# Software Requirements Specification

for

# HallBuddy

Version 1.1

Prepared by

Group #: 12 Group Name: Code Monk

| | | |
|---|---|---|
| Apoorv Tandon | 220192 | apoorvt22@iitk.ac.in |
| Mridul Gupta | 220672 | mridulg22@iitk.ac.in |
| Rohan Batra | 210868 | rohanb21@iitk.ac.in |
| Tanishq Maheshwari | 221128 | tanishqm22@iitk.ac.in |
| Taneshq Zendey | 221123 | taneshq22@iitk.ac.in |
| Krutuparna Paranjape | 210536 | krutuparna21@iitk.ac.in |
| Ritesh Hans | 220893 | riteshhans22@iitk.ac.in |
| Mrdul Agarwaal | 210632 | mrdula21@iitk.ac.in |
| Ayush | 220259 | ayushs22@iitk.ac.in |
| Samarpan Verma | 220943 | samarpanv22@iitk.ac.in |

Course: CS253

Mentor TA: Vaibhav Tanvar

Date: 26.01.2024

# Index

## Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
|---|---|---|---|
| 1.1 | Apoorv Tandon<br>Ayush<br>Mridul Gupta<br>Mrdul Agarwaal<br>Krutuparna Paranjape<br>Rohan Batra<br>Samarpan Verma<br>Ritesh Hans<br>Tanishq Maheshwari<br>Taneshq Zendey<br>Ayush | First Draft. This version provides the details of the software and the vision of the team regarding its functionalities. | 25/01/24 |

# 1 Introduction

## 1.1 Product Scope

The proposed software is a comprehensive Hostel Management System designed to streamline and enhance the operational efficiency of a hostel facility. This system provides a user-friendly interface with two distinct login functionalities: Resident login for hostel residents and Administrator login for Hall Administration for managing the hostel's day-to-day operations.

The Hostel Management System offers a comprehensive solution with the following features:

- Two types of login functionalities, i.e. admin and resident to regulate administrative access.
- Streamlined process for residents to book guest rooms online. They will be able to check availability and request bookings along with request status updates.
- Users can access and update their room cleaning records, contact cleaning staff to schedule room cleaning. Admin can also monitor cleaning records.
- Quick complaint redressal mechanism for residents. Residents can view already lodged complaints, upvote them and create new complaints. Admin can manage the complaints by filtering and sorting them by tags. They can also update the complaint status and notify it to the users.
- Real time comprehensive breakdowns of hostel dues under heads like electricity bill, mess charges, fines, etc. for users and admin.
- Quick and easy access to detailed hostel map, Fire-Exit Path, emergency contacts, warden details, Hall SIS Guard’s contact details and security information.
- Enables admin to make important announcements and broadcast messages for effective communication with all users.
- Users can see the catalogue of available items in the various hostel shops and contact the shopkeeper. Admin may update the catalogue on request of shopkeeper.

There are several benefits associated with the product:

- Facilitates quick and effective communication between users and admin. Admin can easily get reviews and suggestions from users regarding services like mess, cleaning, maintenance, etc.
- Admins have access to various kinds of data which helps with staff performance review, effective complaint redressal, etc.
- Offers a centralized platform that streamlines processes, saving time and resources for both users and admin.
- Makes the hostel services more accessible to the users by making processes quick, efficient, significantly more transparent and minimizing manual tasks.
- Allows administrators to manage hostel maps and fire exit pathways, enhancing emergency preparedness within the hostel.
- Saves a huge load of paperwork, thus reducing red-tapism, and increasing efficiency.

## 1.2 Intended Audience and Document Overview

### 1.2.1 Intended Viewers:

The SRS document caters to developers, project managers, administrators, users, testers, and documentation writers. Developers would use it for technical details, project managers for scope and resources, administrators for system functionalities and tools in hostel management. Users would understand the interface and tailored functionalities. Testers may verify against requirements, and documentation writers can use it as a reference for manuals and technical documentation.

### 1.2.2 Sections and Organization:

The document is organized into sections for a clear understanding. It begins with an Introduction providing an overview, followed by the Overall Description, Specific Requirements, Other Non-Functional Requirements, and Other Requirements, each addressing specific aspects of the system.

### 1.2.3 Sequence for Important Sections:

- For developers or testers, the recommended reading sequence includes starting with an overview (2.1), functionality (2.2), design and implementation (2.3), interfaces (3.1), and functional requirements (3.2).
- On the other hand, users are advised to concentrate on the scope (1.1), overview (2.1), and functionality (2.2). Following this, they should delve into specific requirements, navigating through sections 3.1.1, 3.1.3, and 3.2.

## 1.3 Definitions, Acronyms and Abbreviations

- Admin – Hostel administration (warden, accountant, HEC, etc.)
- GUI-Graphical User Interface
- HEC - Hall Executive Committee (Hostel Administration committee)
- HTML-Hyper Text Markup Language: the standard markup language for documents designed to be displayed in a web browser.
- JS-Java Script
- OTP - One Time Password
- Resident – Hostel residents

- SIS - Security and Intelligence Services
- SRS – Software Requirement Specification
- Stakeholder: Any person with an interest in the project who is not a developer.
- UI/UX – User Interface/User Experience: The point of human-computer interaction and communication in a device.

## 1.4 Document Conventions

**Formatting Conventions:**

- The document is composed using Arial font, size 12, featuring single spacing and 1-inch margins.
- Headings in subsections are underscored for emphasis.
- Terms highlighted in bold within the same font space indicate that explanations are provided in footnotes or separately within the section.
- Bullet point ordering serves as a tool for listing and typesetting.
- Italics are employed to highlight comments.

## 1.5 References and Acknowledgments

The following websites were referred to in the process of making this document:

1. Figma documentation for User Interface Design: https://help.figma.com/hc/en-us/articles/14552804059927-Lesson-4-Document-and-manage-your-system

We'd also like to acknowledge the help of our TA, Mr. Vaibhav Tanvar, for their valuable input in the creation of this document.

