

## EXPERIMENT – 3

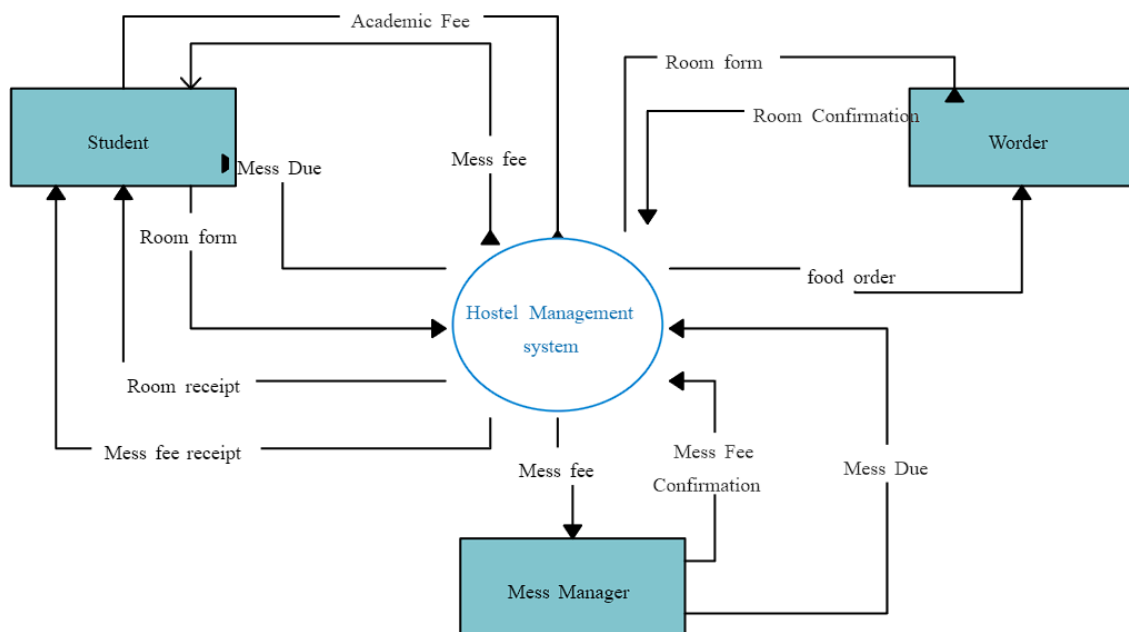
**AIM:** To draw DATA FLOW DIAGRAM (DFD) at Level 0 and Level 1.

