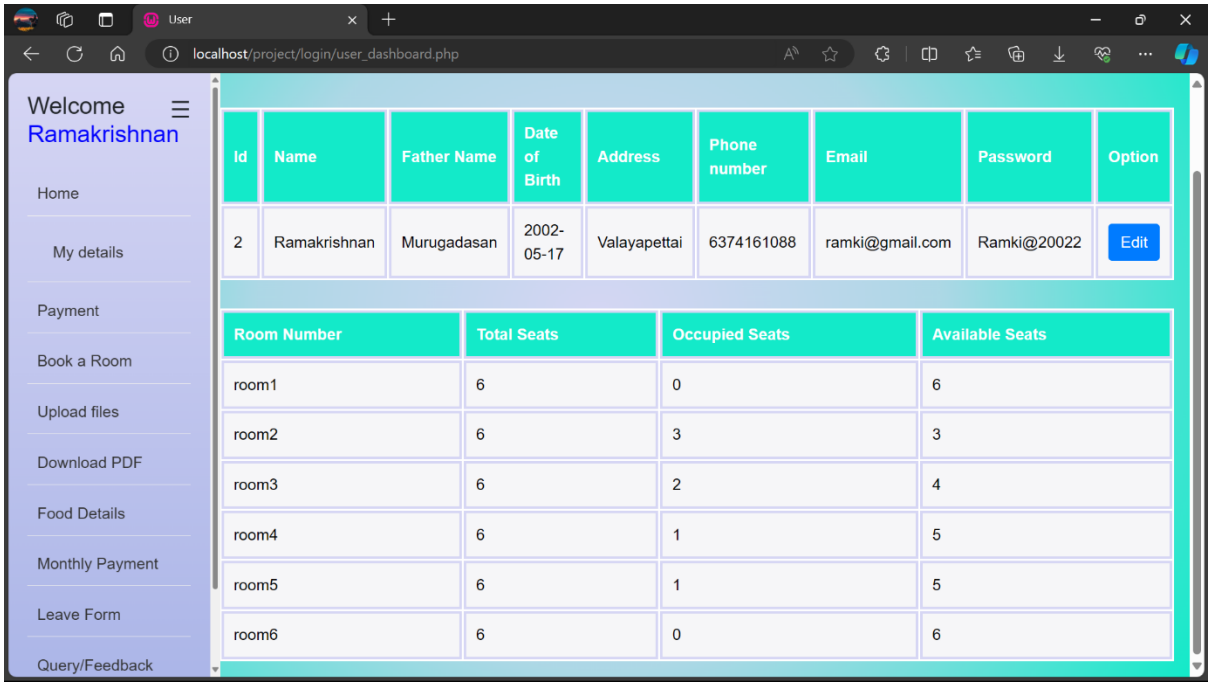
USER MODULE



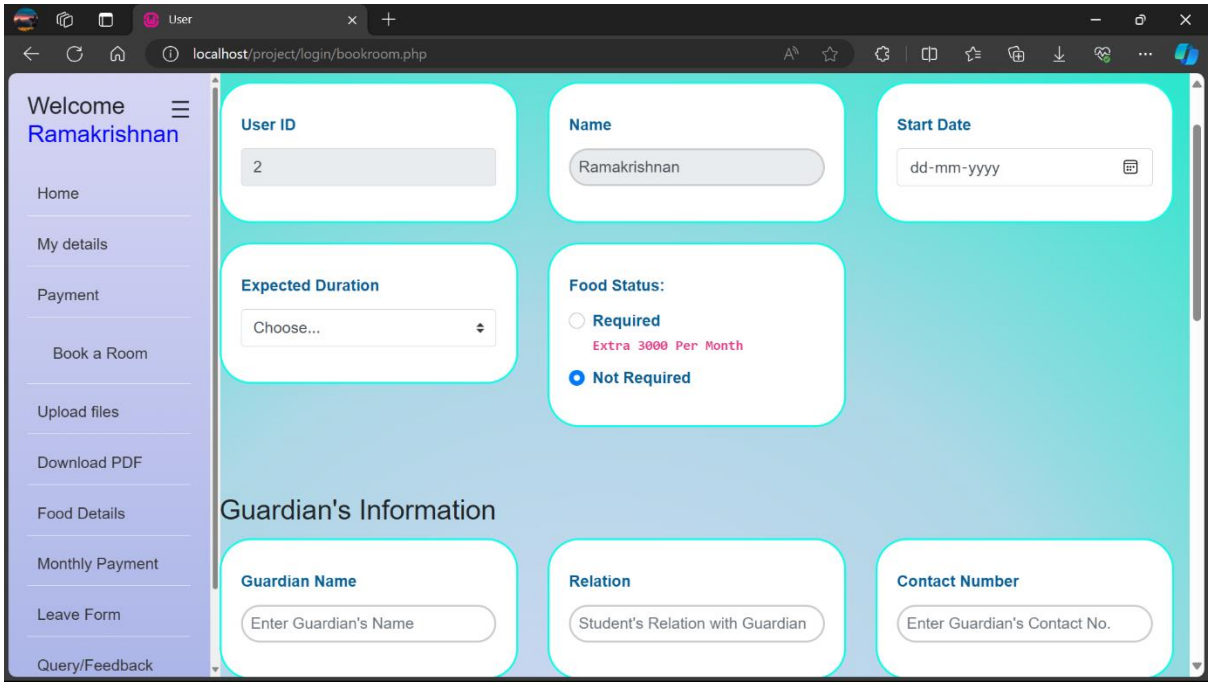