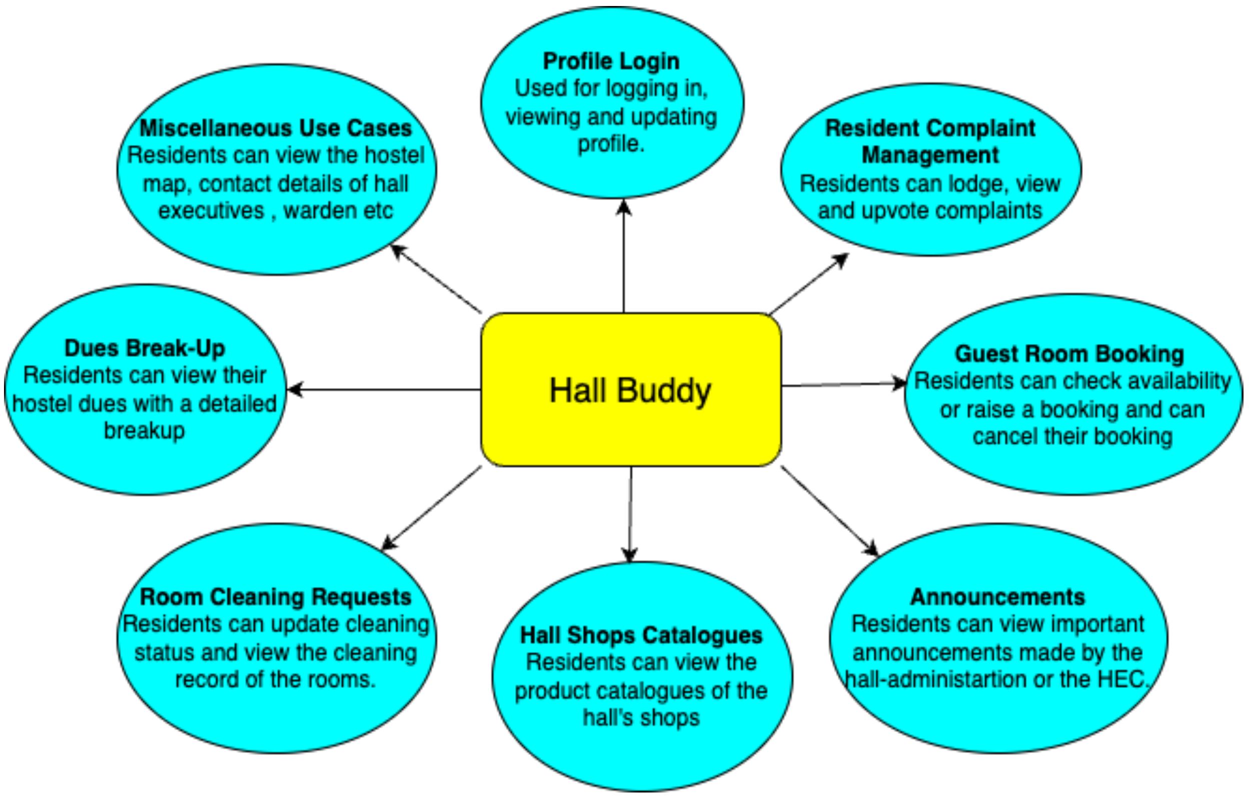
We also would like to thank Prof. Indranil Saha for providing the SRS template and teaching the concepts.

# 2 Overall Description

## 2.1 Product Overview

HallBuddy is a Hall Management System which would be useful for all hostel residents and the hostel administration. The IITK campus does have Pingala for general complaints but other services like guest room booking, hostel-complaint management, room cleaning requests are not incorporated in the same. Also Pingala is a campus-generic software but not built to cater to Hostel Residents needs. Students need to do guest room booking manually by visiting the hall office. The process can be made hassle free if switched online. The cleaning facility in each hostel is inconsistent. An online portal wherein the student residing in the hostel marks the date on which his/her room was cleaned would keep it in check. There would also be a forum wherein students could discuss common hall queries and post mess food reviews. When posting a complaint, they can check if similar complaints have been registered recently as to avoid the load on the admin server. There would be a separate section for the hostel shops wherein the catalogues of various hostel shops would be displayed so that students can see if the product they need is listed on the catalogue and can convey their demands to the hall shop owners.

The admin login can be used to check whether the complaints have been resolved, see mess reviews and the frequency of cleaning done by the staff.


Profile Login
Used for logging in, viewing and updating profile.
Miscellaneous Use Cases
Residents can view the hostel map, contact details of hall executives , warden etc
Resident Complaint Management
Residents can lodge, view and upvote complaints
Dues Break-Up
Residents can view their hostel dues with a detailed breakup
Hall Buddy
Guest Room Booking
Residents can check availability or raise a booking and can cancel their booking
Room Cleaning Requests
Residents can update cleaning status and view the cleaning record of the rooms.
Hall Shops Catalogues
Residents can view the product catalogues of the hall's shops
Announcements
Residents can view important announcements made by the hall-administartion or the HEC.

## 2.2 Product Functionality

This product is intended to come with functionalities that emphasize student service and vibrant interactions. It includes the following features:

- Profile (Login)
- Resident Complaint Management
- Guest room Booking
- Hall shop catalogue
- Room cleaning requests
- Mess food review

## 2.3 Design and Implementation Constraints

The design & implementation constraints for this software product may include the following -

- Users must have their correct usernames and passwords to log in to their accounts and do necessary operations.
- Response time for the website should be low ensuring efficient use
- Users may access from any computer that has Internet browsing capabilities and an Internet connection. General knowledge of basic computer skills is required to use the product.
- If the software is accessed through various web browsers, ensuring compatibility with various browser and versions is important.
- Establishing reliable data backup and recovery mechanisms is crucial to prevent data loss in the event of system failures or other unforeseen circumstances.
- If the hostel has a diverse population, then language preferences may be needed to be considered in order to ensure inclusivity and ease of use for residents.

## 2.4 Assumptions and Dependencies

The following assumptions and dependencies are considered for the software:

- User Co-operation: We assume that the hostel administration and students would co-operate in using and adapting to the new system
- We assume that there would be adequate resources for support, maintenance and updates after the software is deployed
- The success of the application depends on support from hostel administration and students

No other specific dependencies or assumptions are considered at this point.

# 3 Specific Requirements

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

The UI for the Hall Management System web app is designed for seamless user interaction. The login page ensures secure access, requiring user authentication. The dashboard provides an intuitive overview of hall activities, featuring real-time updates on hall announcements and complaints. The interface is clean and organized, allowing users to efficiently navigate through features like room cleaning status, guest room booking, complaint registration etc. Emphasizing user-friendly design, it prioritizes easy access to key functionalities, enhancing the overall user experience for efficient hall management.

