COMSATS University Islamabad, Abbottabad Campus

Department of Computer Science

CSC392 Object-Oriented Software Engineering

**Project Proposal**

**HOSTEL MANAGEMENT SYSTEM**

|  |
| --- |
| **Group Members** |
| FA21-BSE-015  Hanzla Nouman |
| FA21-BSE-019  Laiba Binta Tahir |
| FA21-BSE-027  Muhammad Irfan |
| FA21-BSE-080  Arfah Ali |

…

Table of Contents

[CHAPTER 1 PROJECT PROPOSAL 4](#_Toc134130007)

[Introduction 4](#_Toc134130008)

[Vision and Business Case 5](#_Toc134130009)

[Use-Case Model 6](#_Toc134130010)

[Supplementary Specification 7](#_Toc134130011)

[Glossary 8](#_Toc134130012)

[Risk List & Risk Management Plan 8](#_Toc134130013)

[CHAPTER 2: USE CASES 10](#_Toc134130014)

[Use Case Diagram 10](#_Toc134130015)

[Use Cases Distribution 11](#_Toc134130016)

[Brief Level Use Cases 12](#_Toc134130017)

[Laiba binta tahir (FA21-BSE-019 – 4A) 12](#_Toc134130018)

[Use Case: login. 12](#_Toc134130019)

[Use Case: Add student . 13](#_Toc134130020)

[Use Case: view student. 13](#_Toc134130021)

[Use Case: delete student. 13](#_Toc134130022)

[Use Case: add mess menu. 13](#_Toc134130023)

[Use Case: view mess menu. 14](#_Toc134130024)

[Use Case: logout. 14](#_Toc134130025)

[Arfah Ali (FA21-BSE-080 – 4A) 14](#_Toc134130026)

[Use Case: View Incidents. 14](#_Toc134130027)

[Use Case: Apply for Hostel. 14](#_Toc134130028)

[Hanzla Nouman (FA21-BSE-015) 14](#_Toc134130029)

[Use Case: Allocate Room 15](#_Toc134130030)

[Use Case: View Rooms 15](#_Toc134130031)

[Muhammad Irfan (FA21-BSE-027) 15](#_Toc134130032)

[Use Case: Add Warden 15](#_Toc134130033)

[Use Case: Delete Warden 15](#_Toc134130034)

[Fully Dressed Use Cases 17](#_Toc134130035)

[Laiba binta tahir (FA21-BSE-019 – 4A) 17](#_Toc134130036)

[Use Case: Login. 17](#_Toc134130037)

[Use Case: Add student. 18](#_Toc134130038)

[Use Case: view Student. 20](#_Toc134130039)

[Use Case: Delete Student. 21](#_Toc134130040)

[Use Case: add mess menu. 23](#_Toc134130041)

[Use Case: view mess menu. 24](#_Toc134130042)

[Use Case: logout. 26](#_Toc134130043)

[Arfah Ali (FA21-BSE-080 – 4A) 27](#_Toc134130044)

[Use Case UC1: Apply for Hostel. 27](#_Toc134130045)

[Use Case UC1: View Incidents. 29](#_Toc134130046)

[Hanzla Nouman (FA21-BSE-015) 31](#_Toc134130047)

[Use Case UC1: Allocate Room 31](#_Toc134130048)

[Use Case UC2: View Rooms 32](#_Toc134130049)

[Muhammad Irfan (FA21-BSE-027) 33](#_Toc134130050)

[Use Case UC1: Add Warden 33](#_Toc134130051)

[Use Case UC2: Delete Warden 35](#_Toc134130052)

[PROTOTYPES 37](#_Toc134130053)

[CHAPTER 3: DOMAIN MODEL 41](#_Toc134130054)

[Domain Model 41](#_Toc134130055)

[CHAPTER 4: SYSTEM SEQUENCE DIAGRAM (SSD) 41](#_Toc134130056)

[**System Sequence Diagram** 42](#_Toc134130057)

[**Arfah Ali (FA21-BSE-080-4A)** 42](#_Toc134130058)

# CHAPTER 1 PROJECT PROPOSAL

## Introduction

1.1 PROJECT OVERVIEW

The hostel management system is web-based software to provide university students accommodation to the university hostel more efficiently. This project also keeps details of the hostellers and applied students. It is headed by multiple administrators . This document is intended to minimize human works and make hostel allocation an easier job for students and hostel authorities by providing online application for hostel, automatically select the students from the waiting list and mess calculation, complaint registration, etc. Students will get approval notification in their mails. Hostellers can view hostel fee and mess menu by login into the online system.

1.2 PROBLEM STATEMENT

There are a lot of drawbacks in keeping and maintaining a hostel. Especially with a manual system. Since most hostels are being run by only one hostel manager, the number of students in a room are sometimes not known by the warden.

This particular project deals with the problems on managing a hostel and avoids the problems which occur when carried manually Identification of the drawbacks of the existing system leads to the designing of computerized system that will be compatible to the existing system with the system which is more user friendly and more GUI oriented.

We can improve the efficiency of the system, thus overcome the drawbacks of the existing system.

1.3 OBJECTIVES

• To make it easier for data collection, storage and referencing reliable.

• To maintain the students as hostellers and waiting list students separately.

• To process allotment list.

## 

## Vision and Business Case

***Vision:***

Our hostel management system aims to provide a streamlined and efficient solution for managing hostel operations, improving the student experience, and increasing overall productivity. By leveraging modern technologies, we seek to transform hostel management into an automated and user-friendly process that benefits both students and administrators alike.

***Business Case:***

Hostel management is a complex and time-consuming process that requires significant resources and effort to manage effectively. Traditional methods of managing hostels often involve manual processes such as paper-based record keeping, which can be error-prone, time-consuming, and difficult to manage.

Our hostel management system seeks to address these challenges by providing a comprehensive solution that automates many of the tasks involved in hostel management. This includes features such as room allocation, meal plans, student profiles, visitor management, and maintenance tracking. By automating these tasks, we aim to save time and increase the efficiency of hostel operations.

Additionally, our system includes features that enhance the student experience, such as the ability to view their room details, meal plans, and other relevant information through a user-friendly interface. This not only improves the overall student experience but also reduces the workload of administrators by reducing the number of inquiries and requests they receive from students.