**OUTPUT:**

### HOSTEL MANAGEMENT SYSTEM

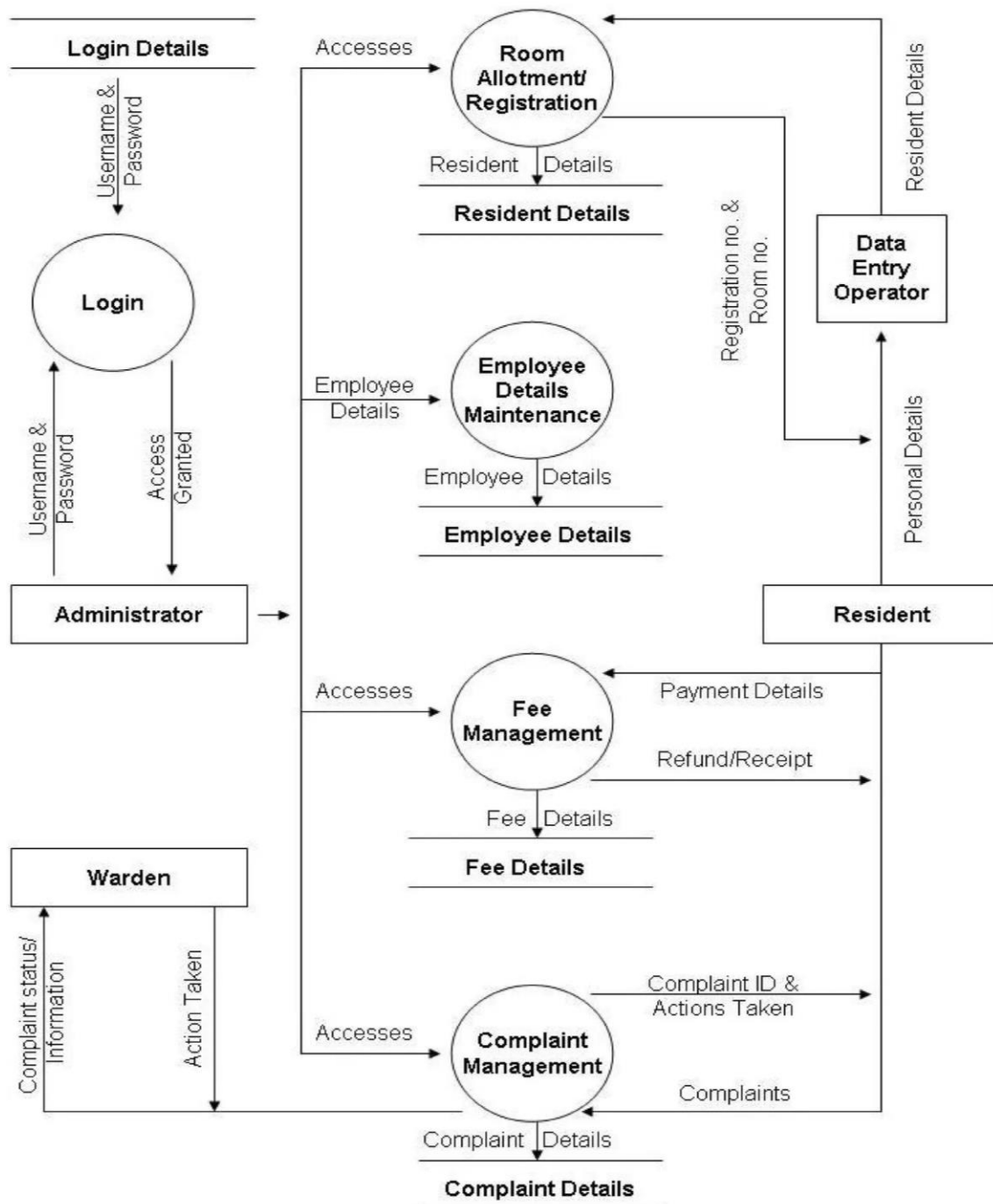
DFD LEVEL 0:

Level 0 DFDs, also known as context diagrams, are the most basic