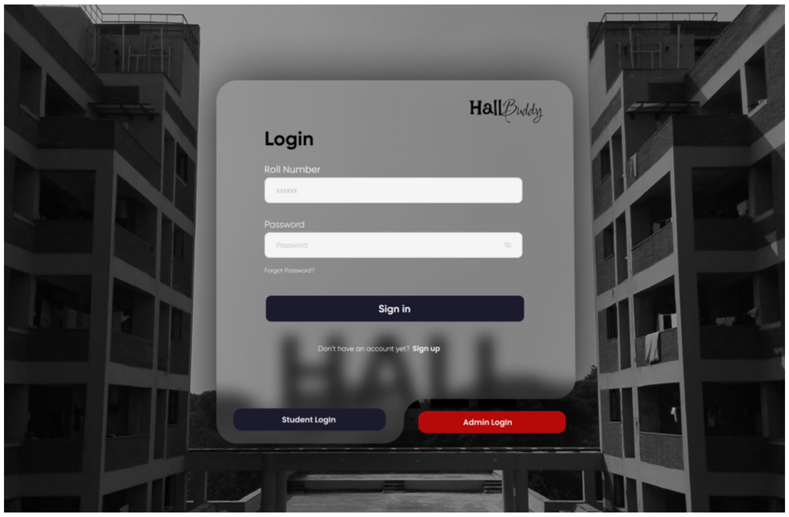
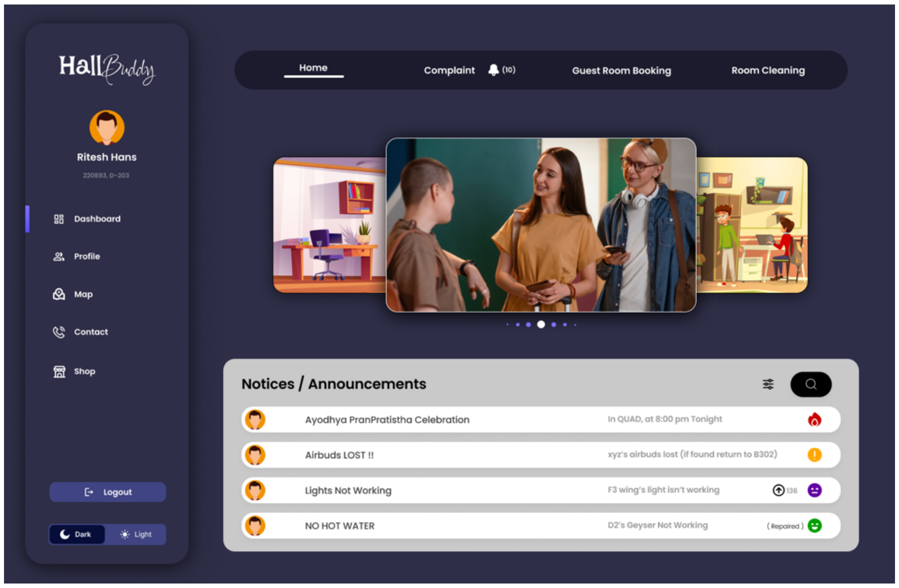
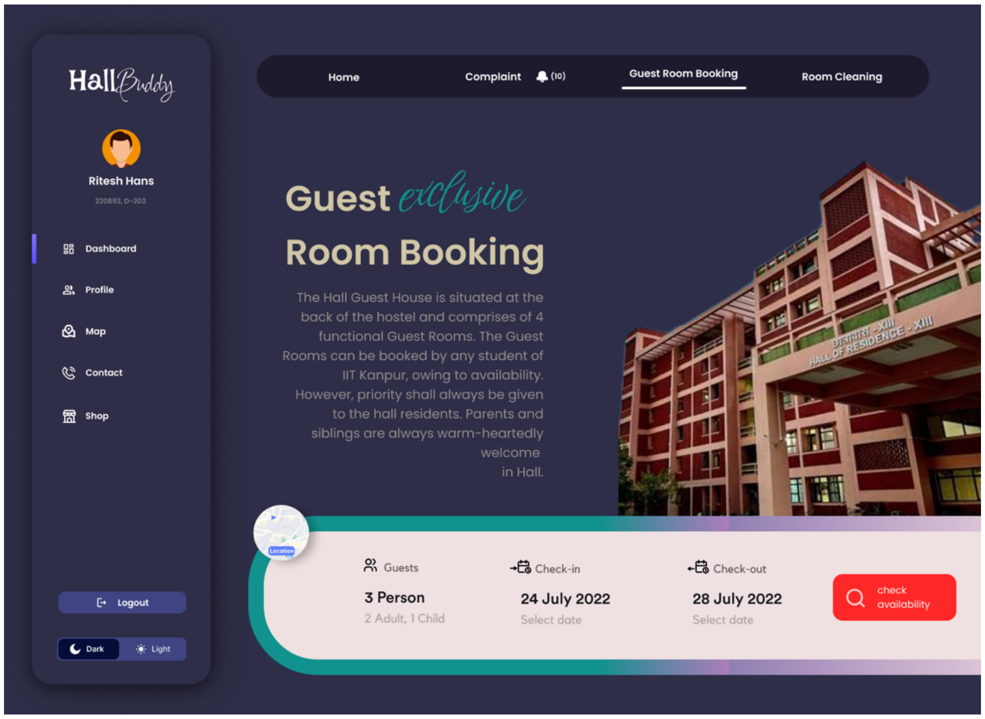


Fig 1 – Login Page



Ayodhya PranPratistha Celebration — In QUAD, at 8:00 pm Tonight
Airbuds LOST !! — xyz's airbuds lost (if found return to B302)
Lights Not Working — F3 wing's light isn't working — 136
NO HOT WATER — D2's Geyser Not Working — ( Repaired )