Figrue 8.2.1 index.html



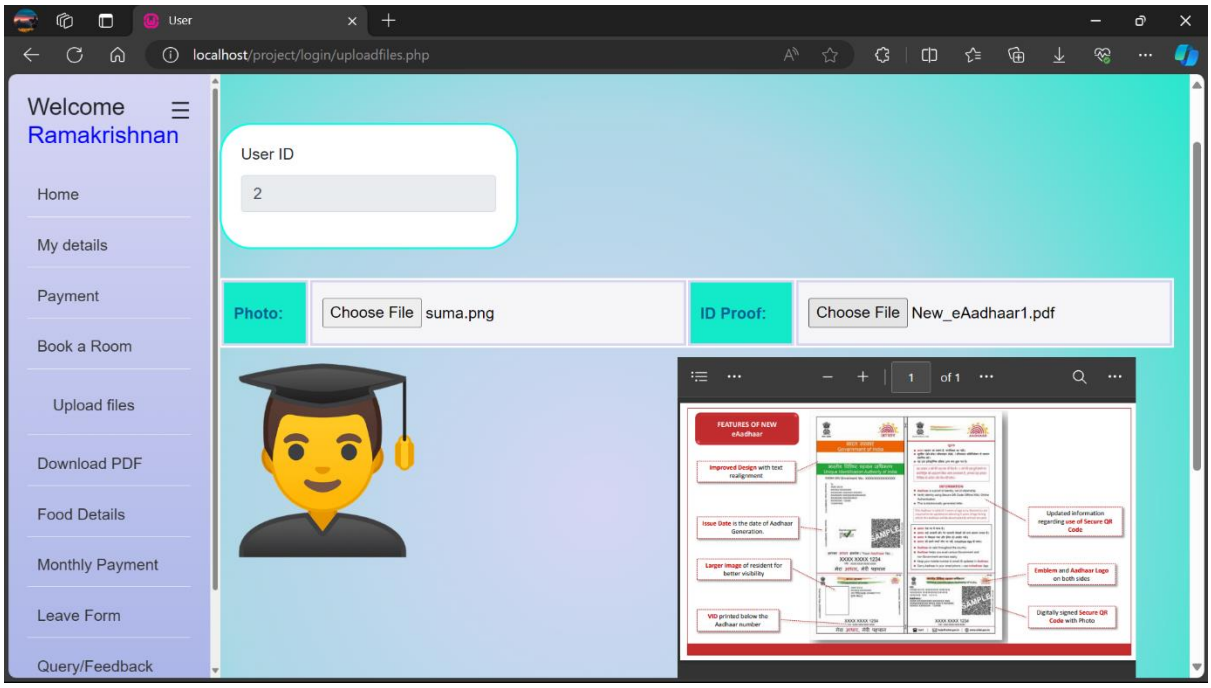
Figrue 8.2.2 login.php



