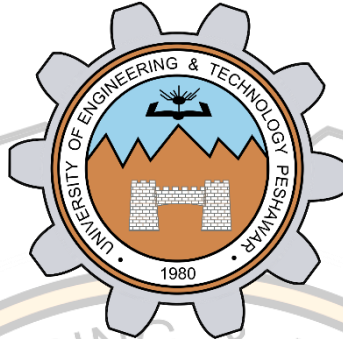


**UNIVERSITY OF ENGINEERING AND TECHNOLOGY  
PESHAWAR (JALOZAI CAMPUS)**

**Department Of Computer Science &IT**



**HOSTEL MANAGEMENT SYSTEM**

**By**

**MUHAMMAD AHMAD KHAN (23JZBCS0238)**

**ALEENA KHAN (23JZBCS0229)**

**FAIZAN ULLAH WAZIR (23JZBCS0242)**

**BACHELOR**

**IN**

**COMPUTER SCIENCE**

**2023-2027**

# HOSTEL MANAGMENT SYSTEM

By

MUHAMMAD AHMAD KHAN (23JZBCS0238)

ALEENA KHAN (23JZBCS0229)

FAIZAN ULLAH WAZIR (23JZBCS042)

## PROJECT

Presented to the Department of Computer Science & IT  
UNIVERSITY OF ENGINEERING AND TECHNOLOGY PESHAWAR (JALOZAI  
CAMPUS)

In partial fulfillment of  
The requirement for the semester project of

## BACHELOR IN COMPUTER SCIENCE

**Project Documentation Committee:**

**External Examiner**

\_\_\_\_\_

**Internal Supervisor**

Mam Abnash Zaman

\_\_\_\_\_

**HOD**

Dr. Yousaf Khalil

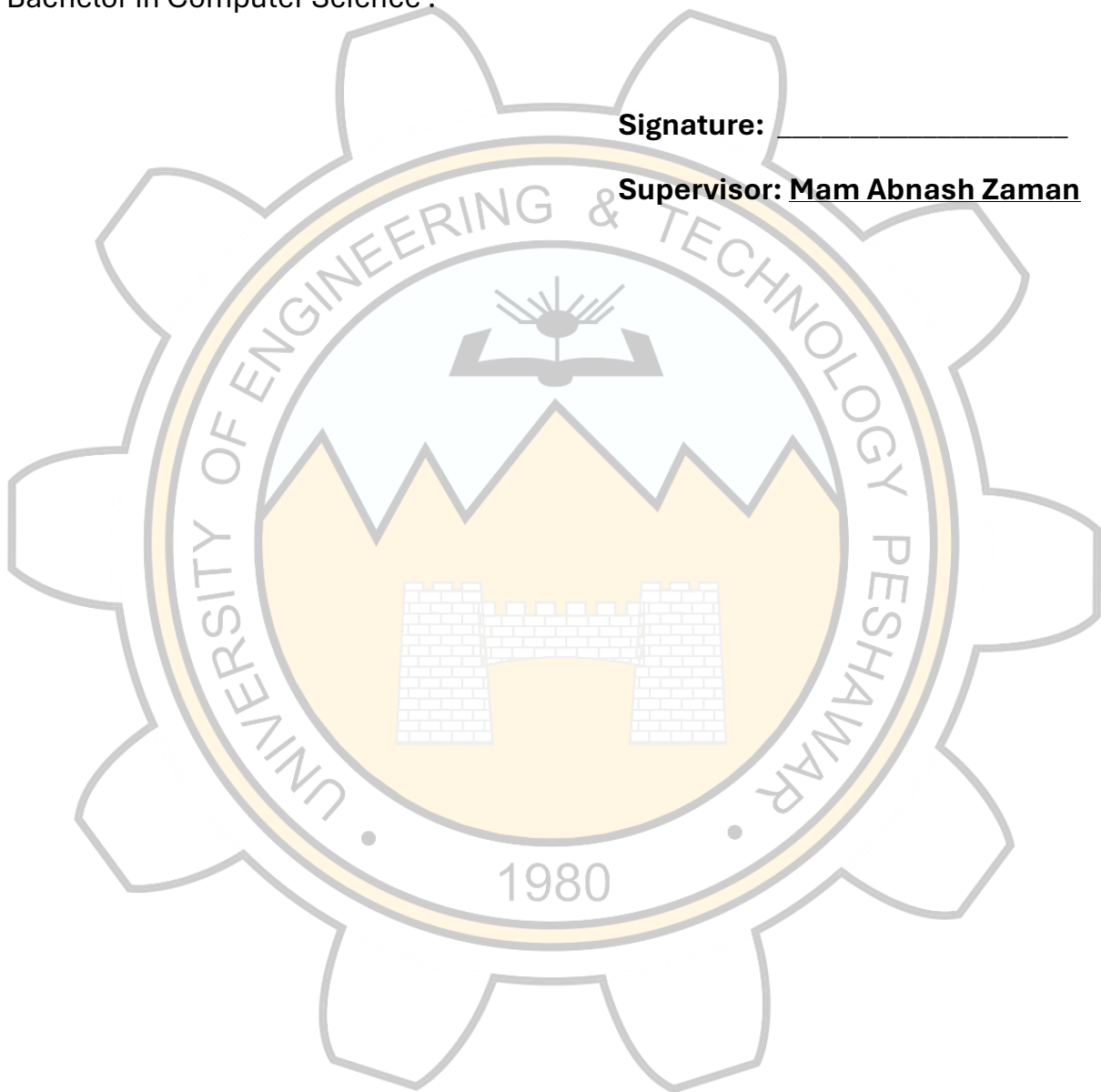
\_\_\_\_\_

## PROJECT COMPLETION CERTIFICATE

It is certified that the work contained in the project entitled topic **“HOSTEL MANAGMENT”** has been carried out and completed by **“MUHAMMAD AHMAD KHAN, ALEENA KHAN , FAIZAN ULLAH WAZIR”** under my supervision of his/her Bachelor in Computer Science .

Signature: \_\_\_\_\_

Supervisor: Mam Abnash Zaman



## DECLARATION OF ORINALITY

I hereby declare that the work contained in this project and the intellectual contents of this report are product of my own work, this report as previously published in any form nor does it contain any verbatim of the published resources which could be treated as infringement of the international copyright law, except where due reference is made in the text of this project.

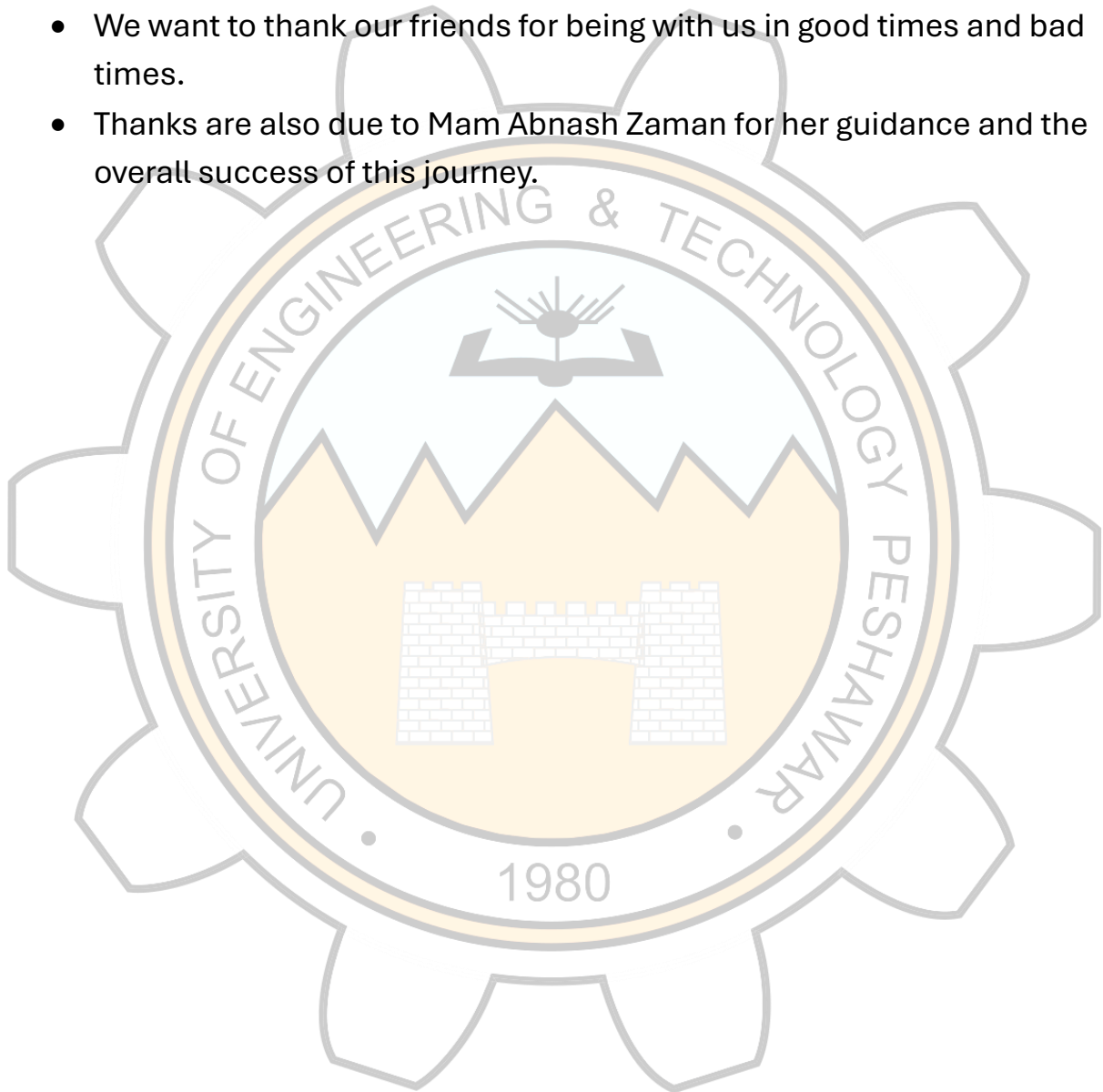
I also declare that I do understand the terms 'copyright' and 'plagiarism' and in case of any copyright violation or plagiarism found in this work, I will be held fully responsible of the consequences of any such violation.