Overall, our hostel management system aims to provide a cost-effective and efficient solution for managing hostel operations while also improving the student experience. By leveraging modern technologies and automation, we believe that our system will help hostels to run more smoothly, improve productivity, and ultimately increase profitability.

## Use-Case Model

Functional Requirements for Hostel Management System

* Allocaton /Booking
* record reservations
* record the student first name
* record the student’s last name
* record the number of students
* record the room number
* display the default room fee display whether or not the room is guaranteed.
* generate a unique confirmation number for each.
* record the expected check-in date and time
* The system shall record the expected checkout date and time
* The system shall record customer feedback

2.  Food

* The system shall track all meals purchased in the hotel .
* The system shall record payment and payment type for meals
* The system shall bill the current room if payment is not made at time of service
* The system shall accept reservations for services.

3. Management

* display the hotel occupancy for a specified period of time (days; including past, present, and future dates).
* display projected occupancy for a period of time (days).
* display room revenue for a specified period of time (days).
* display food revenue for a specified period of time (days).
* display an exception report, showing where default room and food prices have been overridden
* allow for the addition of information, regarding rooms, rates, menu items, prices, and user profiles
* allow for the deletion of information, regarding rooms, rates, menu items, prices, and user profiles
* allow for the modification of information, regarding rooms, rates, menu items, prices, and user profiles
* allow managers to assign user passwords

## Supplementary Specification

There are a lot of software requirements specifications included in the non-functional requirements of the Hostel Management System, which contains various processes, namely Security, Performance, Maintainability, and Reliability.

**Security:**

● user Identification: The system needs the user to recognize herself or himself using the phone.

● Logon ID: Any users who make use of the system need to hold a Logon ID and password.

● Modifications: Any modifications like insert, delete, update, etc. for the database can be synchronized quickly and executed only by the ward administrator.

● Front Desk Staff Rights: The staff at the front desk can view any data in the Hostel Management system.

● Administrator rights: The administrator can view as well as alter any information in the Hostel Management System.

**Performance:**

● Response Time: The system provides acknowledgment in just one second once the 'user's information is checked.

● Capacity: The system needs to support at least 1000 people at once.

● User-Interface: The user interface acknowledges within five seconds.

● Conformity: The system needs to ensure that the guidelines of the Microsoft accessibilities are followed.

**Maintainability:**

● Back-Up: The system offers efficiency for data backup.

● Errors: The system will track every mistake as well as keep a log of it.

**Reliability:**

● Availability: The system is available all the time.

## Glossary

|  |  |
| --- | --- |
| *HMS* | Hostel management system |
| *DFD* | Data flow diagram |
| *UC* | Use case diagram |
|  |  |

## 

## Risk List & Risk Management Plan

**Risk List:**

***Technical Risks*** *-* These risks include system failures, software bugs, and other technical issues that may disrupt the system's operations or cause data loss.

***Security Risks*** *-* These risks include data breaches, unauthorized access to the system, and other security threats that may compromise the system's integrity and the safety of students' information.

***Human Risks*** *-* These risks include human error, misconduct, and other issues that may arise from the actions of administrators or other users of the system.

***Operational Risks*** *-* These risks include issues related to the management of the hostel, such as room allocation, billing and payments, maintenance tracking, and other operations that may affect the system's performance.

**Risk Management Plan:**

***Technical Risks*** *-* To mitigate technical risks, the system will be regularly tested for bugs and other technical issues. A backup system will also be implemented to ensure that data is not lost in case of a system failure.

***Security Risks*** *-* To mitigate security risks, the system will be protected by advanced security protocols such as encryption, firewalls, and regular vulnerability testing. Access to the system will be restricted to authorized users only, and data will be regularly backed up and secured.

***Human Risks*** *-* To mitigate human risks, the system will be designed to limit the potential for user error, such as providing clear and intuitive user interfaces and requiring authorization for critical functions. Administrators will also be required to undergo training and follow strict protocols to ensure the safety and integrity of student data.

***Operational Risks*** *-* To mitigate operational risks, the system will be designed to automate and streamline hostel operations, such as room allocation and maintenance tracking. The system will also provide detailed analytics and reports to help administrators identify and address operational issues before they become major problems.

Overall, by implementing a comprehensive risk management plan, the hostel management system can minimize the potential for risk and ensure the smooth and efficient operation of the system.

# 

# CHAPTER 2: USE CASES

## Use Case Diagram



## Use Cases Distribution

|  |  |  |
| --- | --- | --- |
| S#. | Group Member | Assigned Use Cases |
| 1 | FA21-BSE-015  Hanzla Nouman | Allocate rooms:   * Add room * View room * Generate fee * Terminate allocation * Assign facilities to students * Booked room |
| 2 | FA21-BSE-019  Laiba Binta Tahir | Login   * Add menu. * View menu.   Manage students.   * Add student * View student * Delete student   Logout |
| 3 | FA21-BSE-027  Muhammad Irfan | Register  Manage warden.   * Add warden. * View warden. * Delete warden. * modify warden.   Manage staff   * Add security * View security * Delete security * Add cook * View cook * Delete cook |
| 4 | FA21-BSE-080  Arfah Ali | Select facilities  Apply for hostel  Add incidents  Submit fee(hostel/mess)  View incidents.  Feedback from students |

## Brief Level Use Cases

### Laiba binta tahir (FA21-BSE-019 – 4A)

Login

* Add menu.
* View menu.

Manage students.

* Add student.
* View student
* Delete student.

Logout

#### Use Case: login.

The user selects the login option in the hostel management system. The user enters their username and password. The system verifies the user's credentials. If the user's credentials are valid, the system logs the user into the system and displays the appropriate dashboard based on their role (student or administrator). If the user's credentials are invalid, the system displays an error message and prompts the user to re-enter their credentials.

#### Use Case: Add student .

The user selects the "Add Student" option from the main menu. The system prompts the user to enter the student's personal information, including name, contact details, and identification information. The user enters the student's information into the system. The system validates the information and confirms that it is complete and accurate. The user selects the student's preferred room type and bed preference. The system checks the availability of the preferred room and allocates a room to the student if available. The user enters the student's fee details and confirms the payment. The system confirms the payment and updates the student's record accordingly. The system displays a confirmation message to the user.