data flow diagrams. They provide a broad view that is easily digestible but offers little detail.



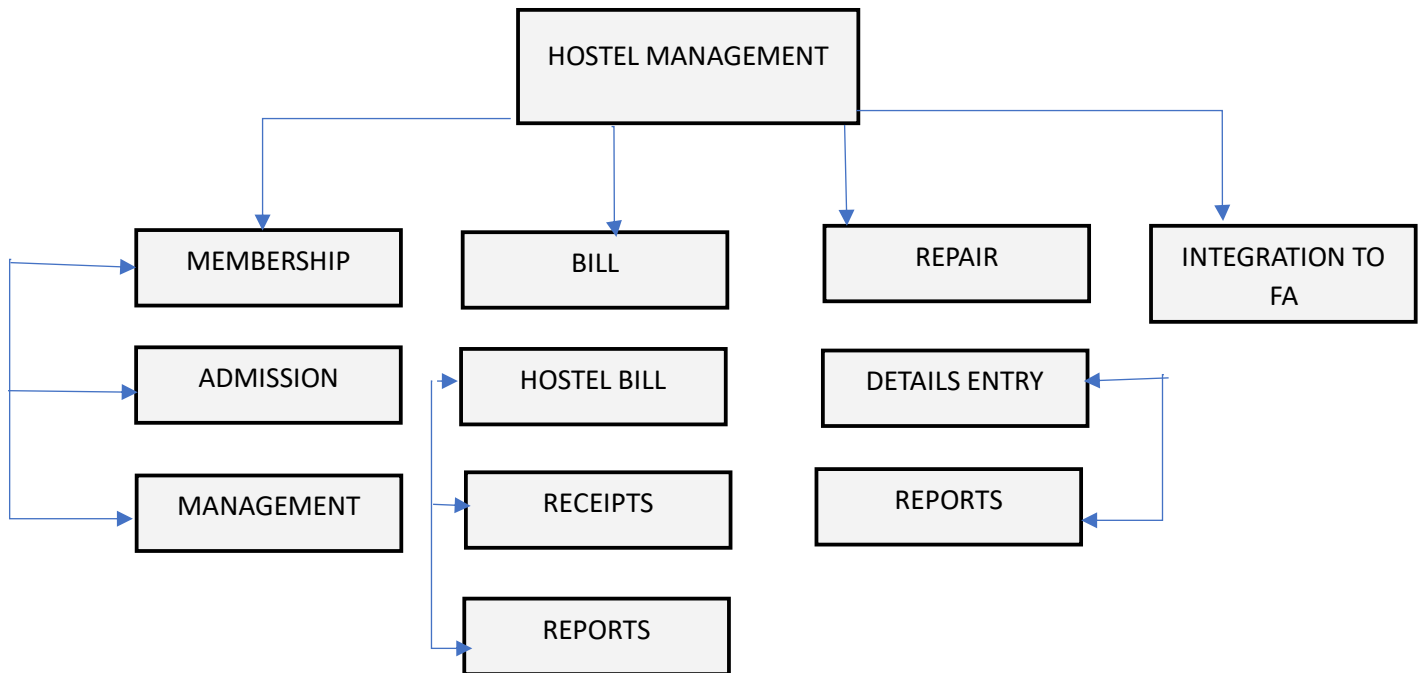
## DFD LEVEL 1:

A level 1 DFD notates each of the main sub-processes that together form the complete system. We can think of a level 1 DFD as an “exploded view” of the context diagram.



## STRUCTURED CHART:

A structure chart is a top-down modular design tool, constructed of squares representing the different modules in the system, and lines that connect them.

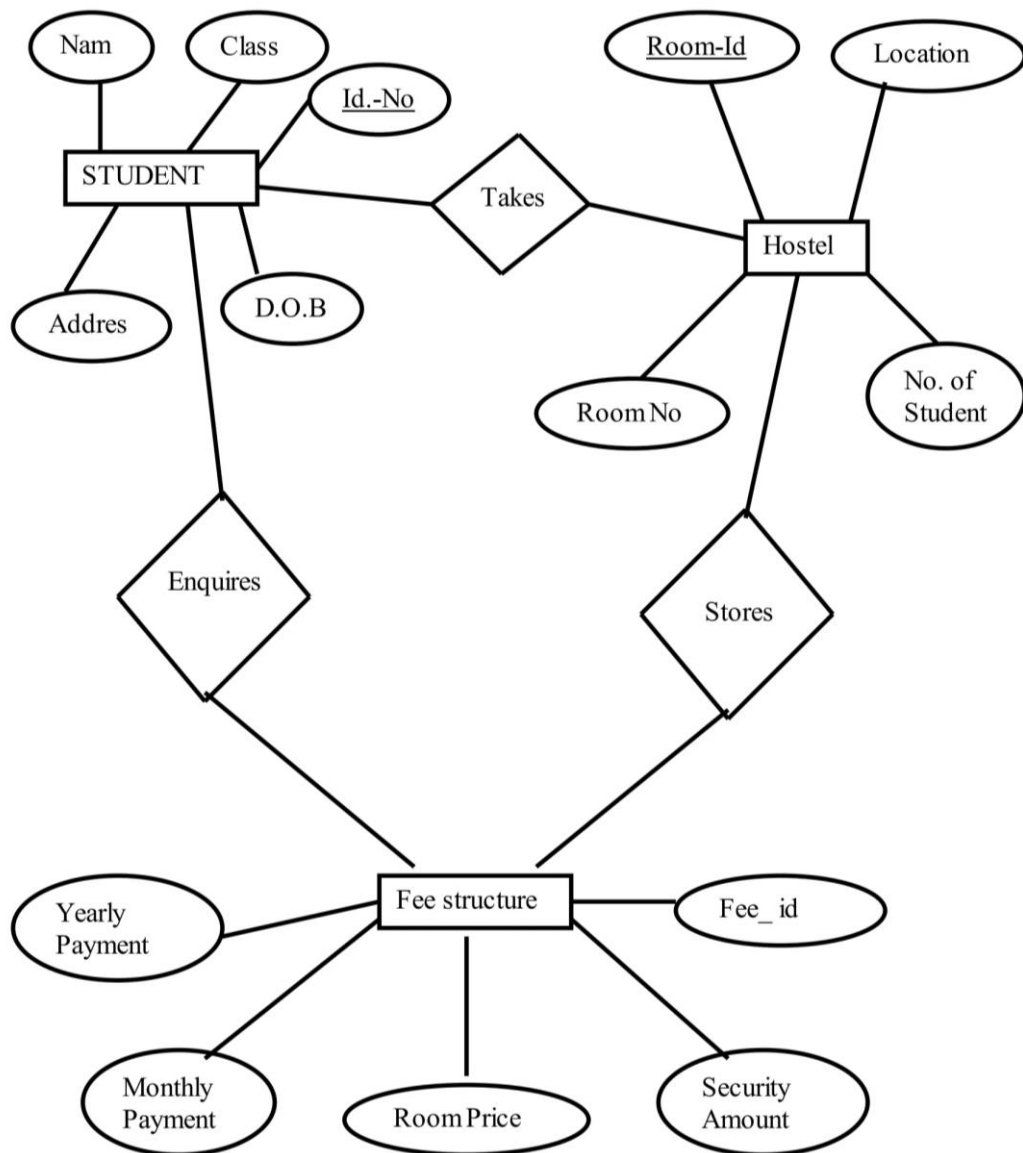


## EXPERIMENT – 4

**AIM:** To draw ER Diagram

**OUTPUT:**

### HOSTEL MANAGEMENT SYSTEM



# **EXPERIMENT -1**

**AIM:** To prepare Problem Statement for Hostel Management System

## **HOSTEL MANAGEMENT SYSTEM**

The Hostel Management System is developed for advancing the activities of the hostel. The main benefit of this software is to remove manual systems. Since most hostels are being run by only one hostels are being run by only one hostel manager, it would be difficult for him to manage all the data about each student and room, which are either saved on some papers or sometimes received. If these papers are missing or damaged, one would never be able to know the information. If the hostel is large enough, the employer might not even know the number of students in a room or if a room is empty or not. This project will be a great relief to the hostel manager. It will help to carry out the activities of the Hostel in an efficient way.

The existing system is highly manual, involving a lot of paperwork and calculations and therefore may be erroneous. This has led to inconsistency and inaccuracy in the maintenance of data. The data, which is stored on the paper only, may be lost, stolen, or destroyed due to natural calamities like fire and water. The existing System is sluggish and consumes a lot of time causing inconvenience to students and the employees. Due to the manual nature, it is difficult to update, delete, add, or view the data. Since the number of Students is extremely difficult.

A proposed Hostel Management system should be equipped with a comprehensive set of features to efficiently oversee the day-to-day operations of a Hostel or dormitory. These features encompass user management, including various access levels for administrators, staff and students, and robust user authentication. Room allocation capabilities should facilitate the assignment of rooms to new students, as well as the management of check-in and check-out processes, along with handling room change requests. Inventory management functionality helps keep track of hostels furniture and equipment, ensuring they are well-maintained and adequate replenished. Student information profiles, complete with personal and contact details, emergency contacts, and academic information should be easily accessible. Billing and payment features should support the creation of the structures, the generation bills and efficient payment tracking including late fee notifications. Additionally, the system should offer attendance tracking, visitor management, and facilities reservation capabilities.

A hostel management system comes with its fair share of disadvantages. Firstly, there are substantial initial costs associated with its implementation, encompassing, expenses for software licenses, necessary hardware, and staff training. For smaller hostels operating on tight budgets, this upfront investment composes a significant financial challenge. Additionally, technical issues, including software glitches & compatibility problems, can disrupt hostel operations, leading to guest dissatisfaction & operational inefficiencies. Maintenance & regular updates are essential but can be time-consuming & may have some additional costs. Moreover, there is a learning curve for staff members adapting to the new system, potentially resulting in operational hiccups until proficiency is achieved. Dependency on technology, data security, consumers, migration limitations, compatibility issues, guest privacy worries & staff resistance to change are among the other drawbacks that hostel management systems may present.

A well-designed Hostel Management system is an indispensable tool for efficiently managing all aspects of hostel operations. It empowers administrators, staff, and students with streamlined processes, enhanced communication and improved decision-making and resource optimization. This system's adaptability ensures it remains relevant in the ever-changing landscape of educational institutions ultimately, it contributes significantly to enhancing the overall quality of the student's life.