### Signatures and Details:

Name	Roll Number	Signature
Muhammad Ahmad Khan	23jzbc0238	_____
Aleena Khan	23jzbc0229	_____
Faizan Ullah Wazir	23jzbc0242	_____

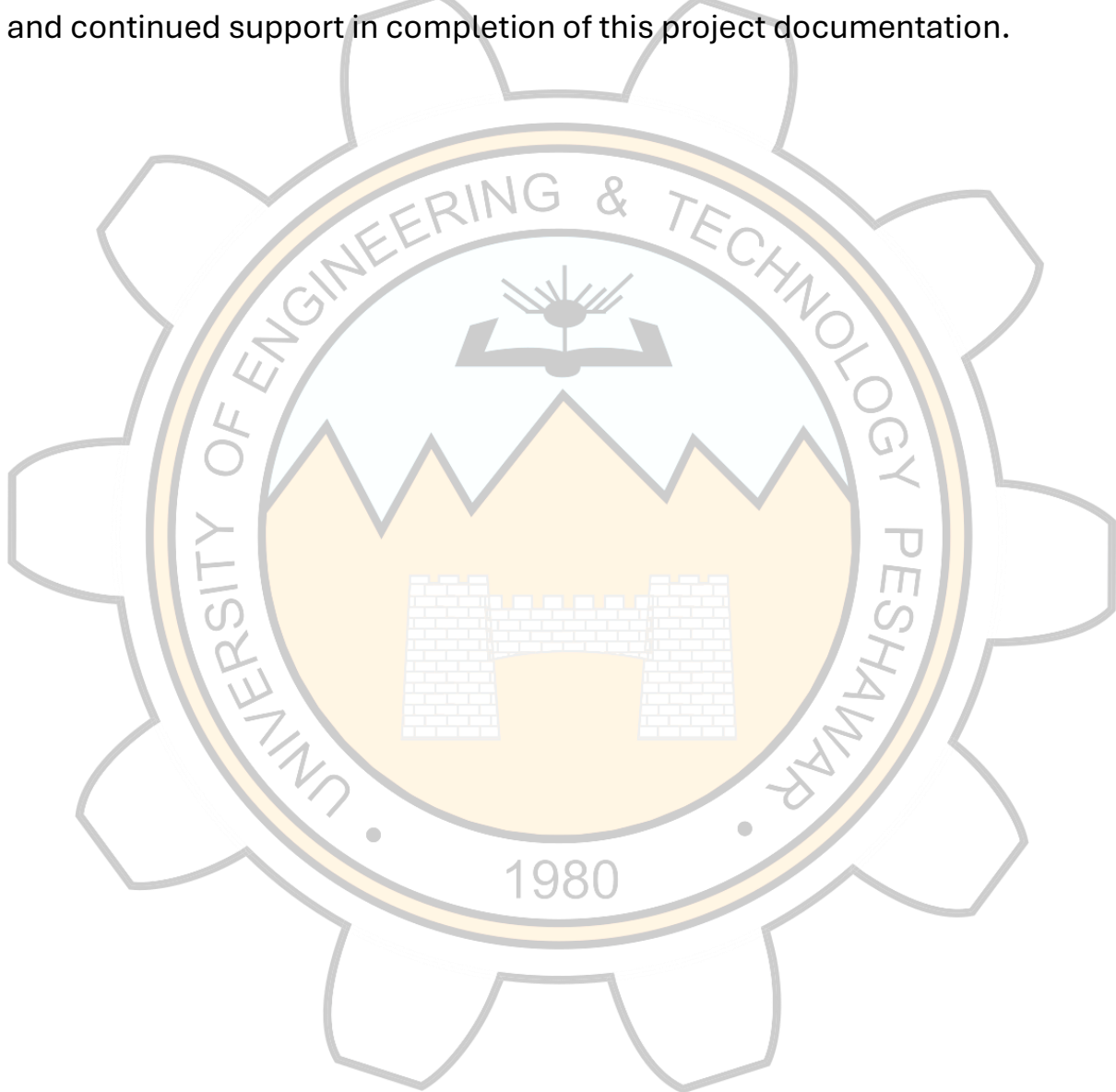
## DEDICATION

- We wouldn't be here without the love, help and support our families have given us.
- We want to thank our friends for being with us in good times and bad times.
- Thanks are also due to Mam Abnash Zaman for her guidance and the overall success of this journey.



## ACKNOWLEDEgements

Special thanks to **Mam Abnash Zaman**, my supervisor for their guidance and continued support in completion of this project documentation.



## ABSTRACT

The Hostel Management System is a database-driven solution designed to streamline the administrative operations of student hostels. This project focuses on developing a centralized system to manage room allocations, student records, fee payments, and hostel resources efficiently. Traditional hostel management often involves manual record-keeping, which is prone to errors and delays. By implementing a structured relational database, this system ensures data consistency, reduces redundancy, and enhances the overall reliability of hostel operations. The project was developed using [database system, e.g., MySQL], and includes key functionalities such as student registration, room availability tracking, fee management, and report generation. The system aims to improve efficiency, reduce paperwork, and provide an organized platform for hostel administrators.



## TABLE OF CONTENTS

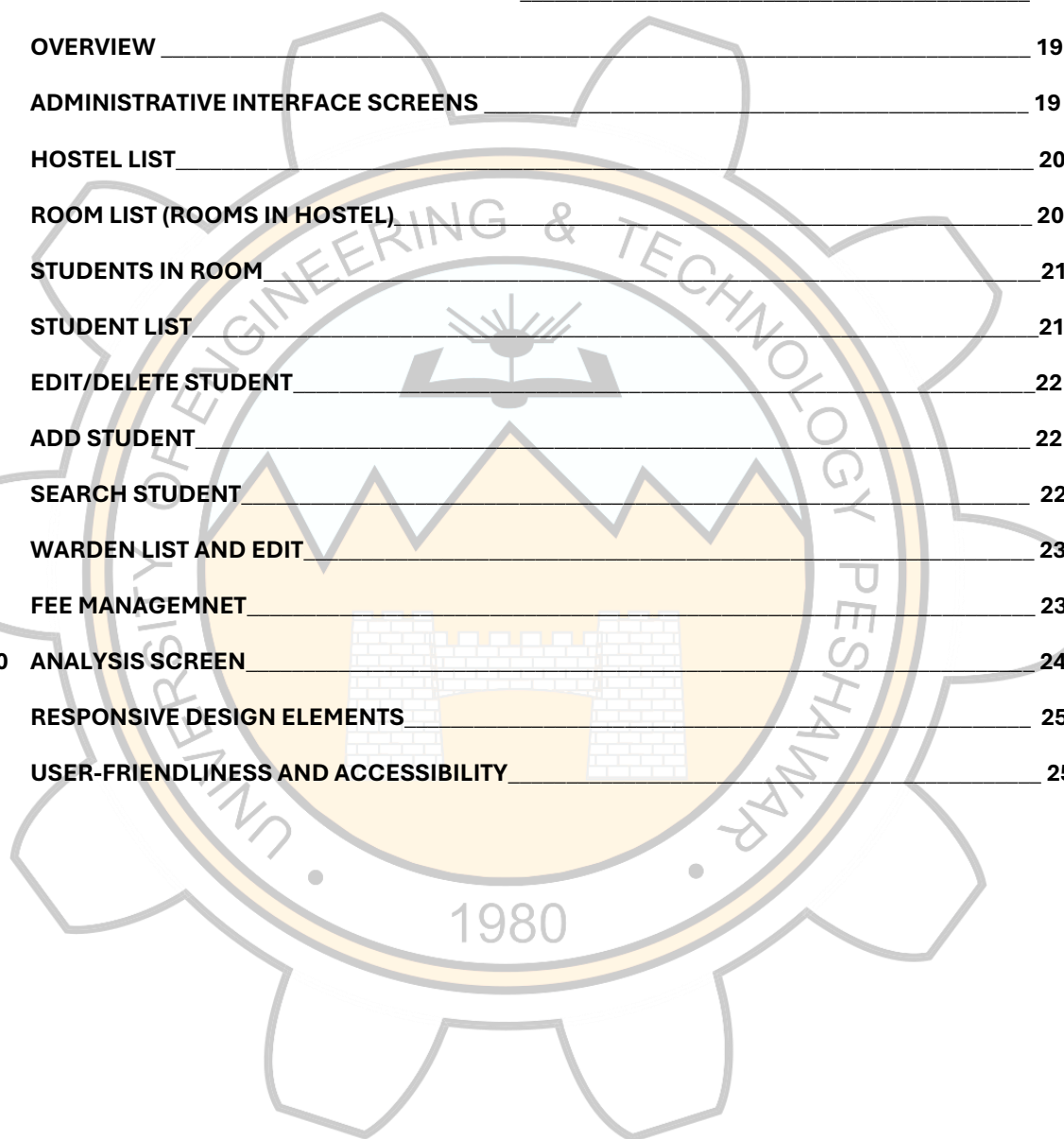
DEDICATION	II
ACKNOWLEDGEMENTS	III
ABSTRACT	IV
CHAPTER NO. 1	
1 INTRODUCTION	1
1.1 SCOPE	1
1.2 ACRONYMS, AND ABBREVIATIONS	2
1.3 ASSUMPTIONS AND DEPENDENCIES	2
1.4 OBJECTIVES	3
1.5 PROJECT BRIEF	3
1.6 PROJECT STRUCTURE	4
CHAPTER NO. 2	
2 LITERATURE REVIEW	6
2.1 BACKGROUND	6
2.2 DEFINITIONS & TERMS	6
2.3 SOFTWARE & TOOLS	7
CHAPTER NO. 3	
3 ANALYSIS AND DESIGN	9
3.1 SOFTWARE REQUIREMENT SPECIFICATIONS	9
3.1.1 PROBLEM STATEMENT	9
3.1.2 CONSTRAINTS AND LIMITATION	9
3.1.3 FUNCTIONAL REQUIREMENTS	10
3.1.4 NON-FUNCTIONAL REQUIREMENTS	10
CHAPTER NO. 4	
4 DATABASE ANALYSIS AND DESIGN	12
4.1 DATABASE OVERVIEW	12
4.1.1 TOOLS USED	12



4.1.2	DATA DICTIONARY	12
4.2	ENTITY-RELATIONSHIP DIAGRAM	14
4.3	NORMALIZATION	15
4.4	DATABASE SNAPSHOTS	15

## CHAPTER NO. 5

5	GRAPHICAL USER INTERFACE SNAPSHOTS	19
5.1	OVERVIEW	19
5.2	ADMINISTRATIVE INTERFACE SCREENS	19
5.2.1	HOSTEL LIST	20
5.2.2	ROOM LIST (ROOMS IN HOSTEL)	20
5.2.3	STUDENTS IN ROOM	21
5.2.4	STUDENT LIST	21
5.2.5	EDIT/DELETE STUDENT	22
5.2.6	ADD STUDENT	22
5.2.7	SEARCH STUDENT	22
5.2.8	WARDEN LIST AND EDIT	23
5.2.9	FEE MANAGEMNET	23
5.2.10	ANALYSIS SCREEN	24
5.3	RESPONSIVE DESIGN ELEMENTS	25
5.4	USER-FRIENDLINESS AND ACCESSIBILITY	25



# 1 INTRODUCTION

The primary goal of the **Hostel Management** database is to organize and manage all data related to students, staff, rooms, fees, and complaints in a structured and relational format that supports efficient data retrieval, updating, and reporting.