#### Use Case: view student.

The user selects the "View Student" option from the main menu. The system prompts the user to enter the student's ID or name. The user enters the student's ID or name and submits the query. The system retrieves the student's details from the database and displays them in a clear and easy-to-understand format. The user can view the details of the selected student(s) and perform further actions, such as updating their records, contacting them, or generating reports.

#### Use Case: delete student.

The admin logs into the hostel management system and selects the "Delete *student*" option from the main menu. The system retrieves the list of all registered *students* and displays their details, such as name, contact information, on the screen. The admin selects the *student* to delete from the list and confirms the action. The system removes the selected *student* from the system, revokes their access to the system, and updates the records accordingly. The system then displays a confirmation message to the admin.

#### Use Case: add mess menu.

The user selects the "Add Mess Menu" option from the main menu. The system prompts the user to select the week/month for which the menu is to be updated. The user selects the desired week/month. The system retrieves the list of available food items and prompts the user to add new items to the menu for each meal (breakfast, lunch, and dinner) for each day of the selected week/month. The user adds the new food items for each meal/day based on availability, preference, and budget. The system confirms the updated menu and displays it in a clear and easy-to-understand format. The system updates the food stock and preparation plan accordingly.

#### Use Case: view mess menu.

The user selects the "View Mess Menu" option from the main menu. The system displays the mess menu for the current date and time. The user can view the menu and make decisions about their meals.

#### Use Case: logout.

The user selects the "Logout" option from the main menu. The system prompts the user to confirm that they want to log out. The user confirms the action by selecting "Yes." The system terminates the user's session and logs them out of the system. The system displays a confirmation message to the user.

### Arfah Ali (FA21-BSE-080 – 4A)

* View facilities
* Apply for hostel
* Add incidents
* Submit fee(hostel/mess)
* View incidents and reports.
* Feedback from students

#### Use Case: View Incidents.

The user navigates to the "Incidents" section of the Hostel Management System. The system displays a list of all incidents reported in the hostel, sorted by date and time. The user can filter the incidents by date range, location, or severity. The user can view the details of each incident, including the date and time of the incident, location, description, severity level, and status. The user can update the status of the incident if necessary.

#### Use Case: Apply for Hostel.

The student navigates to the "Apply for Hostel" section of the Hostel Management System. The system displays the application form for the student to fill in their personal details, preferred room type, and any other relevant information. The student fills out the application form and submits it to the system. The system checks the eligibility of the student, such as their academic standing, hostel room availability, and any other relevant criteria. The hostel staff reviews the application and verifies the information provided by the student. If the application is approved, the hostel staff assigns a room to the student and notifies them of their room allocation. If the application is rejected, the system notifies the student of the rejection.

### Hanzla Nouman (FA21-BSE-015)

* Allocate Room
  + Allocate the Room to the Student
  + Terminate Allocation
* Add Room
* Delete Room
* View Rooms (Room no, Building, Location etc.)
* Assign Facilities
* Remove Facility

#### Use Case: Allocate Room

The warden logs into the hostel management system and selects the "Allocate Room" option from the main menu. The system displays the list of available rooms, and the warden selects an available room based on the student's preferences and needs. The system allocates the selected room to the student, updates the availability status of the room in the system, and generates a fee receipt for the allocated room. The warden prints the fee receipt and hands it over to the student.

#### Use Case: View Rooms

The warden logs into the hostel management system and selects the "View All Rooms" option from the Rooms Menu. The system displays a list of all the rooms, including allocated and unavailable rooms, along with their room number, room type, and availability status. The warden can scroll through the list and view the details of each room.

### Muhammad Irfan (FA21-BSE-027)

* Manage Wardens
  + Add warden.
  + View wardens
  + Delete warden.
* Manage staff
  + Add security.
  + View security.
  + Delete security
  + Add cook
  + View cook
  + Delete cook

#### Use Case: Add Warden

The admin logs into the hostel management system and selects the "Add Warden" option from the main menu. The system prompts the admin to enter the details of the new warden, such as their name, contact information, and job title. The admin enters the details and confirms the addition of the new warden. The system creates a new user account for the warden with login credentials and grants appropriate access to the system. The admin receives a confirmation message from the system.

#### Use Case: Delete Warden

Delete Warden: The admin logs into the hostel management system and selects the "Delete Warden" option from the main menu. The system retrieves the list of all registered wardens and displays their details, such as name, contact information, and job title, on the screen. The admin selects the warden to delete from the list and confirms the action. The system removes the selected warden from the system, revokes their access to the system, and updates the records accordingly. The system then displays a confirmation message to the admin.

## Fully Dressed Use Cases

### Laiba binta tahir (FA21-BSE-019 – 4A)

#### Use Case: Login.

**Use Case Name:** Login

**Scope:** Hostel Management System

**Primary Actor:** Administrator or Student

Goal in Context: The primary goal of this use case is to allow administrators or students to log into the hostel management system to access their respective features and functionalities.

**Preconditions:**

The user has a valid account with the hostel management system.

The user has a device with internet access.

The user has their login credentials, including their username and password.

**Trigger:**

The user attempts to log into the hostel management system by entering their login credentials.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **User Interaction** | **System Response** |
| 1. User enters their login credentials (username and password). | 1. System validates the user's credentials by checking them against the credentials stored in the database. |
| 2. User clicks the login button. | 2. If the credentials are valid, the system logs the user into the hostel management system and displays the main menu. |
|  | 3. If the credentials are not valid, the system displays an error message indicating that the login attempt has failed. |

**Extensions:**

If the user enters an incorrect username or password, the system will display an error message and prompt the user to re-enter their credentials.

If the user forgets their password, they can request a password reset link or contact the administrator for assistance.

**Alternate Flow:**

If the user does not have a valid account with the hostel management system, they can create an account by clicking the "register" button on the login page.

**Postconditions:**

The user is successfully logged into the hostel management system and has access to their respective features and functionalities.