# 1. Introduction

## 1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to comprehensively define the functional and non-functional requirements of the Hostel Management System (HMS). It serves as a foundational document that provides clear and detailed guidance to software developers, testers, project managers, and all stakeholders involved in the design, development, and implementation of the HMS.

## 1.2 Scope

The scope of the HMS encompasses the development of a robust and user-friendly software application tailored for hostel management. The HMS aims to automate and streamline various hostel-related processes, facilitating efficient and error-free management of student accommodations, fees, inventory, and reporting.

## 1.3 Definitions, Acronyms, and Abbreviations

- HMS (Hostel Management System): The software application designed to automate and streamline hostel management processes.
- SRS (Software Requirements Specification): A document that outlines the functional and non-functional requirements of the HMS.
- UI (User Interface): The graphical or command-line interface through which users interact with the system.
- DBMS (Database Management System): Software used to manage the storage and retrieval of data in the system.
- API (Application Programming Interface): A set of rules and protocols that allows different software applications to communicate with each other.
- GDPR (General Data Protection Regulation): European Union data protection and privacy regulation.

## 1.4 Document Conventions

Throughout this SRS document, the following conventions will be adhered to for clarity and consistency:

- Section Structure: Sections and subsections will be organized using numbered headings (e.g., "2. Overall Description," "3. Specific Requirements") for easy navigation and reference.
- Appendices: Appendices will be used to provide additional information or reference materials and will be labelled as "Appendix A".
- Emphasis: Key terms or phrases will be italicized (e.g., *User Registration*) for emphasis.
- Lists: Bulleted and numbered lists will be used for presenting items and requirements in a structured manner.

## 1.5 References

This SRS document may refer to the following documents, standards, or external sources:

- [studylib.net](http://studylib.net)
- [slideshare.net](http://slideshare.net)

## 1.6 Overview

The Hostel Management System (HMS) is designed to address the complexities and challenges associated with hostel administration and management. By offering a centralized and automated platform, the HMS aims to enhance the efficiency, accuracy, and transparency of hostel-related operations.

This SRS document presents a detailed account of the requirements that the HMS must fulfil. It outlines the functionalities that the system will offer, the interactions it will support, and the quality attributes it must adhere to. Additionally, this document provides a comprehensive understanding of the system's context, including its users, constraints, assumptions, and dependencies.

The subsequent sections of this SRS will delve into specific aspects of the HMS, including functional requirements, non-functional requirements, system features, and external interfaces. It will serve as a reference point for all project stakeholders, guiding the development team in the creation of a reliable and user-friendly Hostel Management System that meets the needs of both hostel staff and students.



## **2. Overall Description**

### **2.1 Product Perspective**

The Hostel Management System (HMS) is a stand-alone software application that integrates seamlessly into the existing hostel management infrastructure. It interacts with other systems, such as the database management system (DBMS) for data storage, payment gateways for fee processing, and external services for communication.

### **2.2 Product Functions**

The HMS is designed to automate and streamline various hostel management tasks, enhancing efficiency and accuracy. Its core functions include:

#### **2.2.1 User Registration**

Description: Users, including hostel staff and students, can register their profiles in the system, providing essential information such as personal details and roles.

#### **2.2.2 Room Allocation**

Description: Hostel staff can allocate rooms to registered students based on their preferences and availability, ensuring efficient room management.

#### **2.2.3 Fee Management**

Description: The system manages student fees, including fee calculations, payment processing, and automated reminders for overdue payments.

#### **2.2.4 Check-In and Check-Out**

Description: The HMS facilitates the check-in and check-out processes for students, providing a smooth transition into and out of hostel accommodation.

#### **2.2.5 Inventory Management**

Description: Hostel staff can efficiently track and manage hostel inventory, ensuring timely restocking of essential items.

#### 2.2.6 Reporting

Description: The system generates comprehensive reports for administrators, allowing them to monitor hostel occupancy, financials, and inventory.

#### 2.2.7 Security and Access Control

Description: Robust security measures are implemented to safeguard user data and control access to the system based on user roles.