This system employs a **relational database model** to store data in the form of interlinked tables. Each table is carefully designed with defined attributes, primary keys, and foreign key relationships to maintain referential integrity. For instance, the *Students* table holds personal and academic details, the *Rooms* table contains information on available and occupied rooms, and the *Payments* table tracks fee transactions and dues. These tables are interconnected in such a way that data consistency is maintained across the system.

## 1.1 SCOPE

The project targets hostel administrators, staff, and students. It is designed to be scalable for hostels of different sizes. The system will have functionalities such as login authentication, student registration, room assignment, fee payment history, and report generation.

The scope of this project covers both administrative and student-facing functionalities. Administrators will be able to register students, assign rooms, manage room availability, monitor payment records, and respond to student complaints. Students, on the other hand, can log into the system to view their profile, room details, payment status, and submit maintenance requests or complaints.

## 1.2 Acronyms, and Abbreviations

Acronym	Description
	Hostel Management System
DBMS	Database Management System
SQL	Structured Query Language
MySQL	A popular open-source relational database management system
API	Application Programming Interface
PHP	Hypertext Preprocessor (a server-side scripting language)
UI	User Interface

Acronym	Description
UX	User Experience
IDE	Integrated Development Environment
CRUD	Create, Read, Update, Delete
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
MVC	Model View Controller
OOP	Object-Oriented Programming
SDK	Software Development Kit
DB	Database
REST	Representational State Transfer
IUB	The Islamia University of Bahawalpur
DCSIT	Department of Computer Science & Information Technology

### 1.3 ASSUMPTIONS AND DEPENDENCIES

In the development of the Hostel Management System (HMS), the following assumptions and dependencies were taken into consideration:

#### Assumptions:

1. The users (admin and students) will have basic knowledge of how to interact with digital systems via mobile or desktop applications.
2. All users will have access to the internet to use the system effectively.
3. The database server (MySQL) will be operational and connected at all times for seamless functionality.
4. PHP APIs will handle all communication between the frontend (Flutter) and backend (MySQL).
5. Hostel administration will be responsible for maintaining up-to-date student and room allocation data.
6. The system will be used within a single organization (i.e., a university hostel) and not across multiple hostels or campuses.

#### Dependencies:

1. **Flutter Framework:** Used for developing the frontend interface of the mobile application.
2. **MySQL Database:** Stores all system data including student records, room details, fee payments, complaints, and other administrative information.
3. **PHP APIs:** Serve as the bridge between the frontend and backend, handling data transactions.
4. **phpMyAdmin:** Used for managing the MySQL database during development.

5. **Server Hosting:** Required to host APIs and the MySQL database for deployment.
6. **Android SDK:** Required for building and running the Flutter application on Android devices.

.NET for windows application

7. **Development Environment:** A stable IDE such as Android Studio or Visual Studio Code for Flutter development.

## 1.4 OBJECTIVES

The primary objective of this Hostel Management System (HMS) is to design and develop a user-friendly, efficient, and scalable system that streamlines the day-to-day operations of managing a hostel. The system is aimed at both hostel administrators and students, enabling them to interact with hostel facilities digitally and efficiently.

The main objectives of the project are:

1. **To automate hostel administrative tasks** such as student registration, room allocation, fee management, and complaint tracking.
2. **To provide an interactive user interface** for both students and hostel staff using a mobile-friendly application built with Flutter.
3. **To develop a centralized and secure database** using MySQL that stores all hostel-related data including student details, rooms, payments, and complaints.
4. **To integrate the frontend and backend seamlessly** using PHP APIs for efficient data transactions and system communication.
5. **To improve transparency and accuracy** in hostel operations by minimizing manual data entry and paperwork.
6. **To enhance the overall hostel management process** by reducing administrative workload and providing real-time access to relevant data.
7. **To ensure role-based access** so that only authorized users can perform specific actions (e.g., room allocation by admin, complaint submission by students).
8. **To support scalability and future enhancements**, such as biometric check-ins, notifications, or online payment modules.

## 1.5 PROJECT BRIEF

The **Hostel Management System** (HMS) is designed to facilitate the digital management of hostel activities such as student admissions, room assignments, fee tracking, and complaint handling. The system allows hostel administrators to efficiently

manage hostel operations, while providing students with an easy way to interact with the hostel services.

This project was developed using **Flutter** for the front-end mobile application, providing a responsive and modern user interface suitable for Android and iOS platforms. For the backend, **MySQL** was used to store all the system's data including student details, room availability, fee records, and complaints. The communication between the frontend and backend is handled through **PHP APIs**, which were developed using **MySQL Admin**.

The system is designed with a modular architecture that ensures flexibility and scalability. Each component of the system is capable of being updated or expanded without affecting the others, making it ideal for future enhancements such as biometric access control, SMS/email notifications, or online fee payment gateways.

The implementation of this project significantly reduces manual workload, ensures accuracy of data, and promotes a structured and paperless environment within the hostel management workflow.

## 1.6 PROJECT STRUCTURE

This documentation is structured in a way that clearly explains each phase of the **Hostel Management System** project, from conceptualization to implementation. The structure ensures that all necessary technical and academic aspects are covered in a well-organized manner. The chapters are arranged as follows:

- **Chapter 1: Introduction**  
This chapter introduces the project, outlines its scope, objectives, assumptions, dependencies, and software process model used. It also includes a brief overview of the project including required software and hardware.
- **Chapter 2: Literature Review**  
It provides background research, definitions, and understanding of relevant concepts and tools related to hostel management systems. Any supporting studies, related systems, or research work are discussed here.
- **Chapter 3: Analysis and Design**  
This is the core part of the project that discusses the software requirement specifications, both functional and non-functional. It also includes UML diagrams such as Use Case, Activity, Sequence, State-Transition, Class, Component, and Deployment Diagrams that explain the system behavior and structure.
- **Chapter 4: Database Analysis and Design**  
This chapter focuses on the database part of the system. It includes the data dictionary, Entity-Relationship diagrams, Data Flow Diagrams, and normalization process. It also presents database implementation snapshots using MySQL.

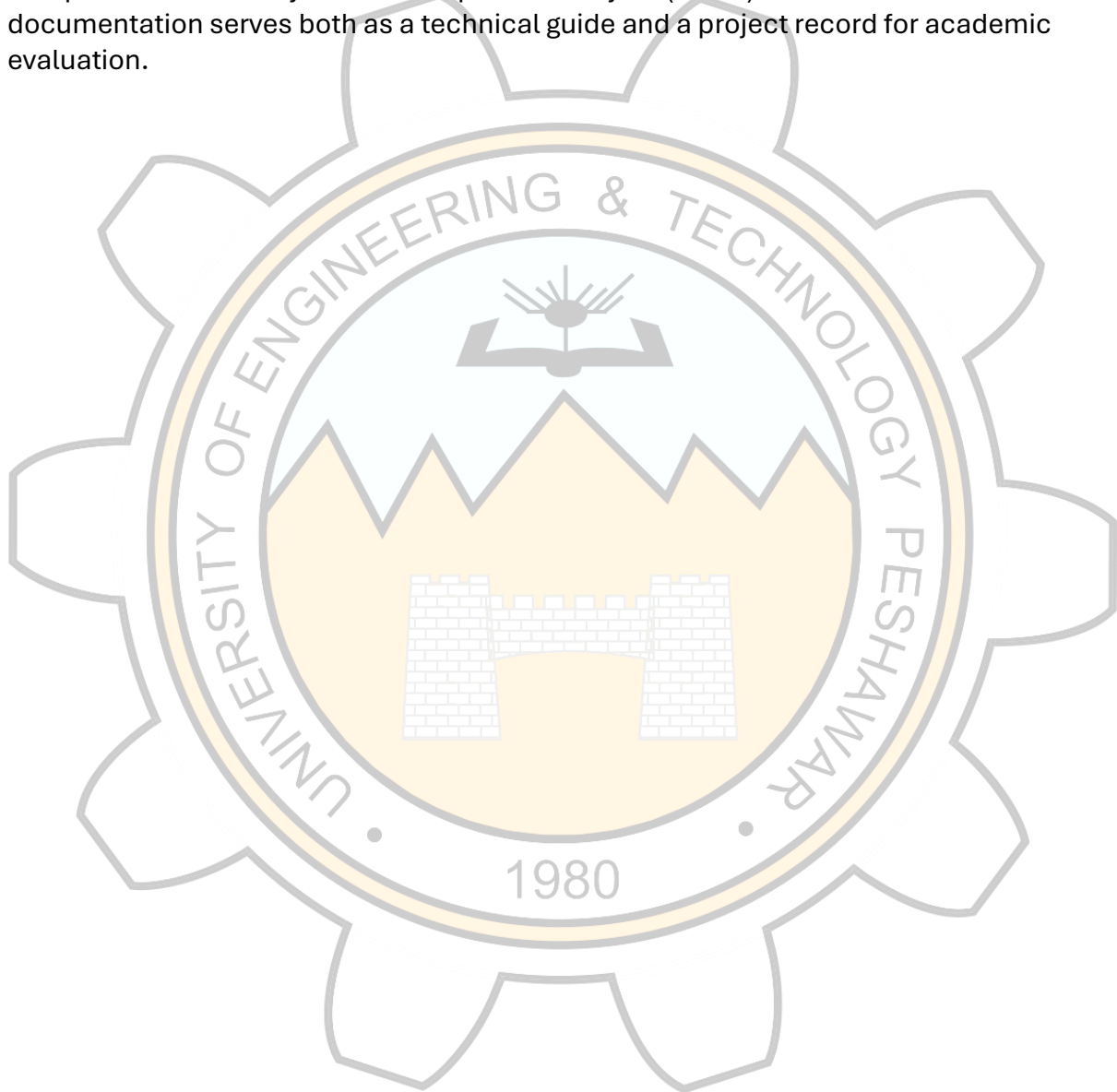
- **Chapter 5: Graphical User Interface Snapshots**

This section displays visual proof of the working system through screenshots of both the Administrator and User interfaces developed using Flutter.