The system logs the user's login activity for security and audit purposes.

**Exceptions:**

If the system experiences technical issues, the user may not be able to log in. In this case, the user can contact technical support or try again later.

#### Use Case: Add student.

**Use Case Name:** Add Student

**Scope:** Hostel Management System

**Level:** User Goal (Primary)

**Stakeholders and Interests:**

Hostel Staff: Wants to manage student records and information efficiently, including registration, room allocation, and fee payment.

Students: Want to register for accommodation in the hostel and pay the necessary fees.

**Preconditions:**

The user is authorized to access the hostel management system.

The student has expressed interest in registering for accommodation in the hostel.

**Postconditions:**

The student's information is added to the hostel management system.

The student can be allocated a room based on availability and preferences.

The student can pay the necessary fees for accommodation and other services.

**Basic Flow of Events:**

|  |  |
| --- | --- |
| **User Interaction** | **System Response** |
| 1. User selects the option to add a new student. | 1. System prompts the user to enter the student's information, such as name, contact information, room number, and any other relevant details. |
| 2. User enters the student's information. | 2. System validates the information provided by the user to ensure that it is complete and accurate. |
| 3. User confirms the addition of the new student. | 3. If the information provided by the user is valid, the system adds the new student's record to the database and displays a success message to the user. |
|  | 4. The success message includes the student's ID, which is assigned by the system, as well as their name and other relevant details. |
| 4. User reviews the new student's record. | 5. System waits for the user to finish reviewing the new student's record before returning to the main menu. |
|  | 6. If the user selects to add another student, the system repeats the above process. Otherwise, it returns to the main menu. |

**Alternative Flows:**

If the student has already paid the fees, the user can skip the fee payment step and proceed to allocate a room.

If there are no rooms available for the student's preferred room type and bed preference, the system will display alternative options for the user to choose from.

**Exceptional Flows:**

If there is an error in the system while adding the student's information, the system displays an error message to the user and prompts them to try again.

If there is a problem with the payment, the system will display an error message and prompt the user to enter the correct details.

#### Use Case: view Student.

**Use Case Name**: View Student

**Scope**: Hostel Management System

**Level**: User Goal (Primary)

**Stakeholders and Interests:**

Hostel Staff: Want to view the details of a particular student, including their personal information, academic performance, and disciplinary record.

Parents/Guardians: Want to view the details of their child's stay in the hostel, including their room allocation, attendance, and progress report.

Students: Want to view their own details and stay updated on their academic and personal progress.

**Preconditions:**

The user is authorized to access the hostel management system.

The details of the student(s) are available in the system database.

**Postconditions:**

The user can view the details of the selected student(s) in a clear and easy-to-understand format.

The student(s) can access their own details and stay updated on their progress and stay in the hostel.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **User Interaction** | **System Response** |
| 1. User selects the option to view a student. | 1. System displays a list of all the students registered in the hostel. |
| 2. User selects the student to be viewed. | 2. System retrieves the student's record from the database and displays the details to the user. |
|  | 3. System displays the student's ID, name, contact information, room number, and any other relevant details. |
| 3. User reviews the student's information. | 4. System waits for the user to finish reviewing the student's information before returning to the main menu. |
|  | 5. If the user selects to view another student, the system repeats the above process. Otherwise, it returns to the main menu. |

**Alternative Flows:**

If the user enters an invalid or non-existent ID or name, the system displays an error message and prompts the user to try again.

If the user is not sure about the student's ID or name, the system can provide suggestions based on the search query.

**Exceptional Flows:**

If there is an error in the system while retrieving or displaying the student's details, the system displays an error message to the user and prompts them to try again.

If the user does not have permission to view the student's details, the system displays an error message and denies access.

#### Use Case: Delete Student.

**Use Case Name:** Delete student.

**Primary Actor:** Administrator

**Goal in Context**: The primary goal of this use case is to allow the administrator to delete a student from the hostel management system and associated records.

**Preconditions**

The administrator is logged in to the hostel management system.

The administrator has the appropriate privileges to delete a student record.

The student record to be deleted exists in the system.

**Trigger:**

The administrator selects the option to delete a student record from the system.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **User Interaction** | **System Response** |
| 1. User selects option to delete a student. | 1. System checks whether student ID exists. |
| 2. User enters ID of student to be deleted. | 2. If student ID exists, system displays confirmation message and asks for confirmation to delete. |
| 3. User confirms deletion. | 3. If confirmed, system removes student record from database. |
|  | 4. System displays success message to user indicating student has been successfully deleted. |
|  | 5. If student ID does not exist, system displays error message indicating that student could not be found and prompts user to try again. |

**Extensions:**

If the student record to be deleted has pending fees or dues, the system will prompt the administrator to settle the outstanding balance before proceeding with the deletion.

If the student record to be deleted is linked to other records, such as room allocations or meal plans, the system will prompt the administrator to reassign these records to another student or delete them along with the student record.

**Alternate Flow:**

If the administrator accidentally selects the wrong student record for deletion, they can cancel the deletion and select the correct record.

**Postconditions:**

The student record and associated records are successfully deleted from the hostel management system.

The system logs the administrator's action for audit purposes.

**Exceptions:**

If the administrator does not have the appropriate privileges to delete a student record, the system will display an error message and prevent the deletion from proceeding.

If the student record to be deleted does not exist in the system, the system will display an error message and prevent the deletion from proceeding.

#### Use Case: add mess menu.

**Use Case Name:** Add Mess Menu

**Scope**: Hostel Management System

**Level**: User Goal (Primary)

**Stakeholders and Interests:**

Hostel Staff: Want to add new food items to the mess menu for the week/month.

Students: Want to have a variety of food options in the mess.

**Preconditions:**

The user is authorized to access the hostel management system.

The mess has been stocked with the necessary food items.

**Postconditions:**

The new food items are added to the mess menu for the selected week/month.

The updated menu is displayed to the students in a clear and easy-to-understand format.

The mess staff can manage food stock and preparation efficiently based on the updated menu.