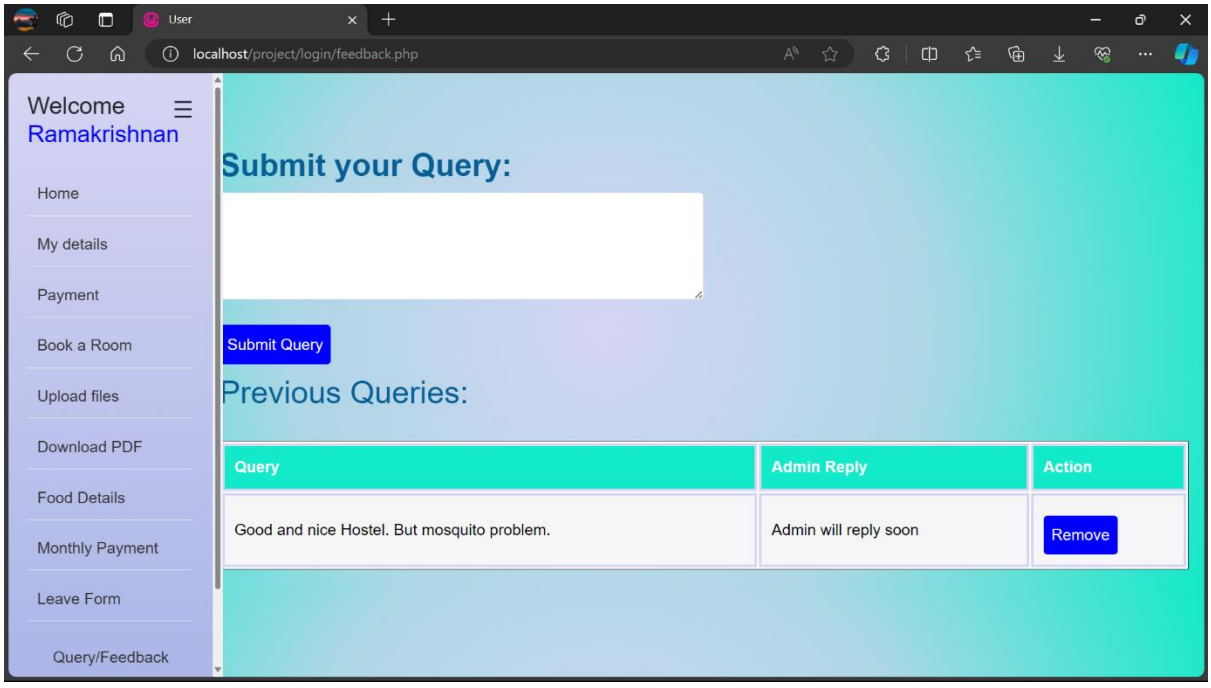
Figrue 8.2.3 user_dashboard.php



Figrue 8.2.4 bookroom.php