- **References**

All external sources, research papers, and documentation used for development or inspiration during the project are cited here using APA referencing style.

Each chapter builds upon the previous one and reflects the progression and completeness of the system development life cycle (SDLC). This structured documentation serves both as a technical guide and a project record for academic evaluation.





## 2 LITERATURE REVIEW

### 2.1 BACKGROUND

Hostel management is an essential administrative function within educational institutions and organizations that provide accommodation facilities to students, staff, or guests. Efficient management of hostels ensures the proper allocation of rooms, maintenance of facilities, fee collection, and overall welfare of residents. Traditionally, hostel management was carried out manually using paper records, which often led to errors, delays, and inefficiencies.

With the advancement of information technology, computerized systems have been introduced to automate hostel operations, thereby improving accuracy, transparency, and ease of management. A Hostel Management System (HMS) integrates various functions such as room allocation, student and staff data management, fee processing, complaint handling, and reporting into a unified platform.

Many educational institutions worldwide have adopted digital solutions to streamline their hostel services. These systems help administrators efficiently manage resources, track residents' information, and enhance communication between management and occupants. Moreover, an effective HMS reduces paperwork, minimizes human errors, and provides timely access to information, which is critical for decision making and operational efficiency.

In the context of this project, a web and mobile-based hostel management system is proposed, utilizing modern technologies like MySQL for database management, Flutter for a responsive user interface, and PHP APIs to integrate frontend and backend components. This integrated approach aims to provide a scalable, user-friendly, and reliable platform for managing hostel affairs effectively.

### 2.2 DEFINITIONS & TERMS

Below are the key definitions and technical terms used throughout the project documentation for the **Hostel Management System**:

- **HMS (Hostel Management System):**  
A software application designed to automate and manage various administrative tasks related to hostel operations such as room allotment, student records, fee management, and maintenance tracking.
- **Database:**  
A structured collection of data that can be easily accessed, managed, and updated. In this project, **MySQL** is used as the backend database to store all hostel-related information securely.

- **Frontend:**  
The part of the application that users interact with. In this project, the frontend is developed using **Flutter**, providing cross-platform (Android, iOS, Web) support with a user-friendly interface.
- **Backend:**  
The server-side component of the application responsible for processing data, implementing logic, and communicating with the database. This project uses **PHP APIs** to handle backend operations and integrate with the frontend.
- **API (Application Programming Interface):**  
A set of defined rules and protocols that allow different software components to communicate. PHP APIs are used in this system to connect the Flutter frontend with the MySQL database.
- **Room Allocation:**  
The process of assigning available hostel rooms to registered students based on predefined criteria like availability, room type, and preferences.
- **Student Profile:**  
A digital record containing personal and academic details of hostel residents including name, roll number, contact, allocated room, fee status, and stay history.
- **Admin Panel:**  
A secure interface used by hostel administrators to manage system features such as adding new students, updating room details, monitoring occupancy, and generating reports.
- **MySQL Admin (phpMyAdmin):**  
A graphical tool used to interact with the MySQL database. It provides an interface for creating, managing, and querying the database.
- **Responsive Design:**  
A UI/UX design approach ensuring that the application interface adjusts and functions smoothly across various devices and screen sizes.

## 2.3 SOFTWARE & TOOLS

To successfully develop and deploy the *Hostel Management System*, a variety of software tools and technologies were used throughout the software development life cycle. Each tool was selected based on its compatibility, efficiency, and ease of use for the project.

### 1. MySQL

- **Purpose:** Database Management System
- **Use in Project:** Used to store and manage all backend data related to students, rooms, admin users, and fee structures. MySQL provides structured storage and querying capabilities, making it an efficient choice for relational data management in this system.

### 2. phpMyAdmin



- **Purpose:** Web-based MySQL database management tool
- **Use in Project:** Used to interact with the MySQL database through a GUI interface. Enabled easy creation of tables, relationships, and running SQL queries during development.

### 3. Flutter Framework

- **Purpose:** Frontend development toolkit by Google
- **Use in Project:** Used to design and build a responsive, cross-platform user interface. Flutter allows development of a single codebase for Android, iOS, and web, providing a seamless experience to both users and administrators.

### 4. PHP (Hypertext Preprocessor)

- **Purpose:** Server-side scripting language
- **Use in Project:** Used to create RESTful APIs that serve as the bridge between the Flutter frontend and MySQL backend. PHP APIs perform CRUD (Create, Read, Update, Delete) operations and ensure secure data transactions.

### 5. Visual Studio Code

- **Purpose:** Source code editor
- **Use in Project:** Used as the primary code editor for writing frontend Dart code (Flutter), PHP backend scripts, and managing project structure and resources.

### 6. Postman

- **Purpose:** API testing and development tool
- **Use in Project:** Used to test PHP APIs during development to ensure correct response structure, secure communication, and functional integrity between frontend and backend.

### 7. .NET Framework

- **Purpose:** Windows-based application development framework
- **Use in Project:** The .NET Framework was used to build desktop-based administrative tools and utilities required for managing hostel data locally on Windows. It allowed for the creation of a lightweight, Windows-native admin interface (optional module) that could access the MySQL database using ADO.NET or compatible MySQL connectors. The framework's reliability, integration with Visual Studio, and ease of deployment made it a suitable choice for rapid local admin operations and prototype testing.

### 8. Git & GitHub

- **Purpose:** Version control system
- **Use in Project:** Used to manage the source code, collaborate, and keep track of changes in the project during development.

### 3 ANALYSIS AND DESIGN

This chapter provides a detailed analysis and design of the Hostel Management System. It includes the Software Requirement Specifications (SRS) and UML diagrams which illustrate the structure and behavior of the system.

#### 3.1 SOFTWARE REQUIREMENT SPECIFICATIONS

##### 3.1.1 Problem Statement

Managing a hostel manually is time-consuming and prone to human error. Administrators face difficulties in keeping track of room allocations, student records, fees, complaints, and maintenance schedules. This often leads to mismanagement, confusion, and inefficient resource utilization.

The goal of the Hostel Management System is to automate these processes through a centralized digital platform. The system will allow administrators to register students, manage room availability, handle payments, monitor complaints, and maintain records efficiently.

This project offers a streamlined, user-friendly interface developed in Flutter, with a secure and efficient backend powered by MySQL. Integration is achieved through PHP APIs managed in MySQL Admin, ensuring reliable communication between frontend and backend components.

##### 3.1.2 Constraints and Limitations

- The system requires an active internet connection to synchronize between Flutter frontend and MySQL backend.
- The system does not currently support biometric authentication or mobile payment gateways.
- Multi-language support is not implemented in the initial version.
- Real-time notifications (e.g., SMS/email alerts) are outside the current scope.
- The admin panel is designed for desktop access; mobile optimization for admin is limited.

### 3.1.3 Functional Requirements

The following table outlines the functional requirements of the Hostel Management System. Each requirement is uniquely identified, verifiable, and prioritized to ensure the completeness and clarity of the system functionality.

Req. No.	Description
FR-01	The system shall allow users to log in with Admin credentials
FR-02	The system shall allow the <b>Admin</b> to create, read, update, and delete student records (name, contact, father's name, etc.).
FR-03	The system shall allow the <b>Admin</b> to create, read, update, and delete hostel records (name, type: boys/girls, capacity).
FR-04	The system shall allow the <b>Admin</b> to create, read, update, and delete room records (room number, type, capacity, current occupancy).
FR-05	The system shall enforce room capacity when allocating or reallocating students to rooms.
FR-06	The system shall allow the <b>Admin</b> to view and update student fee status (Paid, Unpaid, Pending).
FR-07	The system shall allow the <b>Admin</b> to create and manage fee transactions (amount, date, status) and generate printable payment receipts.
FR-08	The system shall allow students to view their own profile, room assignment, and fee status.
FR-09	The system shall provide search and filter functionality on the student list by <b>room, hostel, and fee status</b> .
FR-10	The system shall generate dashboards/reports for overall hostel occupancy, fee collection status, and complaint summaries (e.g., pie charts).
FR-11	The system shall enforce secure API access and input validation for all CRUD operations.

### 3.1.4 Non-Functional Requirements

This section describes the quality attributes and constraints that are essential for the proper functioning of the Hostel Management System. These include performance, reliability, availability, security, maintainability, and portability.

#### 3.1.4.1 Performance

- The system shall respond to 95% of user actions within **2 seconds**.
- The system shall support at least **50 concurrent users** without noticeable degradation in response times.

#### 3.1.4.2 Reliability

- The system shall achieve **99% uptime**, excluding scheduled maintenance windows.
- In the event of an unexpected server reboot or crash, the system shall **automatically recover** to its last consistent state without data loss.

#### 3.1.4.3 Availability

- The system shall be accessible **24/7** via both web and mobile (Flutter) interfaces.
- Any planned maintenance shall be **announced at least 24 hours in advance** to all users.

#### 3.1.4.4 Security

- All user authentication shall use **secure session management** (e.g., token-based or cookie-secure).
- **Passwords** must be stored in the database using a **strong hashing algorithm** (e.g., bcrypt).
- Only users with the appropriate **role-based permissions** (Admin, Student, Warden) can access restricted features.
- All data exchange between the Flutter frontend and PHP APIs shall occur over **HTTPS**, and inputs must be **validated and sanitized** on the server side.

#### 3.1.4.5 Maintainability

- The codebase shall follow a **modular architecture**, separating concerns (e.g., data access, business logic, presentation) to simplify future updates.
- Source code and documentation shall adhere to consistent **naming conventions**, include inline comments where necessary, and maintain an up-to-date project README.