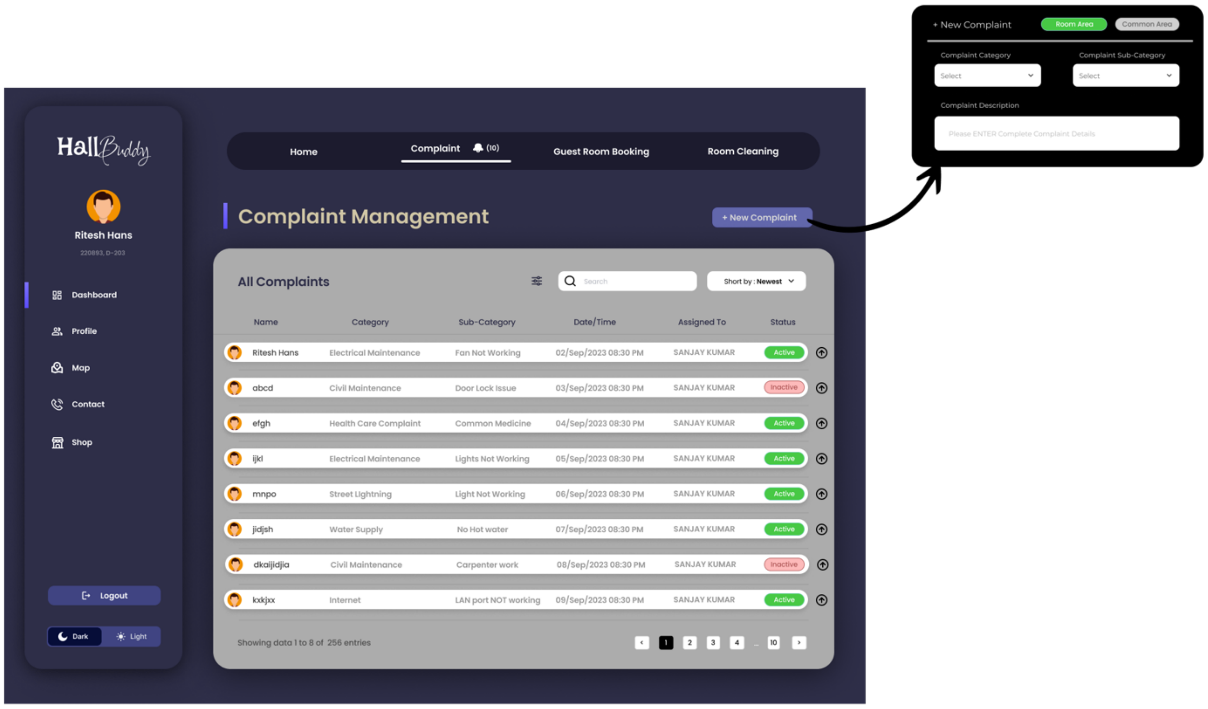
Fig 2: Home Page



The Hall Guest House is situated at the back of the hostel and comprises of 4 functional Guest Rooms. The Guest Rooms can be booked by any student of IIT Kanpur, owing to availability. However, priority shall always be given to the hall residents. Parents and siblings are always warm-heartedly welcome in Hall.

\

Fig 3: Guest Room Booking

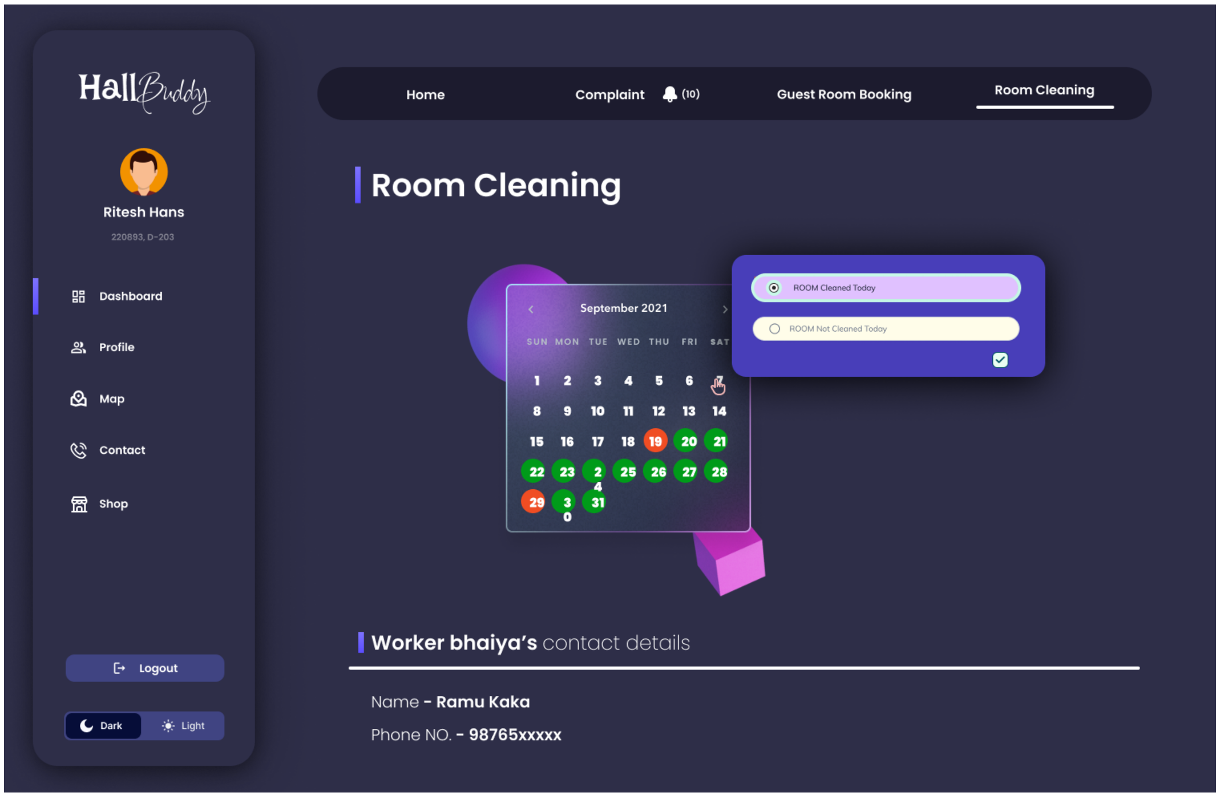


| Name | Category | Sub-Category | Date/Time | Assigned To | Status |
|---|---|---|---|---|---|
| Ritesh Hans | Electrical Maintenance | Fan Not Working | 02/Sep/2023 08:30 PM | SANJAY KUMAR | Active |
| abcd | Civil Maintenance | Door Lock Issue | 03/Sep/2023 08:30 PM | SANJAY KUMAR | Inactive |
| efgh | Health Care Complaint | Common Medicine | 04/Sep/2023 08:30 PM | SANJAY KUMAR | Active |
| ijkl | Electrical Maintenance | Lights Not Working | 05/Sep/2023 08:30 PM | SANJAY KUMAR | Active |
| mnpo | Street Lightning | Light Not Working | 06/Sep/2023 08:30 PM | SANJAY KUMAR | Active |
| jidjsh | Water Supply | No Hot water | 07/Sep/2023 08:30 PM | SANJAY KUMAR | Active |
| dkaijidjia | Civil Maintenance | Carpenter work | 08/Sep/2023 08:30 PM | SANJAY KUMAR | Inactive |
| kxkjxx | Internet | LAN port NOT working | 09/Sep/2023 08:30 PM | SANJAY KUMAR | Active |

Showing data 1 to 8 of 256 entries

‹ 1 2 3 4 ... 10 ›

Fig 4 : Complaint Management



Name - Ramu Kaka
Phone NO. - 98765xxxxx

Fig 5: Room Cleaning

### 3.1.2 Hardware Interfaces

Monitor Screen - The system shall display all the information associated with the website via the monitor screen.
Mouse - The mouse shall execute all command buttons, activate areas for data input and select options from the menus.
Keyboard - The system shall allow the user to enter text using the keyboard into the active areas of the database.
Database Server-

### 3.1.3 Software Interfaces

From the client's side, the website will be accessible from a modern web browser such as Google Chrome, Mozilla Firefox, etc.