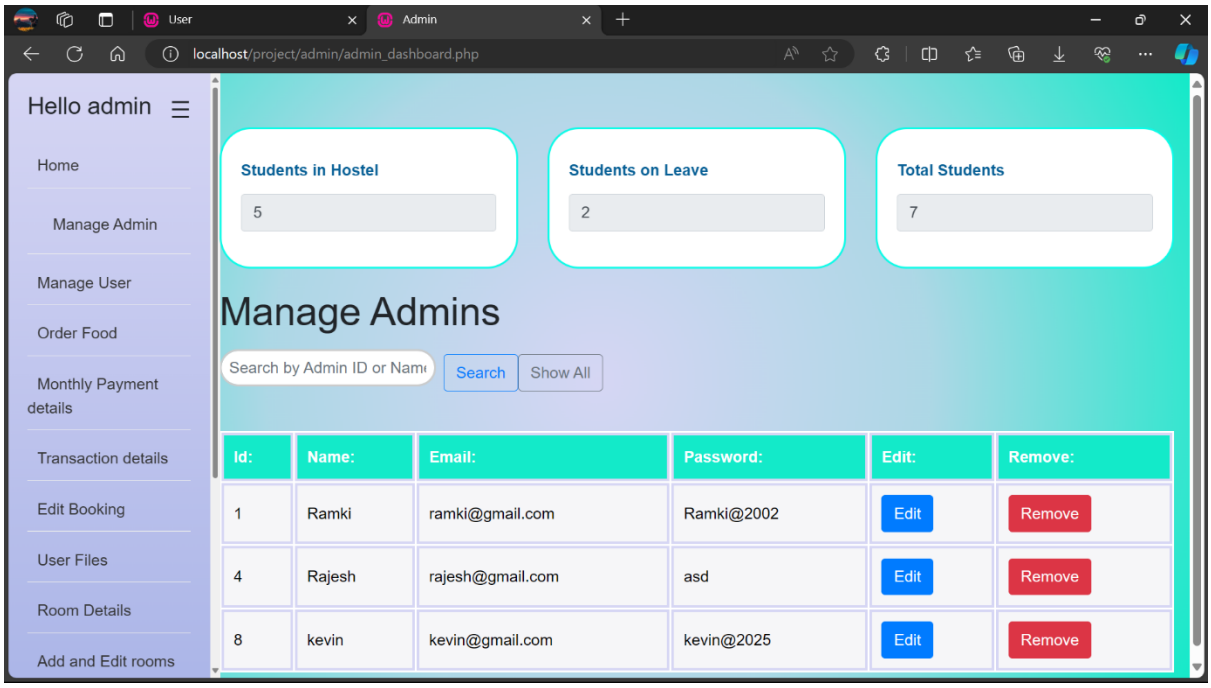


Figrue 8.2.5 uploadfiles.php

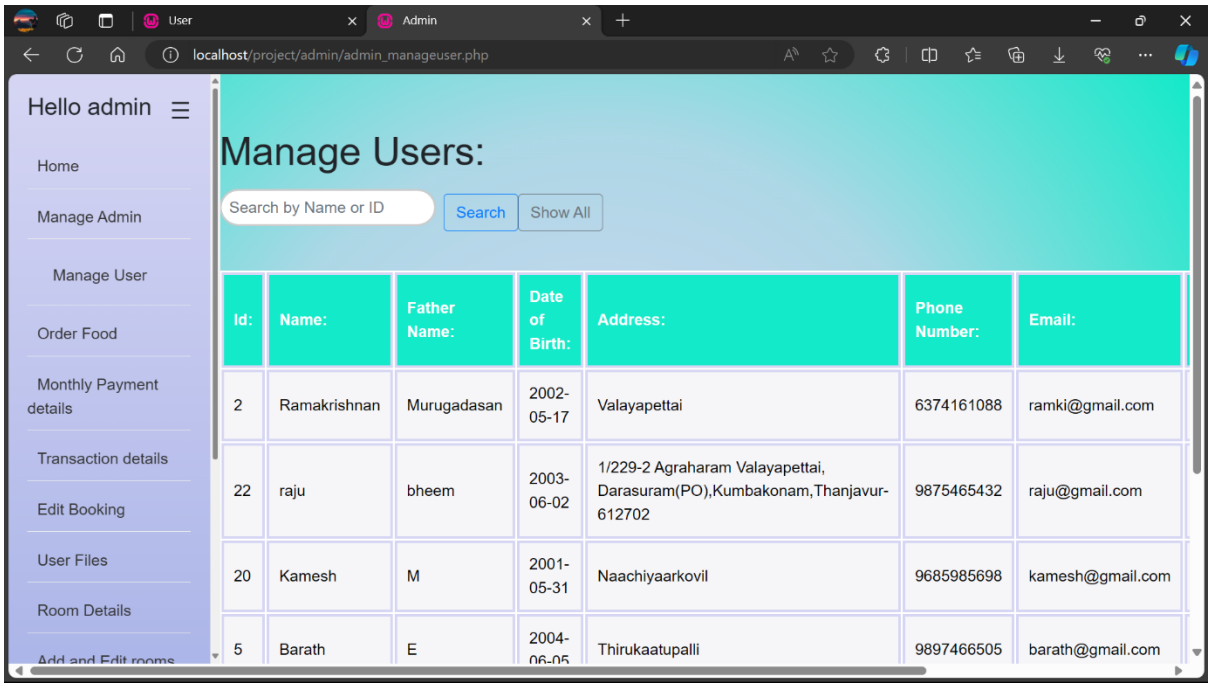