#### 3.1.4.6 Portability

- The mobile application shall run on **Android** (API level 21+) and **iOS** devices via the Flutter framework.
- The backend (MySQL database and PHP APIs) shall be deployable on any standard web-hosting environment supporting **PHP 7.4+** and **MySQL 5.7+**, such as XAMPP, WAMP, or cloud-based LAMP/LEMP stacks.

### DATABASE ANALYSIS AND DESIGN

This chapter presents the detailed analysis and design of the database for the Hostel Management System. The database is designed to store, manage, and retrieve information related to hostel rooms, students, fees, complaints, and other relevant entities. The chapter includes an overview of the database, tools used, data dictionary, UML diagrams related to data flow and ER modeling, normalization, and database snapshots.

#### 4.1 DATABASE OVERVIEW

The database forms the backbone of the Hostel Management System by providing persistent storage and management of all data. It includes tables that handle student information, room details, fee transactions, complaints, and user credentials.

This database is designed with the goals of maintaining data integrity, supporting concurrent users, and providing quick data retrieval.

The system uses **MySQL** as the database management system due to its reliability, wide support, and ease of integration with PHP APIs. The database is accessed via PHP scripts that provide APIs consumed by the Flutter front end.

##### 4.1.1 Tools Used

- **MySQL:** For designing and managing the relational database.
- **phpMyAdmin:** A web interface to create, edit, and maintain the MySQL database.
- **MySQL Workbench** (optional): For visually designing the ER diagrams and writing SQL queries.
- **PHP:** To create APIs that connect the Flutter front end with the MySQL backend.
- **Flutter:** Used for building the user interface which interacts with the backend via APIs.

##### 4.1.2 Data Dictionary

The data dictionary defines the attributes for each table, their data types, constraints, and descriptions.



### 4.1.2 Data Dictionary

#### Student:

Attribute Name	Data Type	Constraints	Description
student_id	INT	PRIMARY KEY, AUTO_INCREMENT	Unique ID for each student
room_id	INT	FOREIGN KEY → Room(room_id)	Assigned room
student_name	VARCHAR(100)	NOT NULL	Student's full name
s_contact_no	VARCHAR(15)		Student's contact number
gender	ENUM('M','F','O')	NOT NULL	Gender (M=Male, F=Female, O=Other)

#### Hostel:

Attribute Name	Data Type	Constraints	Description
hostel_id	INT	PRIMARY KEY, AUTO_INCREMENT	Unique ID for each hostel
warden_id	INT	FOREIGN KEY → Warden(warden_id)	Assigned warden
hostel_name	VARCHAR(100)	NOT NULL	Name of the hostel
total_rooms	INT	NOT NULL	Total number of rooms in the hostel
rooms_occupied	INT	NOT NULL, DEFAULT 0	Number of rooms currently occupied

#### Room:

Attribute Name	Data Type	Constraints	Description
room_id	INT	PRIMARY KEY, AUTO_INCREMENT	Unique ID for each room
hostel_id	INT	FOREIGN KEY → Hostel(hostel_id)	Hostel to which this room belongs
capacity	INT	NOT NULL	Maximum number of students per room
occupied_count	INT	NOT NULL, DEFAULT 0	Current number of students assigned

#### Warden:

Attribute Name	Data Type	Constraints	Description
warden_id	INT	PRIMARY KEY, AUTO_INCREMENT	Unique ID for each warden

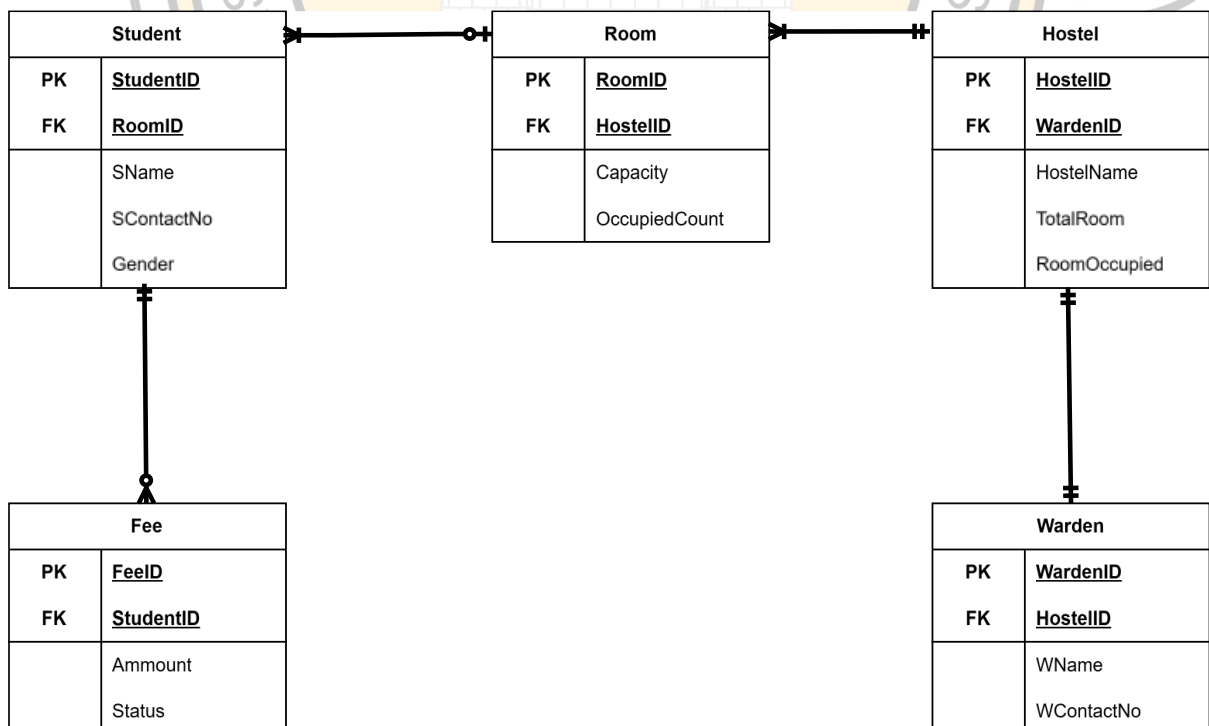
<b>hostel_id</b>	INT	FOREIGN KEY → Hostel(hostel_id)	Hostel under this warden's charge
<b>name</b>	VARCHAR(100)	NOT NULL	Warden's full name
<b>w_contact_no</b>	VARCHAR(15)		Warden's contact number

#### Fee:

Attribute Name	Data Type	Constraints	Description
<b>fee_id</b>	INT	PRIMARY KEY, AUTO_INCREMENT	Unique ID for each fee transaction
<b>student_id</b>	INT	FOREIGN KEY → Student(student_id)	Student who made the payment
<b>amount</b>	DECIMAL(10,2)	NOT NULL	Fee amount
<b>status</b>	ENUM('Paid','Unpaid','Pending')	NOT NULL	Payment status

## 4.2 Entity-Relationship Diagram

The ER Diagram represents the relationships between entities such as Students, Rooms, Fees, Complaints, and Admins.



Insert ER Diagram here

Figure 4.3: Entity-Relationship Diagram of Hostel Management System

### 4.3 NORMALIZATION

To ensure data consistency and reduce redundancy, the database tables are normalized up to the **Third Normal Form (3NF)**.

- **First Normal Form (1NF):** All tables have atomic columns; no repeating groups.
- **Second Normal Form (2NF):** All non-key attributes are fully functionally dependent on the primary key.
- **Third Normal Form (3NF):** No transitive dependencies exist; all attributes depend only on the primary key.

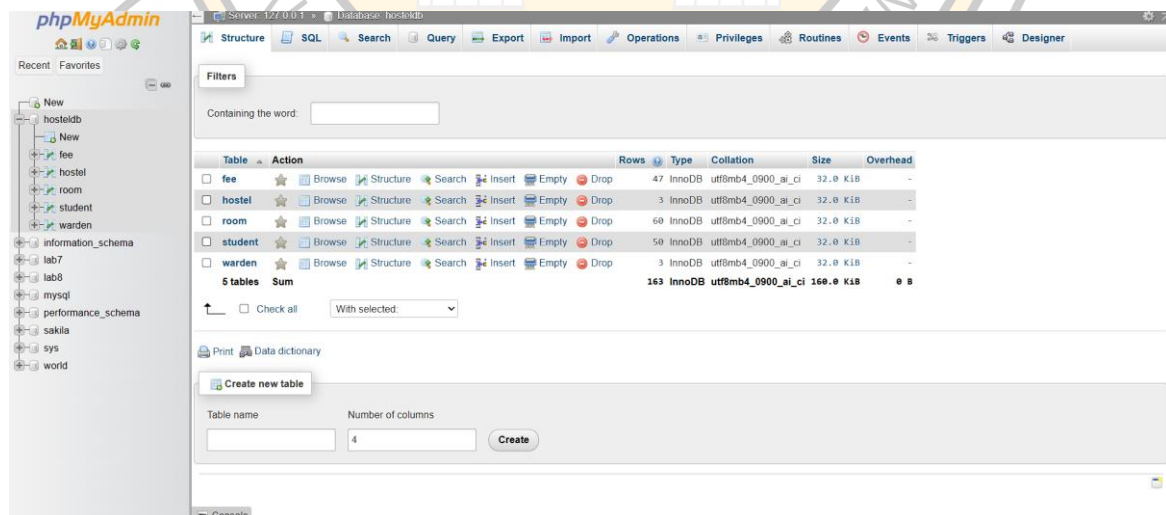
Example:

The Student table separates contact details and login credentials properly, avoiding redundant data storage.

### 4.4 DATABASE SNAPSHOTS

The following screenshots show the implemented tables and sample data from the MySQL database through phpMyAdmin or MySQL Workbench.

Insert snapshot images of Student table, Room table, Fee table, Warden table, Hostel table.