## 3.2 Functional Requirements

(Note: The term '**user**' in a requirement refers to hostel residents The term '**admin**' here refers to the hostel administration).

### 3.2.1 Users can Sign-Up and Login:

a) New users can register by entering Phone-Number which will be used for OTP authentication.

b) Phone-Number will be cross-checked with hostel database during authentication.

c) After, authentication users can setup their profiles by entering Name, Roll Number, Room Number, Phone-No. and IITK email address.

d) Users can set-up a password for their profile.

e) System will allow users to login by using their Roll-No. and Password.

f) Users can change their password via OTP verification.

g) Users can use the 'Forgot Password' feature to set-up a new password via OTP authentication.

### 3.2.2 Maintaining User profiles:

a) System ensured that user-credentials are unique for each user.

b) Users can view and update their profile.

### 3.2.3 System will allow Community Communications

#### 3.2.3.1 **Announcements**: -

**a)** Users can view important announcements made by the hall-administration or the HEC.

b) Admin can post announcements/broadcast messages to the entire Hall Community.

#### 3.2.3.2 **Lost/Found:**

a) Users can list their lost items by providing item details such as Name of the item, Brand, Color, and a Brief Description.

b) If users report an unclaimed item under 'Found' category along with the same item details: Name of the item, Brand, Color, and a Brief Description.

### 3.2.4 Complaint Management System

#### 3.2.4.1 User Functionalities:

a) System shall show the current list of pending complaints.

b) Complaints will be tagged with attributes such as location and category (e.g. electricals, plumbing, furniture etc.)

c) Users can upvote an existing complaint (to avoid multiple complaints of the same kind)

d) Complaints will be sorted according to number of votes while viewing pending complaints.

e) Users can lodge a new complaint: lodge a room-specific complaint or a common-area complaint.

#### 3.2.4.2 Admin Functionalities:

a) Admin can view pending complaints. (Filter by tags)

b) Admin can update status of a complaint from Pending à In Progress, merge complaints. This would send a notification to all users who upvoted that complaint.

### 3.2.5 Guest Room Booking System

a) Users can check availability and tariffs of guest rooms on their selected date and raise a booking request.

b) Users can cancel their existing booking.

c) Admin can approve(disapprove) booking requests and send notification to the user.

### 3.2.6 Room Cleaning Services:

a) Users can update cleaning status and view the cleaning record of their rooms.

b) Users can contact the cleaning staff to schedule room cleaning.

c) Admin can access room cleaning status of all rooms and supervise the cleaners.

### 3.2.7 Hostel Map and Fire-Exit Pathways:

a) Users can view a detailed hostel map which will help them navigate to different parts of the hostel.

b) Users can view the Fire-Exit Path in the Hostel.

### 3.2.8 Contact details of Hall Administration:

a) Users can view the contact details of the Warden, Hall Administration, and the HEC.

b) Contact details of the Hall SIS Guard will also be available to the users in the case of emergency.

### 3.2.9 Hall Shop Catalog

a) All the utility stores of the hostel such as stationery shop, barber shop, canteen, departmental shop etc. will be listed on the system along with their timings, contact details of shop owners, menus/catalog of items sold.

### 3.2.10 Dues Break-Up

a) Users can view their hostel dues for a given semester.

b) Dues will include of break-up of mess charges, electricity bill, and extras.

c) Users can view the last date to pay hostel charges and fine charged for paying fees late.

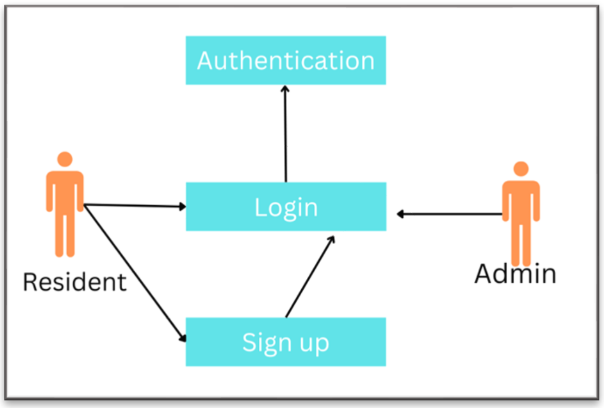
d) Admin will have access to update dues for any resident.

## 3.3 Use Case Model

### 3.2.1 Use Case #1: Sign Up and Login

| | |
|---|---|
| **Author** | Rohan Batra |
| **Purpose** | This use case is used for the Login/ Signup of the Resident and the Admin. Login is used for the registered users and the Signup is used for the first-time users. |
| **Requirements Traceability** | The Requirements will be asked at the Login Page and will provide with Username, Roll Number, Room No. for the session. |
| **Priority** | High |
| **Preconditions** | Possession of login credentials for the Resident – (Roll Number and Password) and Admin (Password) is assumed. |
| **Post conditions** | The user gets authenticated and is logged in the software with the requested access on successful authentication, failing to which an error is displayed. |
| **Actors** | Resident / Admin |
| **Exceptions** | The user forgets the password and thus will select the “Forgot Password” Option. A verification code will be sent to the registered email-id and the user will be able to create a new password after verifying the code. |
| **Includes** | None |
| **Notes/Issues** | Residents will login through using their Roll Number and Password. For the Sign-up Cases, the first login will be through an Admin Sent password after which the users will have to create their own password. Admin Login will be through Password. |