**Main Success Scenario**

|  |  |
| --- | --- |
| 1. User selects "Add Menu" option from the main menu of the hostel management system. | System displays the "Add Menu" form for the user to fill in. |
| 2. User fills in the form with the details of the new menu, such as menu items, prices, and availability. | System validates the input and checks for any conflicts with existing menus or menu items. |
| 3. User submits the form to the system. | System adds the new menu to the system and updates the database. System displays a success message to the user, confirming that the new menu has been added to the system. |
| 5. User can now view, edit, or delete the new menu from the system. |  |

**Alternative Flows:**

If a food item is not available, the user can add an alternative option.

If the user is not sure about the food items or quantities to add, the system can provide suggestions based on previous menus and student feedback.

**Exceptional Flows:**

If there is an error in the system while updating the menu, the system displays an error message to the user and prompts them to try again.

If there is a problem with the food stock or preparation, the system will notify the mess staff and suggest corrective action.

#### Use Case: view mess menu.

**Use Case Name:** View Mess Menu

**Scope:** Hostel Management System

**Level:** User Goal (Primary)

**Stakeholders and Interests:**

Hostel Staff: Wants to create and update the mess menu, communicate changes to the students, and manage meal planning and inventory.

Students: Want to view the mess menu to plan their meals and make dietary decisions.

**Preconditions:**

The user is authorized to access the hostel management system.

The mess menu has been created and updated by the hostel staff.

The mess menu is accessible through the hostel management system.

**Postconditions:**

The user can view the mess menu for the specified date and time.

The user can make informed decisions about their meals based on the mess menu.

The hostel staff can manage the meal planning and inventory based on the mess menu.

**Main Success Scenario**

|  |  |
| --- | --- |
| 1. User selects "View Menu" option from the main menu of the hostel management system. | System retrieves the list of all available menus and displays them on the screen. |
| 2. User selects the menu to be viewed from the list. | System displays the details of the selected menu, including its name, description, and the items included in the menu. |
| 3. User reviews the details of the menu. | System allows the user to navigate back to the list of available menus or exit the menu viewing screen. |

**Alternative Flows:**

If the user wants to view the mess menu for a different date or time, they can select the desired date and time from the menu options.

If the user wants to see more details about a specific dish, they can click on the dish name to view the ingredients and preparation method.

#### Use Case: logout.

**Use Case Name:** logout.

**Scope:** Hostel Management System

**Level:** User Goal (Primary)

**Stakeholders and Interests:**

Hostel Staff: Wants to ensure the security and confidentiality of the data within the system and prevent unauthorized access to the system.

System Administrator: Wants to ensure that the system operates smoothly and efficiently.

**Preconditions:**

The user is currently logged in to the hostel management system.

The user has completed their tasks within the system and wants to log out.

The user is an authorized user of the hostel management system.

**Postconditions:**

The user is logged out of the hostel management system.

The user's session is terminated, and they can no longer access the system until they log in again.

The system is secure, and the data within the system is protected from unauthorized access.

**Main Success Scenario**

|  |  |
| --- | --- |
| 1. User selects "Logout" option from the main menu. | System displays a confirmation message to the user, asking them to confirm that they want to log out. |
| 2. User confirms that they want to log out. | System revokes the user's access to the system and logs them out of the system, returning to the login screen. |

**Alternative Flows:**

If the user selects "No" when prompted to confirm their logout, the system returns to the previous screen without logging the user out.

If the user is inactive for a certain period, the system may automatically log the user out to ensure the security of the system.

**Exceptional Flows:**

If there is an error in the system while the user is logging out, the system displays an error message to the user and prompts them to try again. If the issue persists, the user can contact the system administrator for assistance.

### Arfah Ali (FA21-BSE-080 – 4A)

#### Use Case UC1: Apply for Hostel.

**Use Case Name:** Apply for Hostel

**Scope:** Hostel Management System

**Primary Actor:** Student

**Stakeholders and Interests:**

Student: Wants to apply for a hostel room.

Hostel warden: Wants to receive and process student applications. Wants to ensure that hostel rooms are allocated efficiently.

University: Wants to ensure that all eligible students have access to hostel rooms.

**Preconditions:**

The student has an active university registration.

The student is logged into the Hostel Management System.

The student has not yet been allocated a hostel room (if allocation is done by the system).

The hostel has vacant rooms or upcoming availability of rooms.

The student has the necessary information required to fill the hostel application form.

**Postconditions:**

The student has submitted an application for hostel accommodation.

**Main Success Scenario:**

| **User Interaction** | **System Response** |
| --- | --- |
| 1. Student selects "Apply for Hostel" from the dashboard | 2. System displays the hostel application form. |
| 3. Student fills out the application form with their personal information and room preferences. | 4. System validates the information and confirms successful submission |
| 5. System displays a confirmation message that the application has been submitted | 6. System sends a notification to the user's registered email and/or mobile number |
| 7. Student can view the status of their application in the dashboard | 8. System updates the application status as it progresses through the approval process |
| 9. The warden asks the system to generate fee slip. | 10. The system, and generates a fee receipt for the allocated room. |
| The warden prints the fee receipt and hands it over to the student. | 12. System notifies the student with allocated room and fee to be submitted. |

**Extensions (or Alternative Flows):**

*A1. The student has already been allocated a hostel room.*

The system displays a message indicating that the student has already been allocated a room and cannot apply for another.

The use case ends.

*A2. The hostel has no vacant rooms or upcoming availability.*

The system displays a message indicating that there are no rooms available and prompts the student to try again at a later time.

The use case ends.

*A3. The student does not fill out the application form correctly or completely.*

The system displays an error message and prompts the student to correct the mistakes or fill out the missing information.

The use case continues from step 3.

*A4. The hostel staff rejects the student's application.*

The hostel staff provides a reason for the rejection and notifies the student.

The use case ends.

*A5. The hostel staff needs more information from the student to process the application.*

The hostel staff contacts the student and requests the additional information.

#### Use Case UC1: View Incidents.

**Use Case Name:** View Incidents

**Scope:** Hostel Management System

**Primary Actor:** Hostel Staff

**Stakeholders and Interests:**

Hostel Staff: Wants to view a list of incidents reported in the hostel.

Hostel Manager: Wants to ensure that all incidents are recorded and managed properly.