## Student Table

phpMyAdmin

Recent Favorites

New

hosteldb

New

fee

hostel

room

student

warden

information\_schema

lab7

lab8

mysql

performance\_schema

sakila

sys

world

Server: 127.0.0.1 » Database: hosteldb » Table: student

Browse

Structure

SQL

Search

Insert

Export

Import

Privileges



































































Operations

Triggers

←

→

StudentIDRoomIDStudentNameSContactNoGender

<input type="checkbox"/>	  	12 Ah07	Adeel	03111234567	Male
<input type="checkbox"/>	  	13 Ah09	Tariq	03121234567	Male
<input type="checkbox"/>	  	14 AI01	Shibu	03131234567	Male
<input type="checkbox"/>	  	15 AI01	Shah Zaman	03141234567	Male
<input type="checkbox"/>	  	16 AI02	Usman	03151234567	Male
<input type="checkbox"/>	  	17 AI02	Bilal	03161234567	Male
<input type="checkbox"/>	  	18 AI03	Danish	03171234567	Male
<input type="checkbox"/>	  	19 AI03	Arsalan	03181234567	Male
<input type="checkbox"/>	  	20 AI04	Zohaib	03191234567	Male
<input type="checkbox"/>	  	21 AI04	Kashif	03201234567	Male
<input type="checkbox"/>	  	22 AI05	Rizwan	03211234567	Male
<input type="checkbox"/>	  	23 AI05	Saad	03221234567	Male
<input type="checkbox"/>	  	24 AI06	Hasnain	03231234567	Male
<input type="checkbox"/>	  	25 AI06	Sufiyan	03241234567	Male
<input type="checkbox"/>	  	26 AI07	Imran	03251234567	Male
<input type="checkbox"/>	  	27 AI07	Shahbaz	03261234567	Male
<input type="checkbox"/>	  	28 AI08	Ahsan	03271234567	Male
<input type="checkbox"/>	  	29 AI08	Owais	03281234567	Male
<input type="checkbox"/>	  	30 Ay01	Aleena	03291234567	Female
<input type="checkbox"/>	  	31 Ay01	Rabia	03301234567	Female
<input type="checkbox"/>	  	32 Ay02	Lakshmi	03311234567	Female
<input type="checkbox"/>	  	33 Ay02	Fatima	03321234567	Female

Console

## Hostel Table:

phpMyAdmin

Recent Favorites

Server: 127.0.0.1 » Database: hosteldb » Table: hostel

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

Showing rows 0 - 2 (3 total, Query took 0.0005 seconds.)

SELECT \* FROM `hostel`

☐ Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

	HostelID	WardenID	HostelName	TotalRooms	RoomOccupied
<input type="checkbox"/>	Edit Copy Delete	1	1 Ahmad Hostel	20	10
<input type="checkbox"/>	Edit Copy Delete	2	2 Ali Hostel	20	12
<input type="checkbox"/>	Edit Copy Delete	3	3 Ayesha Hostel	20	8

☐ Check all | With selected: Edit Copy Delete Export

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Query results operations

Print Copy to clipboard Export Display chart Create view

Console

## Room Table:

phpMyAdmin

Recent Favorites

Server: 127.0.0.1 » Database: hosteldb » Table: room

Showing rows 0 - 24 (60 total, Query took 0.0005 seconds)

`SELECT * FROM `room``

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

1 > >> ☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

	RoomID	HostelID	Capacity	OccupiedCount
<input type="checkbox"/> Edit Copy Delete	Ah01	1	2	2
<input type="checkbox"/> Edit Copy Delete	Ah02	1	2	2
<input type="checkbox"/> Edit Copy Delete	Ah03	1	2	2
<input type="checkbox"/> Edit Copy Delete	Ah04	1	2	2
<input type="checkbox"/> Edit Copy Delete	Ah05	1	2	2
<input type="checkbox"/> Edit Copy Delete	Ah06	1	2	1
<input type="checkbox"/> Edit Copy Delete	Ah07	1	2	1
<input type="checkbox"/> Edit Copy Delete	Ah08	1	2	1
<input type="checkbox"/> Edit Copy Delete	Ah09	1	2	1
<input type="checkbox"/> Edit Copy Delete	Ah10	1	2	1
<input type="checkbox"/> Edit Copy Delete	Ah11	1	2	0
<input type="checkbox"/> Edit Copy Delete	Ah12	1	2	0
<input type="checkbox"/> Edit Copy Delete	Ah13	1	2	0
<input type="checkbox"/> Edit Copy Delete	Ah14	1	2	0

Console

## Warden Table:

phpMyAdmin

Recent Favorites

Server: 127.0.0.1 » Database: hosteldb » Table: warden

Showing rows 0 - 2 (3 total, Query took 0.0010 seconds)

`SELECT * FROM `warden``

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

	WardenID	HostelID	Name	WContactNo
<input type="checkbox"/> Edit Copy Delete	1	1	Mr. Ahmad Khan Rajput	03001234567
<input type="checkbox"/> Edit Copy Delete	2	2	Mr. Faizanullah Wazir	03012345678
<input type="checkbox"/> Edit Copy Delete	3	3	Ms. Aleena Khan	03023456789

☐ Check all | With selected: Edit Copy Delete Export

☐ Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Query results operations

Print Copy to clipboard Export Display chart Create view

Console

## Fee Table:

phpMyAdmin

Server: 127.0.0.1 » Database: hosteldb » Table: fee

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

Showing rows 0 - 24 (47 total, Query took 0.0006 seconds.)

`SELECT * FROM `fee``

☐ Profiling [\[ Edit inline \]](#) [\[ Edit \]](#) [\[ Explain SQL \]](#) [\[ Create PHP code \]](#) [\[ Refresh \]](#)

1 > >> ☐ Show all Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

	FeeID	StudentID	Amount	Status
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	1	1	20000.00	Paid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	2	2	20000.00	Unpaid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	4	4	20000.00	Paid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	5	5	20000.00	Unpaid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	6	6	20000.00	Unpaid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	7	7	20000.00	Unpaid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	8	8	20000.00	Unpaid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	9	9	20000.00	Paid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	10	10	20000.00	Paid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	11	11	20000.00	Unpaid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	12	12	20000.00	Pending
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	13	13	20000.00	Paid
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	14	14	20000.00	Pending
<input type="checkbox"/> <a href="#">Edit</a> <a href="#">Copy</a> <a href="#">Delete</a>	15	15	20000.00	Paid

localhost/phpmyadmin/index.php?route=/sql&pos=0&db=hosteldb&table=hostel

### GRAPHICAL USER INTERFACE SNAPSHOTS

This chapter presents the graphical user interface (GUI) of the Hostel Management System, designed solely for **administrative users** using the Flutter framework. The interface is focused on usability, responsiveness, and clarity to ensure efficient management of hostel operations such as student handling, room allocation, fee tracking, and warden listing.

The frontend is integrated with the MySQL backend through PHP APIs. All changes made through the interface are reflected in the database in real-time.

#### 5.1 OVERVIEW

The application supports **only one user role**:

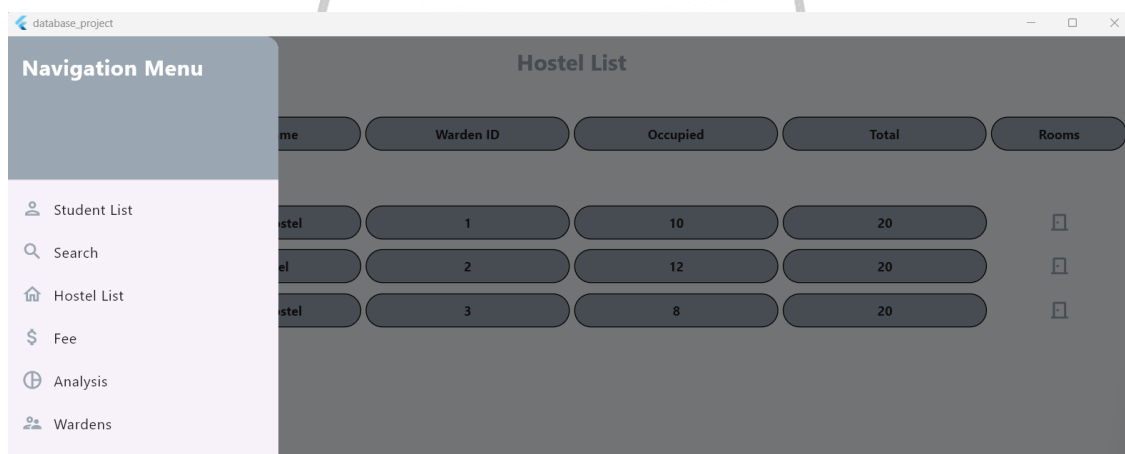
- **Admin**

Key functionalities available in the interface include:

- Adding, updating, and deleting student records
- Viewing hostel and room lists
- Checking room occupancy and free room availability
- Managing fees
- Searching student data
- Viewing wardens and assigning them to hostels

All user actions are validated and synchronized with the backend system securely.

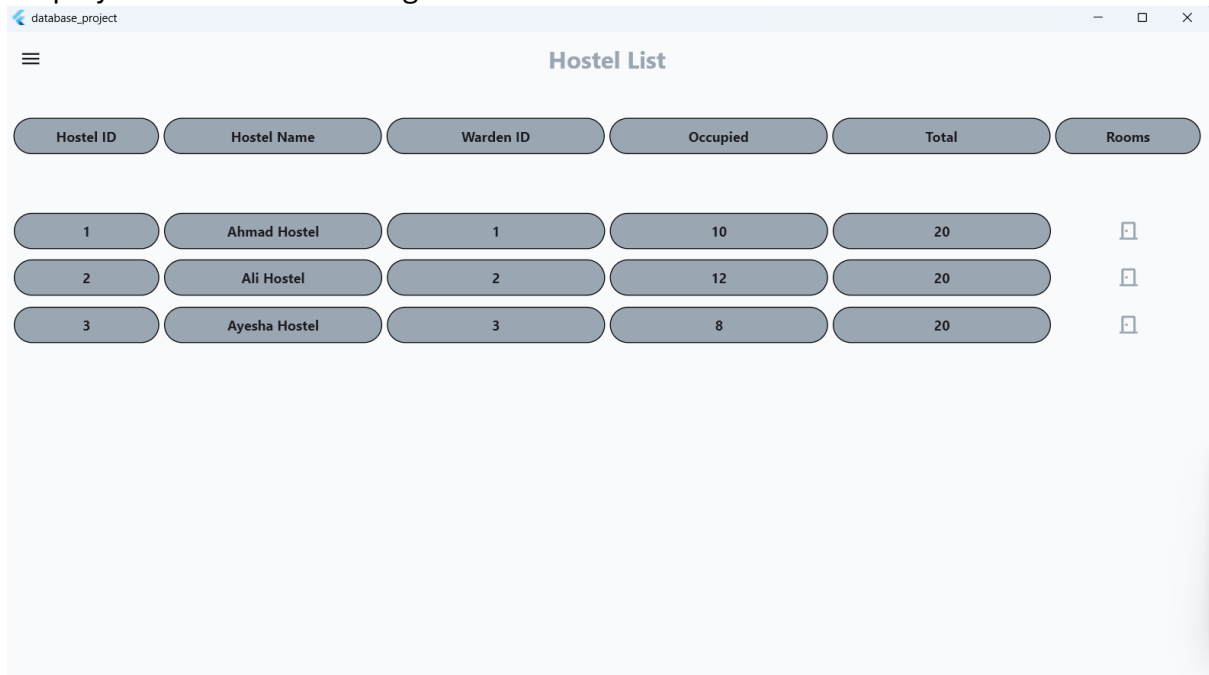
#### 5.2 ADMINISTRATIVE INTERFACE SCREENS






### 5.2.1 Hostel List Screen

#### Description:

Displays all hostels. Selecting a hostel shows the list of rooms within that hostel.