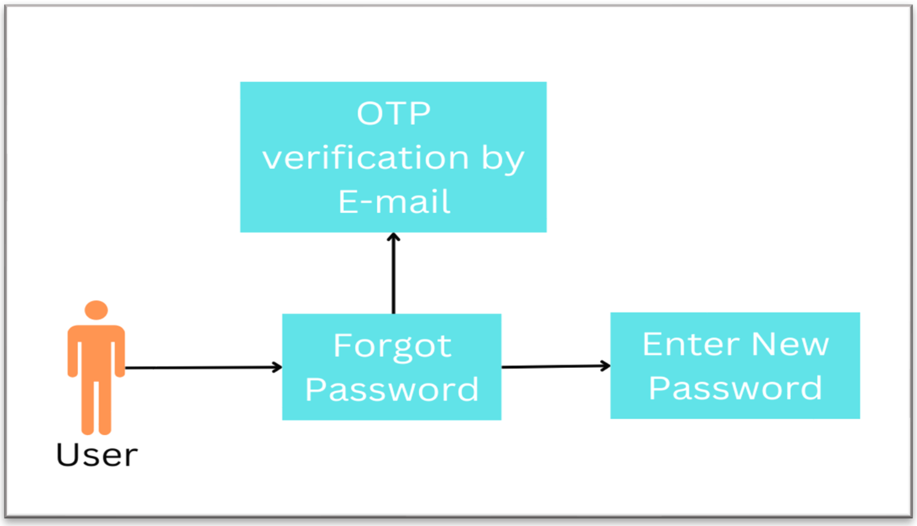
**Diagram**



### 3.2.2 Use Case #2: Forgot Password

| | |
|---|---|
| **Author** | Rohan Batra |
| **Purpose** | This use case is used for new password creation when the user forgets their password. |
| **Requirements Traceability** | User will be asked to verify the code sent on the Registered Email Id after which they will be able to create a new password. |
| **Priority** | High |
| **Preconditions** | User doesn’t know their password and selects the “Forgot Password” Option |
| **Post conditions** | The newly created password will be thence used for that user’s login and authentication |
| **Actors** | Resident / Admin |
| **Exceptions** | An error message is displayed to the user if the user validation of the Verification Code fails. |
| **Includes** | None |
| **Notes/Issues** | The Registered Email id for the user will be taken up from the Data Base. |

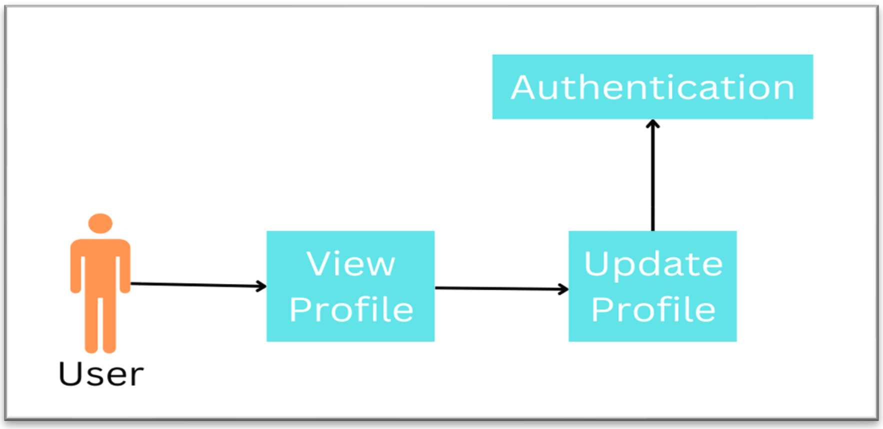
**Diagram:**



### 3.2.3 Use Case #3: View and Update Profile

| | |
|---|---|
| **Author** | Krutuparna Paranjape, Rohan Batra |
| **Purpose** | This use case allows the Resident to view and update their Profile Information including Room Number and Contact Details. |
| **Requirements Traceability** | In the Profile Section the Room Number and the Contact Details of the resident will be displayed, the resident will be able to update them. |
| **Priority** | High |
| **Preconditions** | Resident is logged in to the software and chooses the Profile Section to view / update their profile information. |
| **Post conditions** | The new details entered by the Resident are updated in the database. |
| **Actors** | Resident |
| **Exceptions** | None |
| **Includes** | Use Case #1 |
| **Notes/Issues** | The Profile Details will be read from and written to the Database. |

**Diagram:**



### 3.2.4 Use Case #4: Complaint Management

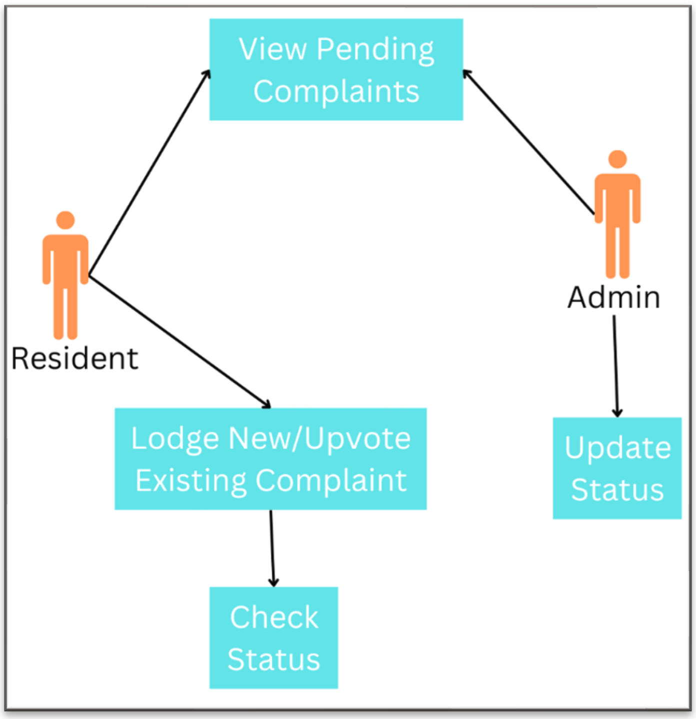
#### 3.2.4.1 Viewing and Upvoting Complaints

| | |
|---|---|
| **Author** | Rohan Batra |
| **Purpose** | This use case allows the users to view the existing complaints and check their response status. The Users will be able to filter down complaints based on the Category Tags and Location and sort them as per the number of upvotes or time of creation etc. The Resident can upvote existing complaints if they concern them. The admin can manage the response status of the complaints as – "Resolved", "In Progress". |
| **Requirements Traceability** | The complaints displayed are maintained in the database with their tags, upvotes and status. The attributes are updated as per the user interaction described. |
| **Priority** | High |
| **Preconditions** | Resident is logged in to the software and selects the "Complaint" Tab from the Dashboard. |
| **Post conditions** | Any interactions of the user- change in status by Admin, Upvoting etc. are recorded in the Database. |
| **Actors** | Resident / Admin |
| **Exceptions** | None |
| **Includes** | Use Case #1 |
| **Notes/Issues** | The Complaints lodged by the Resident will be displayed at the top to them followed by other complaints sorted by number of upvotes by default. |