### 2.3 User Characteristics

The HMS caters to three main types of users, each with specific roles and responsibilities:

- Administrator: Administrators have full access to system features and are responsible for overseeing hostel operations.
- Hostel Staff: Hostel staff members manage room allocations, fees, inventory, and other day-to-day operations.
- Students: Students use the system to check room availability, make payments, and interact with the hostel management process.

### 2.4 Design/implementation constraints

Description: The HMS operates within the following constraints:

- The system must be accessible via standard web browsers (e.g., Chrome, Firefox, Safari).
- It must be compatible with the latest versions of these web browsers.
- Compliance with data security and privacy regulations, such as GDPR, is mandatory.

### 2.5 Assumptions and Dependencies

Description: Assumptions made during the development of the HMS and its dependencies include:

- The availability of suitable hosting infrastructure and resources.
- Integration with external payment gateways for secure transactions.
- Availability of personnel responsible for system maintenance and support.

### 3. External Interface Requirements

#### 3.1 User Interfaces

##### 3.1.1 User Registration Interface

Description: The user registration interface allows hostel staff and students to create accounts.

Inputs:

- User details, including name, username, password, email, and role.

Processing:

- Validates user inputs for data accuracy and security.
- Registers users by storing their details in the database.

Outputs:

- Confirmation message upon successful registration.
- Appropriate error messages if registration fails (e.g., duplicate username, invalid email).

##### 3.1.2 Dashboard Interface

Description: The dashboard serves as the main interface for users to access and manage hostel-related tasks.

Inputs:

- User credentials for authentication.
- User interactions to perform tasks such as room allocation, fee management, check-in/out, and inventory management.

Processing:

- Authenticates users based on provided credentials.
- Provides access to specific functionalities based on user roles (e.g., student, staff, administrator).

Outputs:

- Display of relevant data and functionality options based on user role.

### 3.1.3 Room Allocation Interface

Description: This interface allows staff to allocate rooms to students.

Inputs:

- Student details and room preferences.

Processing:

- Checks room availability.
- Suggests suitable rooms based on preferences.
- Assigns a room to the student and updates room status.

Outputs:

- Confirmation of room allocation.
- Notification if no suitable rooms are available.

## 3.2 Hardware Interfaces

### 3.2.1 Database Server

Description: The HMS interacts with a database server to store and retrieve data.

Inputs:

- SQL queries for data retrieval and modification.

Processing:

- Executes SQL queries to fetch or update data.

Outputs:

- Retrieved data or confirmation of data modification.

### 3.2.2 Payment Gateway Integration

Description: The system communicates with payment gateways for processing fee payments.

Inputs:

- Payment details provided by students.

Processing:

- Sends payment information to the payment gateway.
- Receives payment status (success or failure) from the gateway.

Outputs:

- Confirmation of successful payment or error message in case of payment failure.

### 3.3 Software Interfaces

#### 3.3.1 Database Management System (DBMS)

Description: The system relies on a DBMS to manage the database.

Inputs:

- SQL queries for database operations (e.g., SELECT, INSERT, UPDATE).

Processing:

- The DBMS executes SQL queries and manages data storage.

Outputs:

- Data retrieved or modified based on SQL queries.

#### 3.3.2 Payment Gateway Integration APIs

Description: The HMS integrates with payment gateway APIs for processing online fee payments.

Inputs:

- Payment request data, including payment amount and student information.

Processing:

- Communicates with the payment gateway APIs using secure protocols (e.g., HTTPS).
- Receives and processes payment responses from the gateway.

Outputs:

- Payment confirmation or failure status received from the payment gateway.

## 4. System Features

### 4.1 User Registration

**Description:** This feature allows users (hostel staff and students) to create accounts with their personal details.

**Use Cases:**

- **User Registration:** Users provide their first name, last name, username, password, email, and role during registration.
- **Validation:** Input data is validated to ensure data integrity and security.
- **Data Storage:** Valid user details are stored in the database.

**Outputs:**

- Users receive a confirmation message upon successful registration.
- In case of registration failure (e.g., due to duplicate username or invalid email), users are informed with appropriate error messages.

### 4.2 Room Allocation

**Description:** This feature allows hostel staff to assign rooms to registered students based on their preferences and availability.

**Use Cases:**

- **Room Allocation Request:** Staff initiates room allocation by selecting a student and specifying their preferences (e.g., room type).
- **Availability Check:** The system checks room availability and suggests suitable rooms.