Hostel Residents: Expect that incidents are handled effectively and efficiently.

**Preconditions:**

The user is logged into the Hostel Management System.

The user has the necessary permissions to view incidents.

Incidents have been reported and recorded in the system.

**Postconditions:**

The user has a list of all incidents reported in the hostel.

**Main Success Scenario:**

| **User Interaction** | **System Response** |
| --- | --- |
| 1. Warden selects "View Incidents" from the dashboard | 2. System displays a list of all incidents reported in the hostel. |
| 3. Warden selects a specific incident to view details. | 4. System displays the details of the selected incident, including the time, location, and description. |
| 5. Warden can search for specific incidents by date, location, or keyword. | 6. System filters the incident list based on the user's search criteria. |
| 7. If the incident is resolved, the system updates the status to "closed" and provides a summary of the resolution | 8. System sends a notification to the user with the resolution details |

**Extensions (or Alternative Flows):**

*A1. No incidents have been reported in the hostel.*

The system displays a message indicating that there are no incidents to view.

*A2. The user does not have permission to view incidents.*

The system displays an error message indicating that the user does not have the necessary permissions.

*A3. The user encounters an error while viewing incidents.*

The system displays an error message and prompts the user to try again.

*A4. The user wants to add a new incident.*

The user can navigate to the "Add Incident" section of the Hostel Management System to report a new incident.

*A5. The user wants to export incident data.*

The user can export the incident data to a CSV or Excel file for further analysis**.**

### Hanzla Nouman (FA21-BSE-015)

#### Use Case UC1: Allocate Room

**Use Case Name:** Allocate Room to Student

**Scope**: Hostel Management System

**Level:** User Goal

Primary Actor: Warden

**Stakeholder and Interest:**

* Warden: To allocate the room to the student and generate a fee receipt for the room.
* Student: To get an allocated room and fee receipt for the allocated room.

**Precondition:**

* The warden is logged into the hostel management system.
* The student has completed the necessary application and registration processes.

**Main Success Scenario:**

| **User Interaction** | **System Response** |
| --- | --- |
| 1. The warden selects the "Allocate Room" option from the main menu. | 2. The system retrieves the list of all unallocated rooms and displays them on the screen. |
| 3. The warden selects a room from the list to allocate to the student. | 4. The system checks if the selected room is available and not already allocated to another student. |
| 5. The system prompts the warden to enter the student's name and ID number. | 6. The warden enters the student's name and ID number into the system. |
| 7. The system checks if the entered student information matches the registered student information in the system. | 8. The system allocates the selected room to the student and updates the room status to "Allocated." |
| 9. The system generates a confirmation message with the allocated room number and displays it to the warden. | 10. The warden informs the student of their allocated room. |
|  |  |

**Extensions (Alternative Flows):**

* If there are no available rooms, the system displays a message to the warden indicating that there are no rooms available and prompts the warden to add more rooms to the system.
* If the student changes their mind about the room allocation, the warden can cancel the allocation and select a different room for the student.
* If there are any issues with the fee receipt, the warden can update the details and regenerate the fee receipt.

**Special Requirements:**

* The system should have a list of available rooms.
* The system should have a fee calculation system.

**Technology and Data Variation List:**

* Hostel Management System
* Available Rooms List
* Fee Calculation System

**Open Issues:**

* None.

#### Use Case UC2: View Rooms

**Use Case Name:** View Rooms

**Scope:** Hostel Management System

**Level:** User Goal

**Primary Actor:** Warden

**Stakeholders and Interests:**

* **Warden:** Wants to view all available rooms in the hostel.
* **Students:** Want to know the available rooms in the hostel.

**Preconditions:**

* The Warden must be logged into the Hostel Management System.

**Main Success Scenario:**

| **Actor Action** | **System Response** |
| --- | --- |
| 1. The Warden selects the "View Rooms" option from the main menu. | 2. The system retrieves the list of all available rooms in the hostel. |
| 3. The system displays the list of rooms along with their current occupancy status, room type, and room number. | 4. The Warden can view the list of all available rooms in the hostel. |

**Extensions:**

3a. If no rooms are available in the hostel, the system displays a message stating that all rooms are currently occupied.

| **Actor Action** | **System Response** |
| --- | --- |
| 1. The Warden selects the "View Rooms" option from the main menu. | 2. The system checks the availability of rooms in the hostel. |
| 3. The system displays a message stating that all rooms are currently occupied. | 4. The Warden cannot view the list of available rooms in the hostel. |

**Special Requirements:**

* The system must have access to the database of rooms and their occupancy status.

**Technology and Data Variation List:**

* The Hostel Management System should be accessible via a computer or mobile device.
* The system should display the list of available rooms in a clear and organized manner.

**Open Issues:**

* None.

### Muhammad Irfan (FA21-BSE-027)

#### Use Case UC1: Add Warden

**Use Case Name: Add Warden**

**Scope:** Hostel Management System

**Level:** User Goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

* Admin: Wants to add a new warden to the system.
* Warden: Wants to be added to the system to manage the hostel effectively.

**Preconditions:**

* The admin is logged into the system.
* The admin has the necessary privileges to add a new warden.

**Main Success Scenario:**

| **User Interaction** | **System Response** |
| --- | --- |
| 1. The admin selects the "Add New Warden" option from the main menu. | 2. The system displays a form for adding a new warden on the screen. |
| 3. The admin fills in the details of the new warden, including name, contact information, and job title. | 4. The system validates the input and creates a new warden profile with the entered details. |
| 5. The admin assigns a username and password to the new warden for accessing the system. | 6. The system verifies the uniqueness of the username and password and associates them with the new warden profile. |
| 7. The admin saves the new warden profile to the system. | 8. The system updates the records and displays a success message to the admin. |

**Extensions:**

* 3a. If the admin enters invalid or incomplete information, the system displays an error message and prompts the admin to correct the input.
* 5a. If the admin chooses a username or password that is already taken, the system displays an error message and prompts the admin to select a different one.

**Special Requirements:**

* The system must ensure that each warden has a unique username and password.
* The system must provide appropriate validation and error messages for input fields.