Figrue 8.2.6 userquery.php

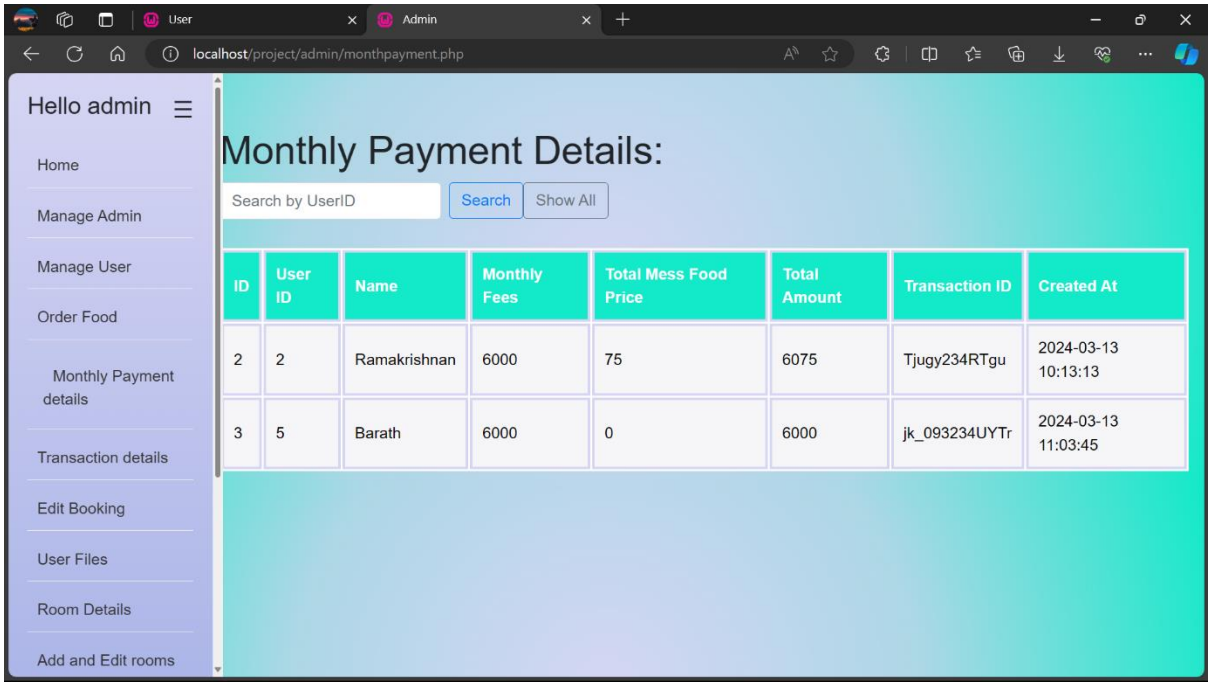
ADMIN MODULE:



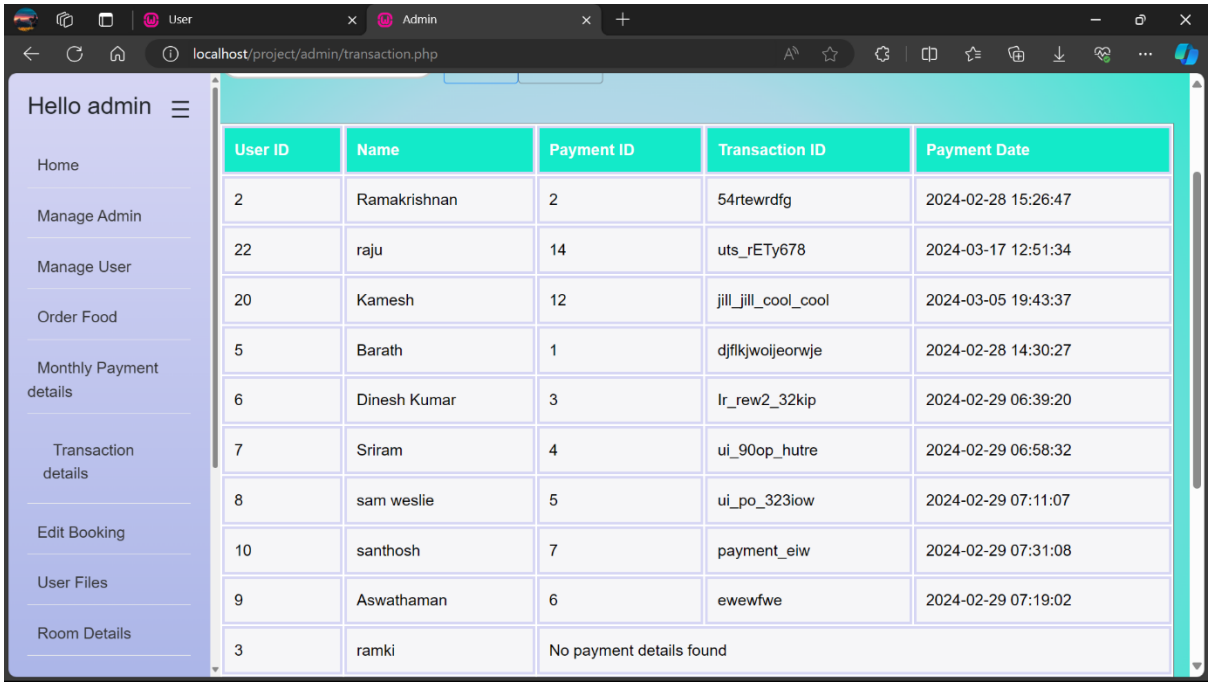
Figrue 8.2.7 admin_dashboard.php



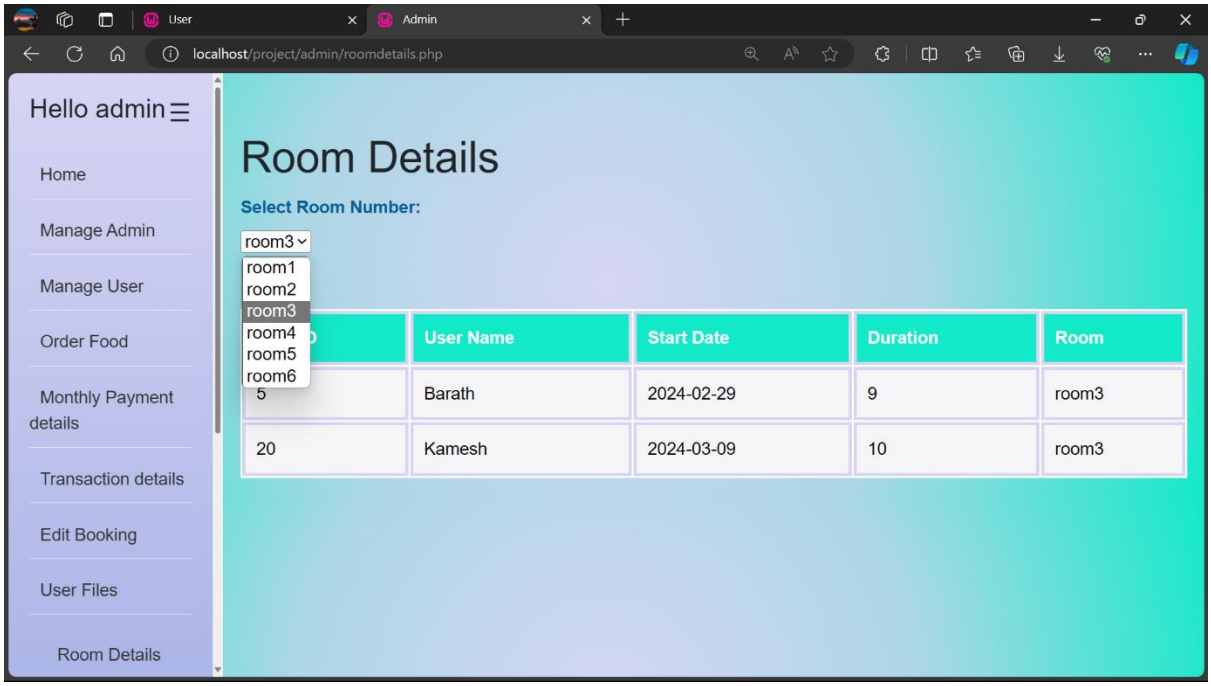
Figrue 8.2.8 admin_manageuser.php