#### 3.3.4.2 Creating New Complaints

| | |
|---|---|
| **Author** | Krutuparna Paranjape, Rohan Batra |
| **Purpose** | This use case allows the Resident to create new complaints. The complaints can be created in the "Room" or the "Common Area" categories. The resident will be able to enter text description, attach images and select suitable filters based on the category of complaint (e.g. Electrical, Civil maintenance etc.). |
| **Requirements Traceability** | In the "Create New Complaint" Dialog Box in the "Complaint" Tab, the complaint details entered by the Resident are captured. |
| **Priority** | High |
| **Preconditions** | Resident is logged in to the software and chooses the "Create New Complaint" option in the "Complaints" Tab. |
| **Post conditions** | Complaint Details entered by the resident are captured and written in the Database. |
| **Actors** | Resident |
| **Exceptions** | It will be essential for the user to provide nonempty title and description of the complaint failing which an error message will be displayed. |
| **Includes** | Use Case #1 |
| **Notes/Issues** | The creation of a new complaint sends a notification to the Admin. A "Common Area" complaint gets added to the existing complaints where other residents can view and upvote it. |

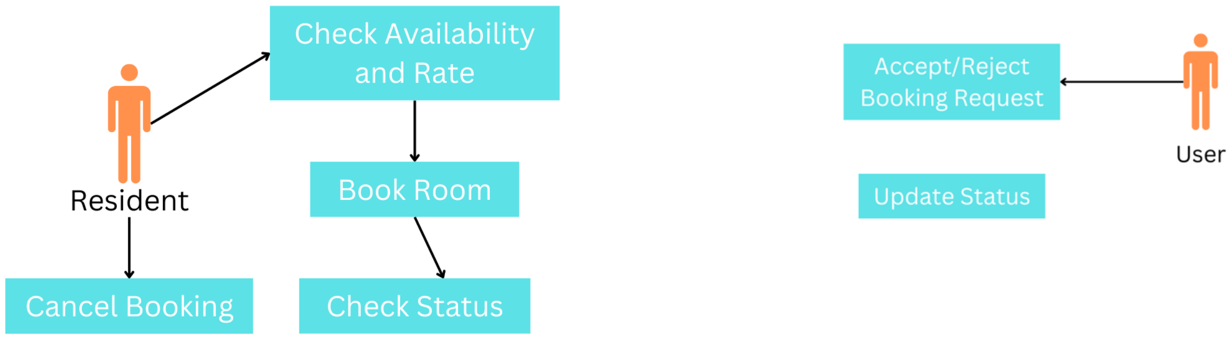
**Diagram:**


View Pending Complaints
Resident
Admin
Lodge New/Upvote Existing Complaint
Update Status
Check Status

### 3.2.5 Use Case #5: Guest Room Booking

| | |
|---|---|
| **Author** | Mridul Gupta |
| **Purpose** | This use case is for handling guest room bookings. The user can view his current bookings and can also request to book a guest room based on availability. The admin must approve the request before the booking is confirmed. This also allows the user to cancel his/her booking |
| **Requirements Traceability** | Guest Room Booking Page, Acceptance or Rejection of request by admin |
| **Priority** | High |
| **Preconditions** | The registered user must be logged in before making a request and there should be a vacancy for the dates specified, for a request to be applied. |
| **Post conditions** | Booking request is registered/Request status is changed. |
| **Actors** | Resident / Admin |
| **Exceptions** | None |
| **Includes** | Use Case #1 |
| **Notes/Issues** | None |

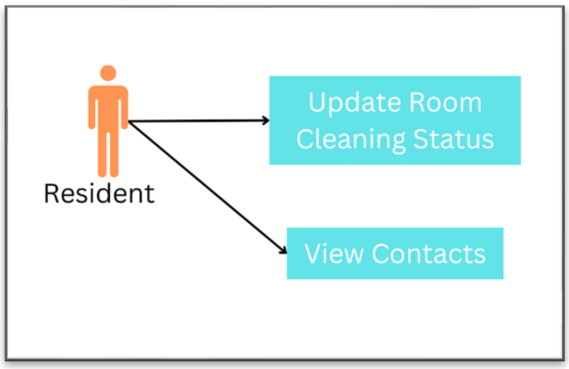
**Diagram:**

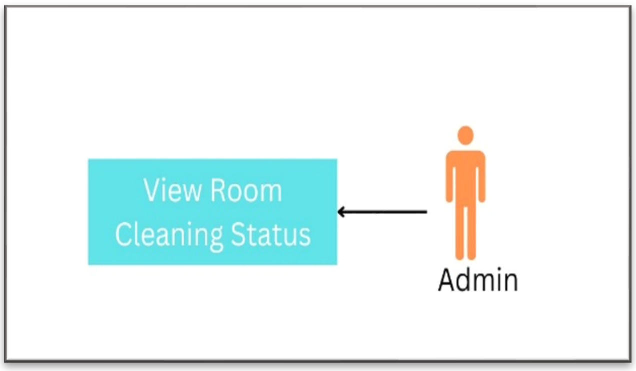


### 3.3.6 Use Case #6: Room Cleaning Status

| | |
|---|---|
| **Author** | Mridul Gupta |
| **Purpose** | The purpose of this user case is to provide users with the ability to update the cleaning status of rooms within the hall management system. This functionality ensures real-time tracking and communication of the cleanliness status of individual rooms, along with providing contact information for the responsible cleaning personnel. This enables efficient management of housekeeping activities. |
| **Requirements Traceability** | Update the cleaning status by user. |
| **Priority** | High |
| **Preconditions** | A registered user must be logged in before updating the cleaning status of his room. |
| **Post conditions** | Room Cleaning Status is viewed/updated. |
| **Actors** | Resident / Admin |
| **Exceptions** | None |
| **Includes** | Use Case #1 |
| **Notes/Issues** | None |