Hostel ID	Hostel Name	Warden ID	Occupied	Total	Rooms
1	Ahmad Hostel	1	10	20	
2	Ali Hostel	2	12	20	
3	Ayesha Hostel	3	8	20	

### 5.2.2 Room List (Rooms in Hostel)

#### Description:

After selecting a hostel, rooms associated with it are shown. Each room displays its occupancy and capacity.

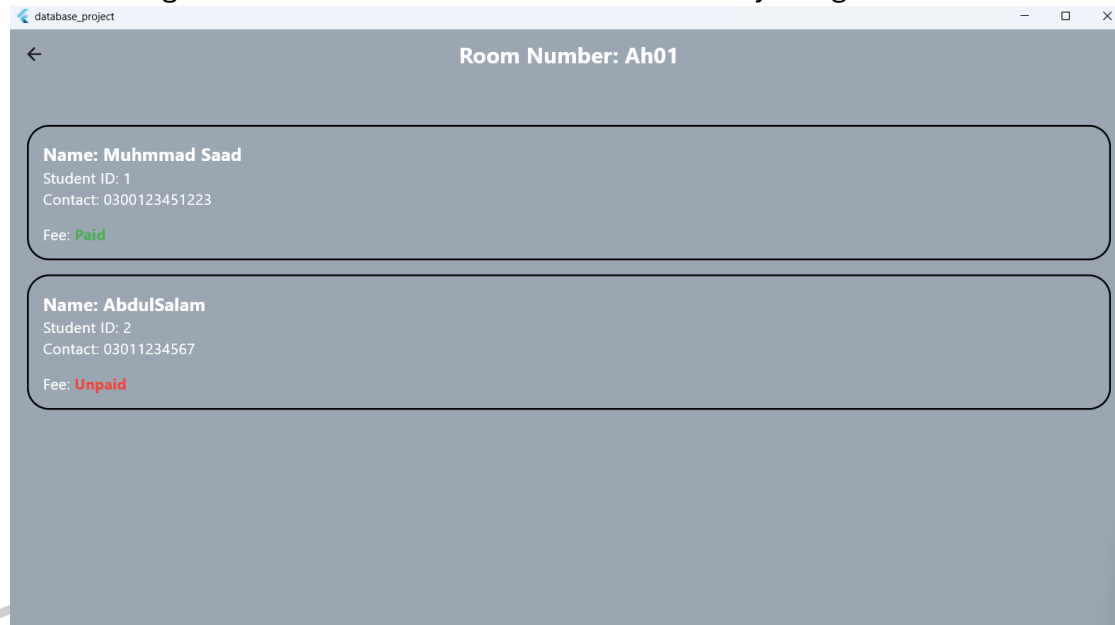


Ahmad Hostel	
Warden: Mr. Ahmad Khan Rajput	
AH01	Occupied: 2/2
AH02	Occupied: 1/2
AH03	Occupied: 2/2
AH04	Occupied: 2/2
AH05	Occupied: 2/2
AH06	Occupied: 1/2
AH07	Occupied: 1/2

### 5.2.3 Students in Room

#### Description:

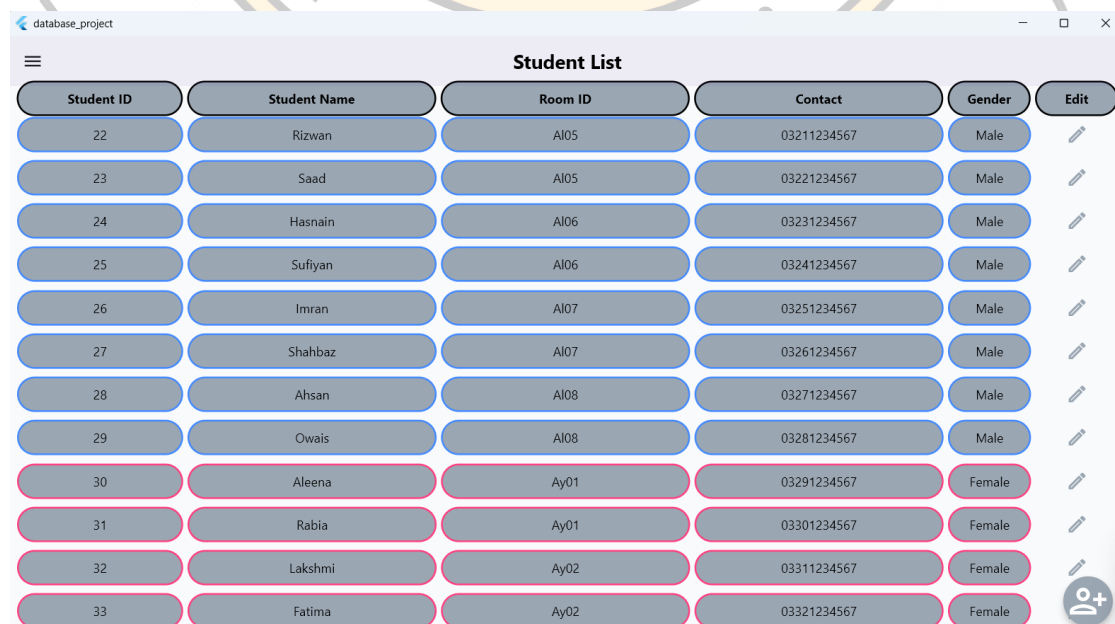
Selecting a room shows the list of students currently assigned to that room.



### 5.2.4 Student List

#### Description:

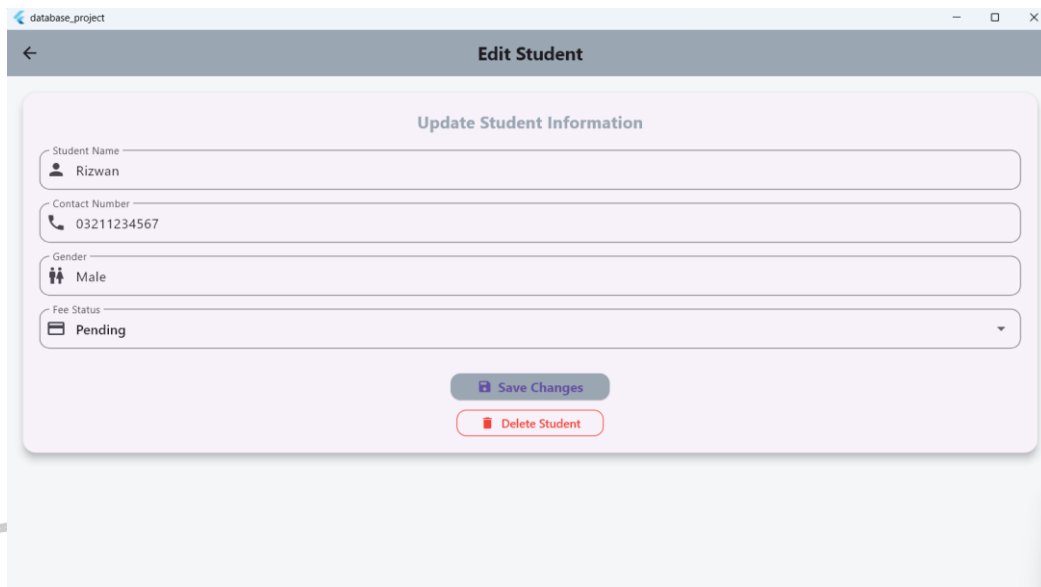
A complete list of all registered students. Tapping a student opens the edit/delete interface.



### 5.2.5 Edit/Delete Student

#### Description:

Allows the admin to update student details or remove them from the system.



The screenshot shows a web browser window titled 'database\_project' with a back arrow and a title bar. The main heading is 'Edit Student'. Below it is a form titled 'Update Student Information'. The form has four input fields: 'Student Name' with the value 'Rizwan', 'Contact Number' with the value '03211234567', 'Gender' with the value 'Male', and 'Fee Status' with a dropdown menu showing 'Pending'. At the bottom of the form are two buttons: 'Save Changes' (blue) and 'Delete Student' (red).