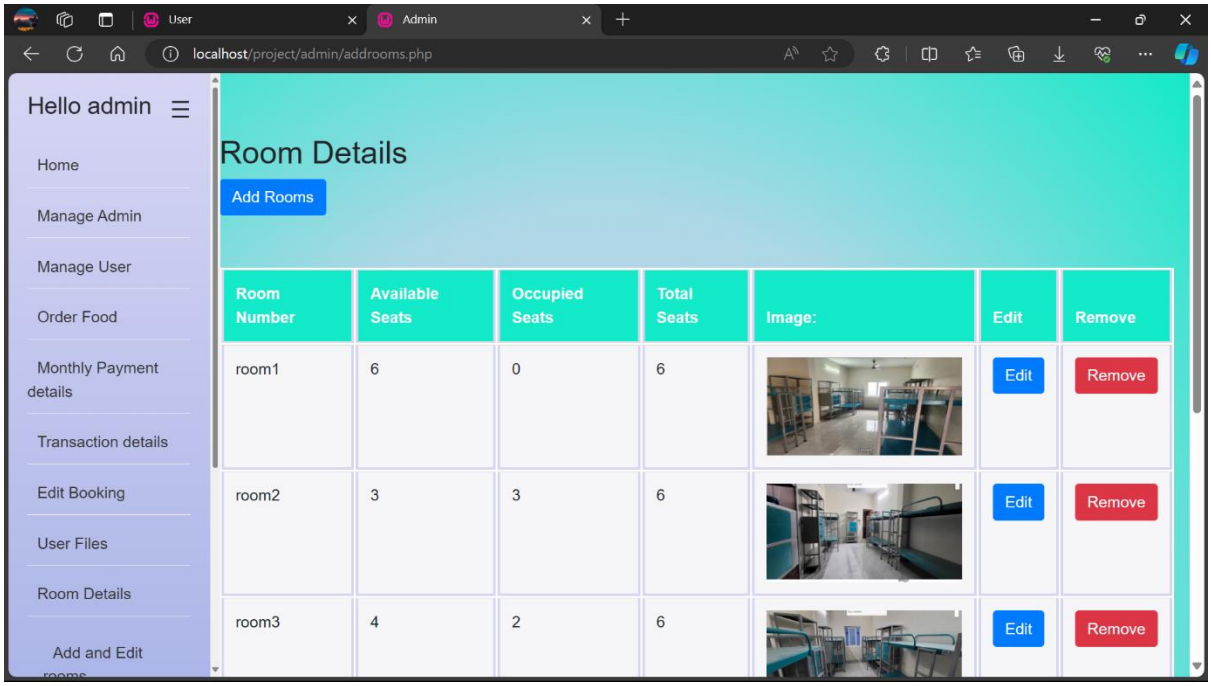
Figrue 8.2.9 monthpayment.php



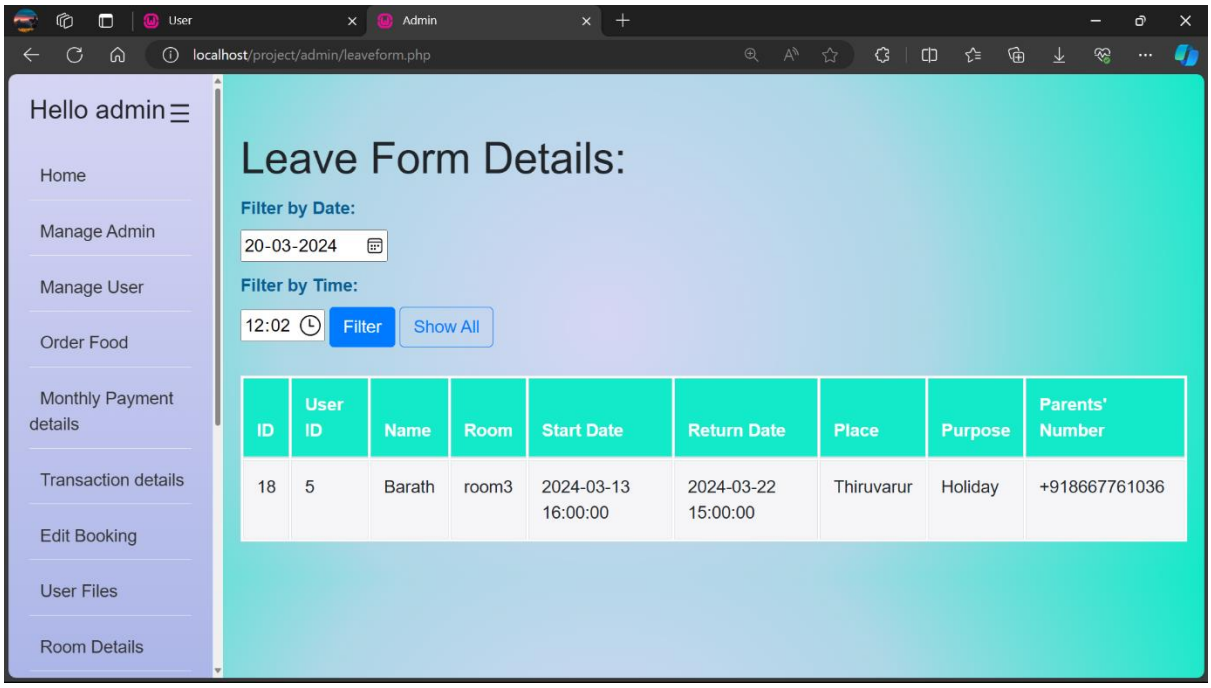
Figrue 8.2.10 transaction.php