**Technology and Data Variation List:**

* The system can store wardens' data in a database or other storage system.
* The system can use various forms of validation to ensure data quality and consistency.

**Open Issues:**

* How should the system handle case where a warden's information needs to be updated or deleted?
* Should the system provide any additional functionality or tools for managing wardens?

#### 

#### Use Case UC2: Delete Warden

**Use Case Name:** Delete Warden

**Scope:** Hostel Management System

**Level:** User goal

**Primary Actor:** Admin

**Stakeholders and Interests:**

* Admin: Wants to delete a warden from the system.
* Warden: May lose access to the system if deleted.

**Preconditions:**

* The admin must be logged into the hostel management system.
* There must be at least one warden registered in the system.

**Main Success Scenario:**

| **User Interaction** | **System Response** |
| --- | --- |
| 1. The admin selects the "Delete Warden" option from the main menu. | 2. The system displays a list of all registered wardens. |
| 3. The admin selects the warden to delete from the list. | 4. The system displays a confirmation message asking the admin to confirm the deletion. |
| 5. The admin confirms the deletion. | 6. The system removes the selected warden from the system, revokes their access to the system, and updates the records accordingly. |
| 7. The system displays a success message to the admin. | 8. The use case ends. |

**Extensions:**

* 3a. If there are no wardens registered in the system, the system displays a message to the admin stating that there are no wardens to delete.
* 5a. If the admin cancels the deletion, the system returns to step 1 of the main success scenario.

**Special Requirements:**

* The system must keep a record of all deleted wardens.
* The system must ensure that the deleted warden's access to the system is revoked immediately.

**Technology and Data Variation List:**

* The system must be compatible with different web browsers and operating systems.
* The system must be able to handle many registered wardens and deleted records.

**Open Issues:**

* How will the system ensure that the deleted warden's personal information is securely deleted from the system?
* How will the system handle any errors that occur during the deletion process?