- **Room Assignment:** Staff assigns a room to the student, and the room status is updated (e.g., from "Vacant" to "Occupied").

#### Outputs:

- Confirmation of room allocation is sent to the student.
- If no suitable rooms are available, the system informs staff and the student.

### 4.3 Fee Management

**Description:** This feature manages student fees, including tracking payments, generating invoices, and sending reminders.

#### Use Cases:

- **Fee Calculation:** The system calculates fees based on predefined criteria (e.g., room type, duration of stay).
- **Fee Payment:** Students make payments using integrated payment gateways.
- **Payment Tracking:** The system tracks payments and updates the student's fee status.
- **Overdue Notifications:** Automatic reminders are sent to students for overdue payments.

#### Outputs:

- Payment confirmations are sent to students.
- Overdue notifications are sent with details on outstanding fees.

### 4.4 Check-In and Check-Out

**Description:** This feature facilitates the check-in and check-out processes for students.

#### Use Cases:

- **Check-In:** Students provide their ID and room details upon arrival. Staff verifies room availability and records the check-in.
- **Check-Out:** Students request check-out, and staff verifies room condition, updates the status, and records the check-out.

#### Outputs:

- Check-in and check-out confirmations are provided to students.
- Room status is updated to reflect occupancy changes.

## 4.5 Inventory Management

**Description:** This feature tracks hostel inventory and sends restocking alerts when necessary.

**Use Cases:**

- **Inventory Updates:** Staff record inventory changes, including additions and deductions.
- **Inventory Status:** The system maintains up-to-date inventory status.
- **Restocking Alerts:** When inventory levels fall below a certain threshold, the system sends restocking alerts to staff.

**Outputs:**

- Inventory status reports are available for staff.
- Restocking alerts are sent via notifications or emails.

## 4.6 Reporting

**Description:** This feature generates various reports for administrators to monitor hostel occupancy, financials, and inventory.

**Use Cases:**

- **Generate Reports:** Administrators can select report types (e.g., occupancy report, financial report) and specify date ranges.
- **Data Retrieval:** The system retrieves relevant data, generates reports in various formats (e.g., PDF, Excel).

**Outputs:**

- Reports are provided to administrators, allowing them to make informed decisions about hostel management.

## 4.7 Security and Access Control

**Description:** This feature ensures the security of user data and controls user access to the system.

**Use Cases:**



- **Authentication:** Users are required to log in with their usernames and securely hashed passwords.
- **Authorization:** User roles determine their access levels (e.g., student, staff, administrator).
- **Access Control:** Unauthorized access attempts are blocked, and access is limited to specific functionalities based on user roles.

#### Outputs:

- Users receive access denied messages if they attempt unauthorized actions.
- User sessions are secured with authentication tokens.

## 5. Other Non-Functional Requirements

### 5.1 Performance

#### 5.1.1 Response Time

**Description:** The system should provide timely responses to user interactions, ensuring a smooth and responsive user experience.

#### Requirement:

- Critical functions (e.g., user authentication, room allocation) should have a response time of less than 2 seconds.
- Non-critical functions (e.g., generating reports) should provide responses within 5 seconds.

#### 5.1.2 Scalability

**Description:** The system should be scalable to accommodate potential growth in the number of users and data volume.

#### Requirement:

- The system should handle a minimum of 100 concurrent users without significant performance degradation.
- Scalability planning should allow for future expansion of the database and infrastructure to support increased load.

## 5.2 Usability

### 5.2.1 User Interface

**Description:** The user interface (UI) should be intuitive and user-friendly to minimize user training requirements.

**Requirement:**

- The UI should follow standard design conventions, making it easy for users to navigate and perform tasks.
- User training materials, including documentation and tutorials, should be provided to aid users in using the system effectively.

## 5.3 Reliability

### 5.3.1 Availability

**Description:** The system should be available to users with minimal downtime for maintenance or unexpected failures.

**Requirement:**

- The system should aim for 99.9% uptime, excluding scheduled maintenance windows.
- Maintenance windows, if required, should be scheduled during off-peak hours to minimize disruption.

## 5.4 Security

### 5.4.1 Data Security

**Description:** The system should ensure the confidentiality, integrity, and availability of user data.

**Requirement:**

- User data should be encrypted during transmission and storage using industry-standard encryption protocols.
- Passwords should be securely hashed and stored.
- Access to sensitive data should be restricted based on user roles, following the principle of least privilege.