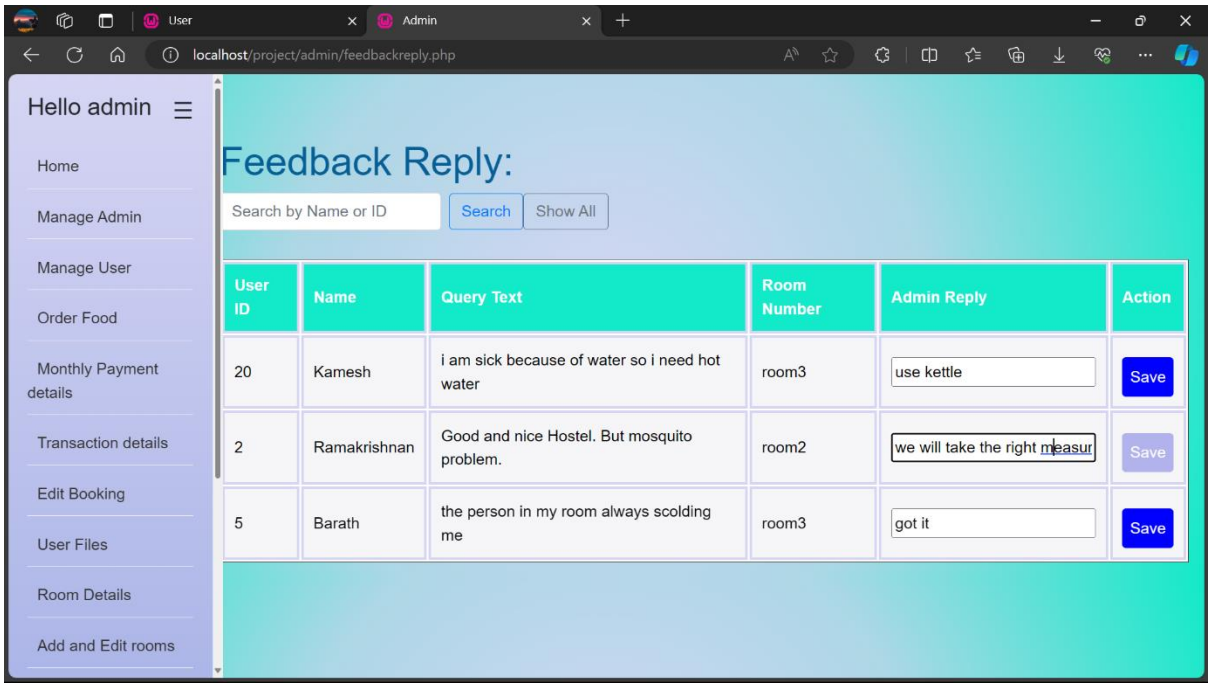
Figrue 8.2.11 roomdetails.php



Figrue 8.2.12 addrooms.php



Figrue 8.2.13 leaveform.php



Figrue 8.2.14 feedbackreply.php