### 5.2.6 Add Student

#### Description:

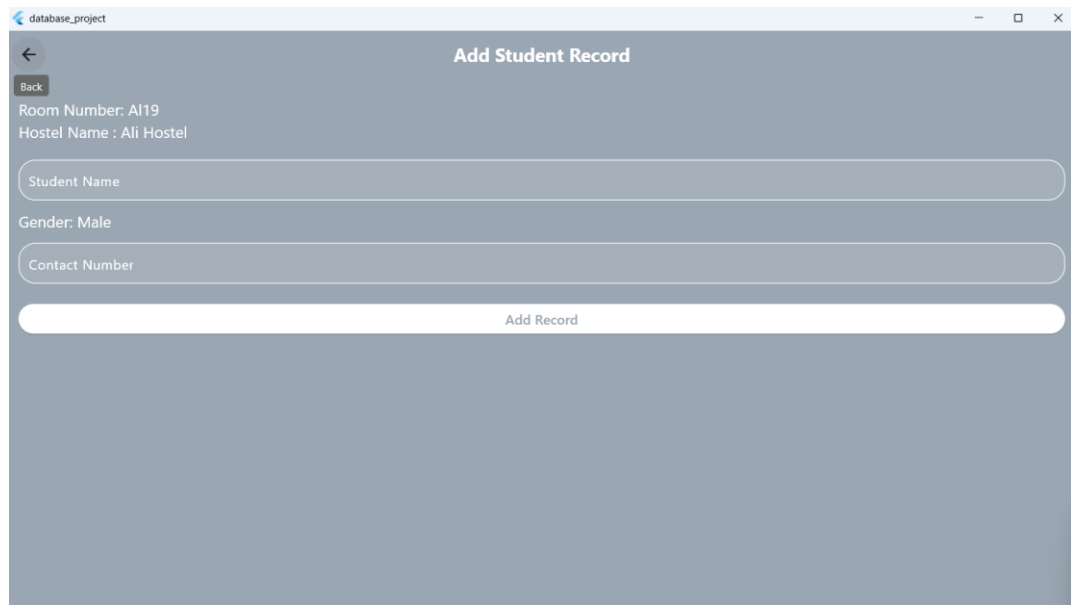
Accessible from the Student List screen. Displays a list of all free rooms. After selecting a room, the admin can enter the student's name and contact. The system determines the gender based on room ID (e.g., AH01 = Male, AY01 = Female).



The screenshot shows a web browser window titled 'database\_project' with a back arrow and a title bar. The main heading is 'List of Free Rooms'. Below it is a list of rooms, each with a room ID, name, and occupancy status. The rooms are: AL17 (Ali Hostel, Occupied: 0 / 2), AL18 (Ali Hostel, Occupied: 0 / 2), AL19 (Ali Hostel, Occupied: 0 / 2), AL20 (Ali Hostel, Occupied: 0 / 2), AY08 (Ayesha Hostel, Occupied: 1 / 2), AY10 (Ayesha Hostel, Occupied: 0 / 2), and AY11 (Ayesha Hostel, Occupied: 0 / 2).

Room ID	Room Name	Occupied	Total
AL17	Ali Hostel	0	2
AL18	Ali Hostel	0	2
AL19	Ali Hostel	0	2
AL20	Ali Hostel	0	2
AY08	Ayesha Hostel	1	2
AY10	Ayesha Hostel	0	2
AY11	Ayesha Hostel	0	2





database\_project

### Add Student Record

Back

Room Number: AI19  
Hostel Name: Ali Hostel

Student Name

Gender: Male

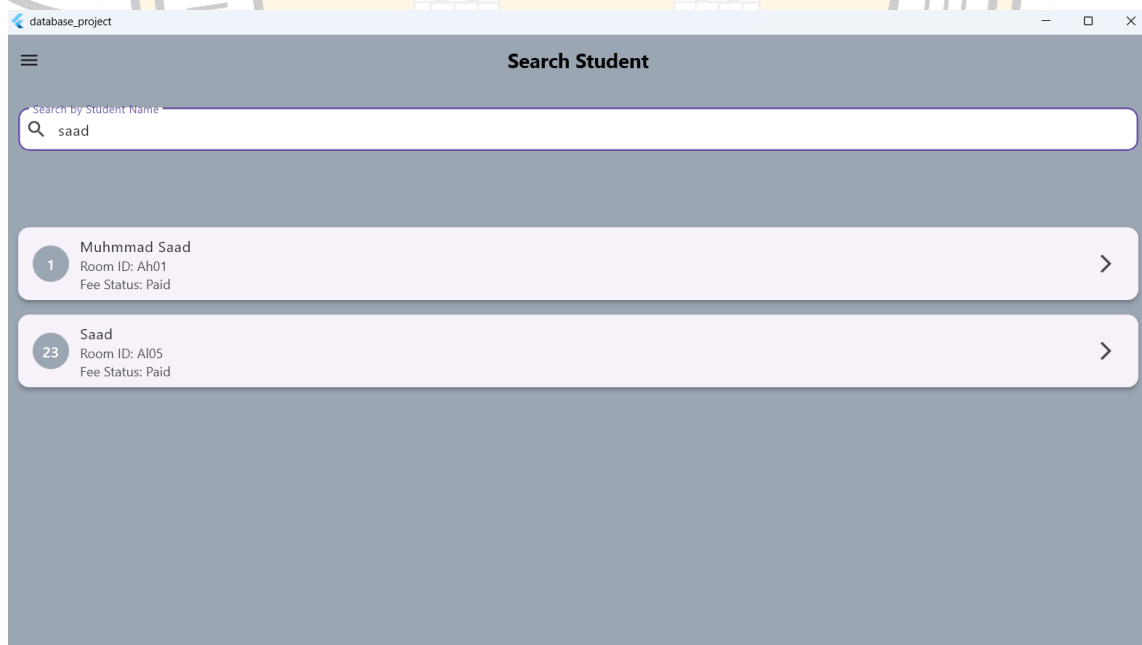
Contact Number

Add Record

## 5.2.7 Search Student

### Description:

Live search as the user types a student name. Matches show student ID and room. Tapping a result allows editing or deleting.



database\_project

### Search Student

Search by Student Name

Q saad

1	Muhmmad Saad Room ID: Ah01 Fee Status: Paid	>
23	Saad Room ID: AI05 Fee Status: Paid	>



## 5.2.8 Warden List and Edit

### Description:

Shows a list of wardens. Pressing the "Edit" button allows modification and saving of warden details.

The screenshot shows a web application window titled "database\_project" with a "Warden List" header. It contains three warden entries, each in a light purple box. The first entry, "Ahmad Hostel", has a "Save" button. The second, "Ali Hostel", and the third, "Ayesha Hostel", each have an "Edit" button (represented by a blue pencil icon).

Warden Name	Contact No	Action
Mr. Ahmad Khan Rajput	03001234567	Save
2. Mr. Faizanullah Wazir	Contact: 03012345678	Edit
3. Ms. Aleena Khan	Contact: 03023456789	Edit

## 5.2.9 Fee Management

### Description:

Displays each student's fee status: Paid, Unpaid, or Pending. The admin can change and save the status.

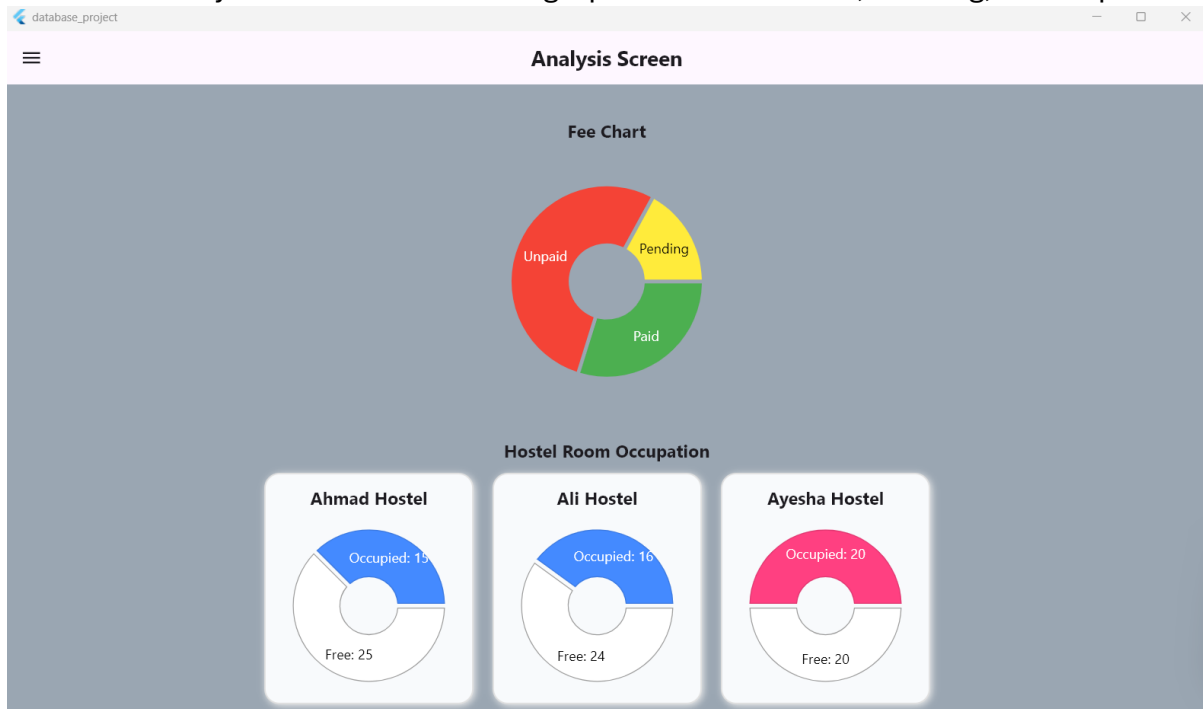
The screenshot shows a web application window titled "database\_project" with a "Fee List" header. A dropdown menu is open, showing options: "All", "Paid", "Unpaid", and "Pending". The menu is currently set to "Paid". Below the menu, there are four student entries, each with a "Save" button. Each entry displays the student's name, ID, Fee ID, and Amount.

Student Name	ID	Fee ID	Amount	Status	Action
Muhammad Saad	1	1	Rs. 20000.00	Paid	Save
Ibrahim Khan	4	4	Rs. 20000.00	Paid	Save
Prithviraj Chauhan	9	9	Rs. 20000.00	Paid	Save
Irfan	10	10	Rs. 20000.00	Paid	Save

### 5.2.10 Analysis Screen

#### Description:

Visual summary of fee distribution using a pie chart: total Paid, Pending, and Unpaid.



### 5.3 RESPONSIVE DESIGN ELEMENTS

- The application uses responsive layouts suitable for mobile and tablet devices.
- Flutter widgets such as ListView, Card, TextFormField, and Drawer ensure smooth scrolling and navigation.
- Backend connectivity allows real-time updates without requiring manual refreshes.

### 5.4 USER-FRIENDLINESS AND ACCESSIBILITY

- **Simple Navigation:** Admin menu and navigation drawer provide quick access to all features.
- **Consistent Layout:** All forms and lists follow a uniform design pattern for familiarity.
- **Input Validation:** Each form validates user input to prevent errors and ensure data integrity.