## 5.5 Compatibility

### 5.5.1 Browser Compatibility

**Description:** The system should be compatible with a range of web browsers to accommodate user preferences.

**Requirement:**

- The system should support the latest versions of major web browsers, including Chrome, Firefox, Safari, and Edge.
- Browser-specific issues should be addressed promptly to ensure consistent functionality.

## 5.6 Data Backup and Recovery

### 5.6.1 Regular Backups

**Description:** Regular backups of the system data should be performed to prevent data loss in case of hardware failures or other emergencies.

**Requirement:**

- Daily backups of the database should be scheduled, with backup data stored securely off-site.
- A backup retention policy should be established to manage backup data.

## 5.7 Compliance

### 5.7.1 Regulatory Compliance

**Description:** The system should comply with relevant data protection and privacy regulations.

**Requirement:**

- The system should adhere to data protection regulations such as GDPR, including providing mechanisms for user consent and data deletion requests.
- Regular audits and compliance checks should be conducted to ensure ongoing adherence to regulations.

## 6.Other Requirements

### Appendix A: Terminology/ Glossary / Definition List

#### A.1 Terms and Definitions

- **HMS (Hostel Management System)**: The software application designed to automate and streamline hostel management processes.
- **SRS (Software Requirements Specification)**: A document that outlines the functional and non-functional requirements of the HMS.
- **UI (User Interface)**: The graphical or command-line interface through which users interact with the system.
- **DBMS (Database Management System)**: Software used to manage the storage and retrieval of data in the system.
- **API (Application Programming Interface)**: A set of rules and protocols that allows different software applications to communicate with each other.
- **GDPR (General Data Protection Regulation)**: European Union data protection and privacy regulation.

#### A.2 Acronyms and Abbreviations

- **UI**: User Interface
- **DBMS**: Database Management System
- **API**: Application Programming Interface
- **GDPR**: General Data Protection Regulation
- **HTTPS**: Hypertext Transfer Protocol Secure

Appendix B: To be determined.

## **Experiment-2**

**Aim:** Understanding an SRS.

# **SOFTWARE REQUIREMENTS SPECIFICATION**

**HMS**

VERSION 1.0

SEPTEMBER 12,2023

HOSTEL MANAGEMENT SYSTEM

# **TABLE OF CONTENT**

<b>1. INTRODUCTION</b>	<b>01</b>
1.1. PURPOSE	01
1.2. SCOPE	01
1.3. DEFINATION, ACRONYMS AND ABBREV	01
1.4. DOCUMENT CONVENTIONS	01
1.5. REFERENCES	02
1.6. OVERVIEW	02
<b>2. OVERALL DESCRIPTION</b>	<b>03</b>
2.1. PRODUCT PERSPECTIVE	03
2.2. PRODUCT FUNCTION	03
2.3. USER CHARACTERSITICS	04
2.4. DESIGN/IMPLEMENTATION CONSTRAINTS	04
2.5. ASSUMPTIONS AND DEPENDENCIES	04
<b>3. EXTERNAL INTERFACE REQUIREMENT</b>	<b>05</b>
3.1. USER INTERFACE	05
3.2. HARDWARE INTERFACE	06
3.3. SOFTWARE INTERFACE	07
<b>4. SYSTEM FEAUTRES</b>	<b>08</b>
4.1. USER REGISTRATION	08
4.2. ROOM ALLOCATION	08
4.3. FEE MANAGEMENT	09
4.4. CHECK IN AND OUT	09
4.5. INVENTORY MANAGEMENT	10
4.6. REPORTING	10
4.7. SECURITY AND ACCESS CONTROL	10
<b>5. OTHER NON-FUNCTIONAL REQUIREMENT</b>	<b>11</b>
5.1. PERFORMANCE	11
5.2. USABILITY	12
5.3. RELIABILITY	12
5.4. SECURITY	12
5.5. COMPATIBLITY	13
5.6. DATA BACKUP AND RECOVERY	13
5.7. COMPILANCE	13
<b>6. OTHER REQUIREMENT</b>	<b>14</b>
6.1. APPENDIX A	14
6.2. APPENDIX B	14