## PROTOTYPES









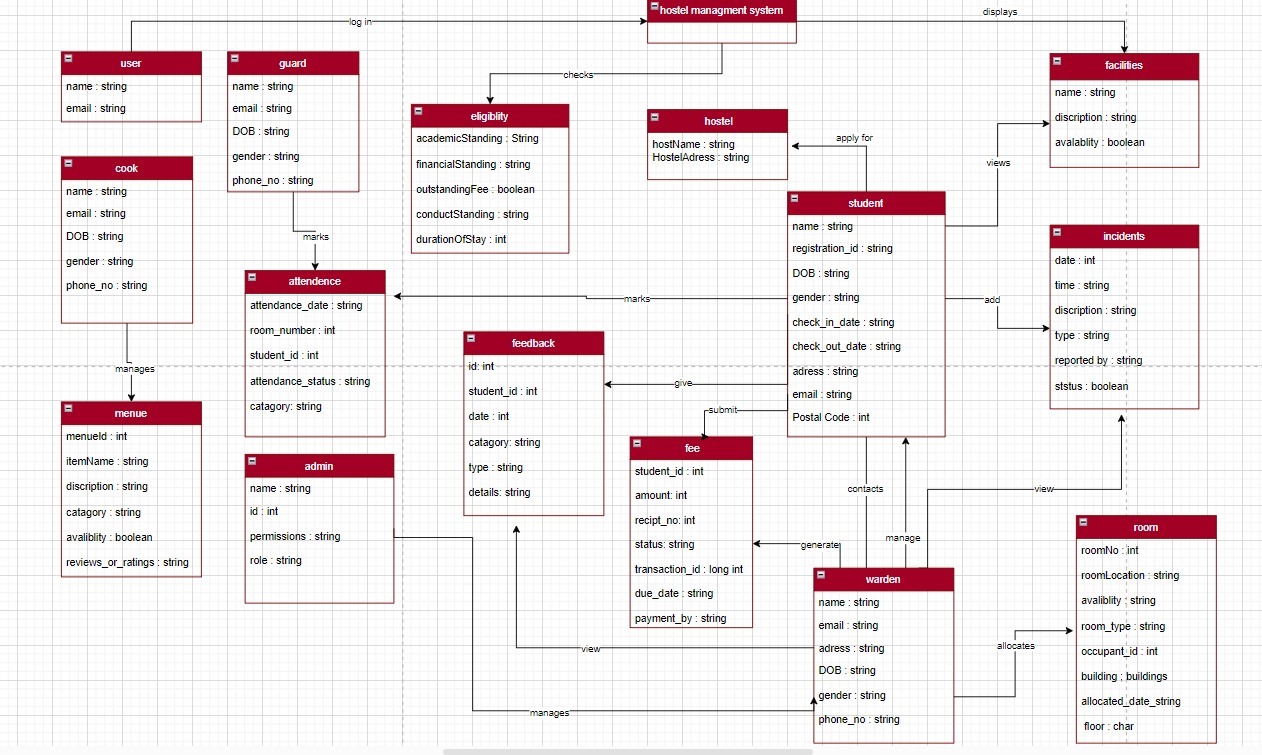






# CHAPTER 3: DOMAIN MODEL

## Domain Model

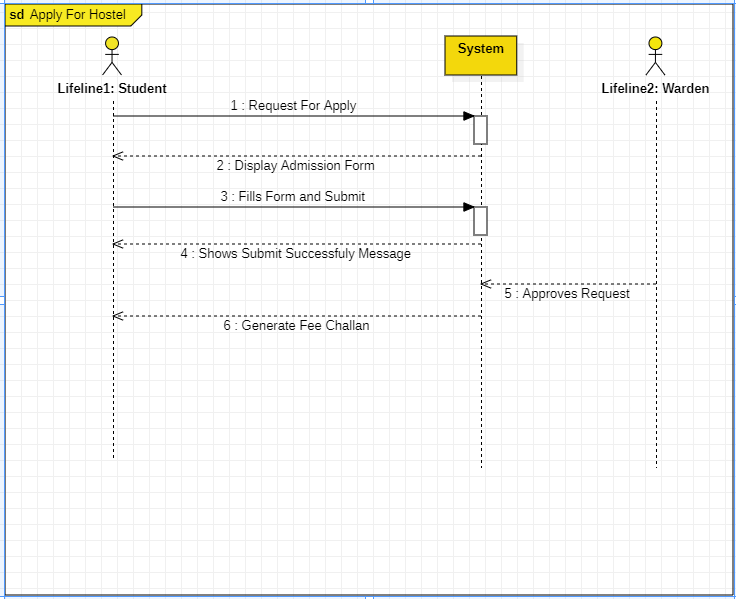


# CHAPTER 4: SYSTEM SEQUENCE DIAGRAM (SSD)

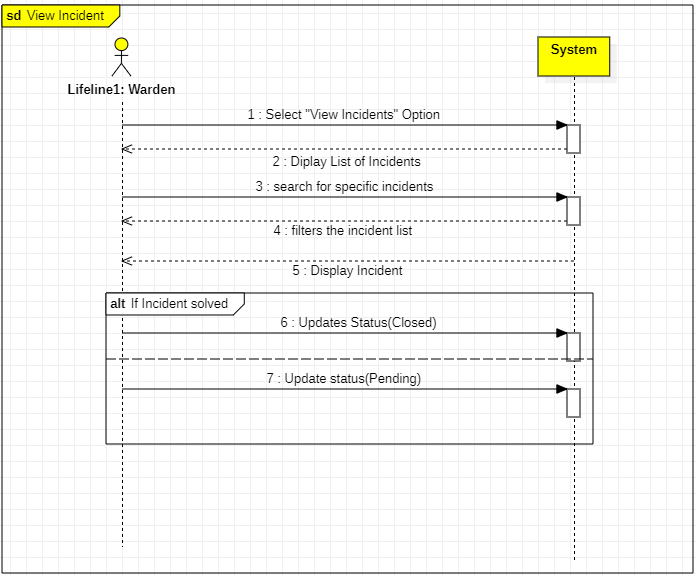
## **System Sequence Diagram**

### **Arfah Ali (FA21-BSE-080-4A)**

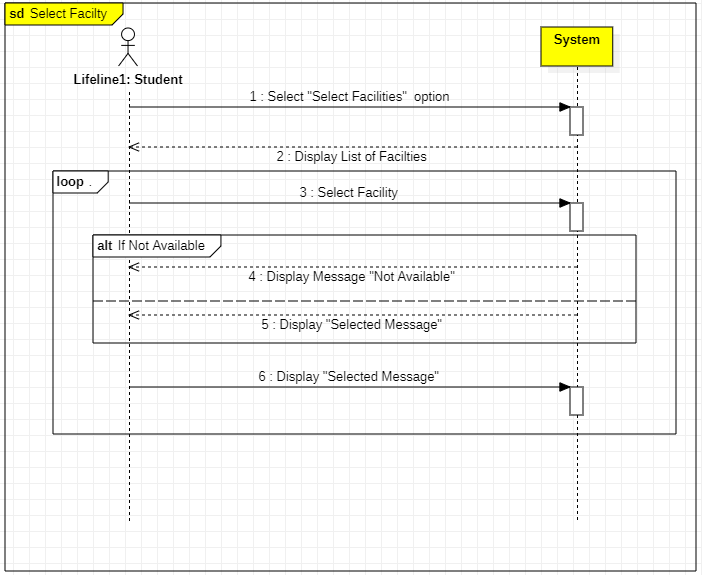
Scenario: Apply For Hostel



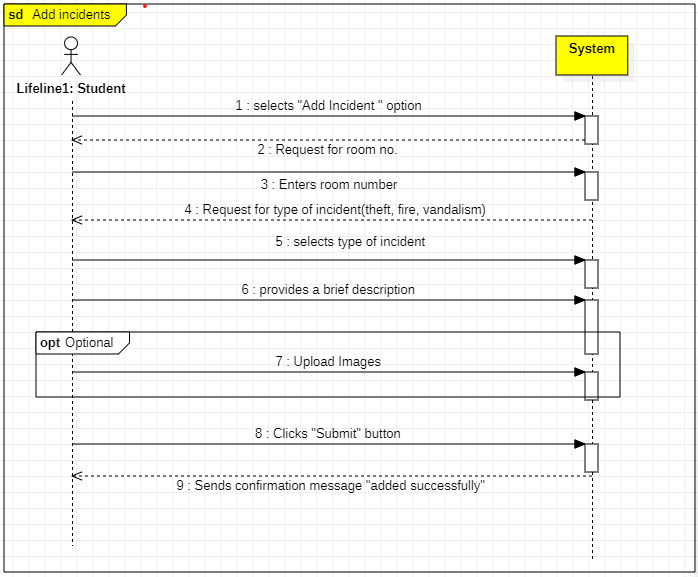
Scenario: View Incidents



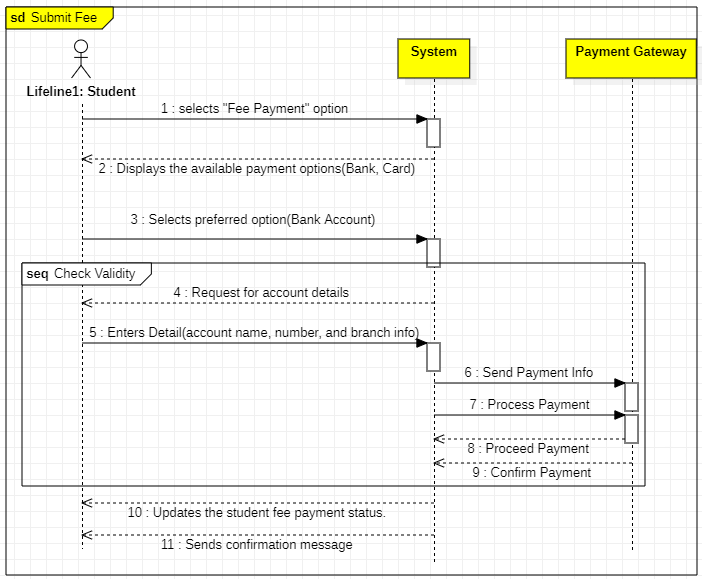
Scenario: Select facilities



Scenario: Add incidents



Scenario: Submit fee (hostel/mess)



Scenario: Feedback from students