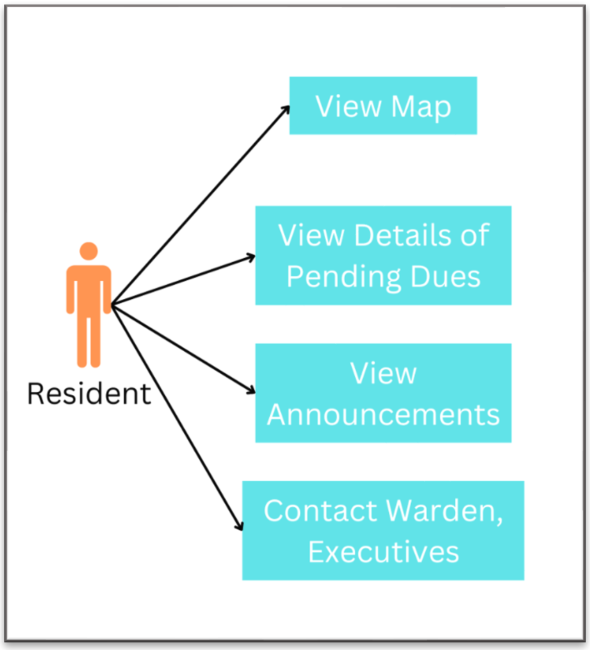
#### Diagram

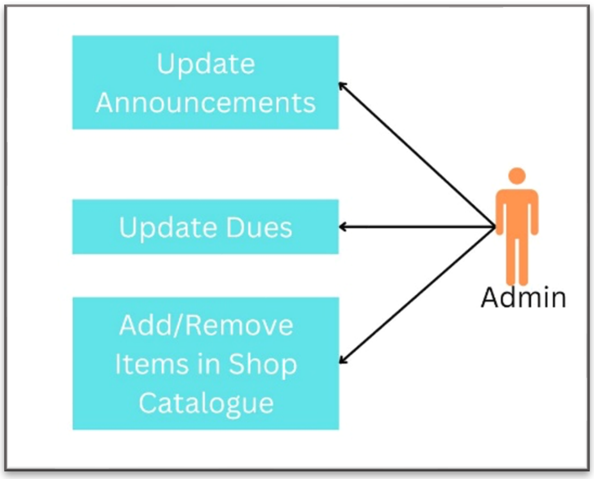




### 3.3.7 Use Case #7: Miscellaneous Use Cases

| | |
|---|---|
| **Author** | Krutuparna Paranjape, Mridul Gupta |
| **Purpose** | This use case covers all miscellaneous actions of the user/admin such as viewing map of the hostel, contacts of the hall executives, view pending dues. |
| **Requirements Traceability** | View different information pages |
| **Priority** | - Medium |
| **Preconditions** | A registered user must be logged in. |
| **Post conditions** | User is able to view the desired information by going to the specific tab. |
| **Actors** | Resident / Admin |
| **Exceptions** | None |
| **Includes** | Use Case #1 |
| **Notes/Issues** | None |





# 4 Other Non-functional Requirements

## 4.1 Performance Requirements

- Latency: API request from DBMS should not have delay of more than 200ms
- Scalability: For implementation in a general hostel set up we assume a traffic of up to 5000 users at a time. The API should be able to handle this traffic simultaneously to be implemented in the general case.
- Reliability: No scope for errors in the mess dues section since monetary transactions will be based on the data collected here.
- Mobile Responsiveness: Webpage should be optimised and made responsive to function on mobile devices as well.
- Ease of Use: Software have an intuitive UI and should not require any training to learn to use.
- Security: The database should not be vulnerable to access by user without admin credentials.
- Integrity: Guest Room booking should be bug free and avoid errors such as duplicate booking and false reservations.

## 4.2 Safety and Security Requirements

- All operations will require login from the user, and passwords will be encrypted using 32-bit hash encoding
- Write operations to the Mess Dues and Shop Catalogue database should be open to access only by Admin-approved users.
- Software should be immune to vulnerabilities like SQL injection attacks.
- Software will allow password recovery via mail verification.
- IP address of any user will be logged by the system.

## 4.3 Software Quality Attributes

### 4.3.1 Availability:

- Features such as guest room booking and mess dues viewing should be available to users 24/7.
- To this end, database will have rollback segments and centralised backup can be performed to recover data in the event of database failure.

### 4.3.2 Flexibility, Scalability:

- The software should adapt to changing hostel requirements such as the introduction or removal of a vendor, multiple messes, different number of guest rooms etc.
- The software will be made modular so that such changes can be incorporated easily.

### 4.3.3 Usability:

- Frontend of the software will be designed in a user- friendly and intuitive manner so that it is simple and easy to use.

### 4.3.4 Maintainability:

- System should have the ability to reset to default (for example mess dues after every semester)
- Programming code will be modular, functional and will be commented thoroughly for ease of maintenance and editing.

### 4.3.5 Portability:

- System should be responsive to mobile/local infrastructure so that users can access essential features like announcements and mess dues instantly.

### 4.3.6 Adaptability:

- Website should be compatible with Windows, Linux, and MacOS.

### 4.3.7 Reliability:

- There will be two databases so that any failure can be recovered easily.
- The software should be reliable for both students (hostel residents) and hotel-administration.

### 4.3.8 Reusability:

- Relevant classes or functions that are already available on the internet should be used in the implementation of the software so that more time can be devoted to other novel features/functions.
- The codes should be reusable for other relevant software.

### 4.3.9 Interoperability:

- The software can run on windows, Linux, iPad etc.

### 4.3.10 Maintainability:

- Adding relevant features according to the feedback of the clients of the software should be easy.

# 5 Other Requirements

**Database Requirements:**
This application requires the student-database maintained by IIT-Kanpur administration.

**Legal Requirements:**
User data must be secure and data-protection policies must be complied with.

# Appendix A – Data Dictionary

**User Class (Hostel Student):**

| **Element Name** | **Description** | **Attributes** | **Operations** |
|---|---|---|---|
| User | Each user registers using his/her Roll Number, Email ID and Password which will be later used to login into their account. Also, the user must add details like Room Number, Contact Number and Name. | 1. Roll-No.: integer<br>2. emailID : string<br>3. password: string<br>4. Room Number: String<br>5. Contact Number: integer<br>6. Name: String | 1) **register ():** registration of a user using the application for the first time.<br>2) **login ()**: called while the user wants to login.<br>3) **edit_profile():** to edit the details of the user.<br>4) **logout ():** used to logout from application |

**Admin Class (Hostel Administration)**

| **Element Name** | **Description** | **Attributes** | **Operations** |
|---|---|---|---|
| Admin | Each member of the hostel administration registers with his unique ID and Password, which will be used for future logins. | 1. name: string<br>2. user_ID: image<br>3. password: image<br>4. contact no: integer | 1. **register ():** registration of admin for the first time<br>2. **login ():** for admin-login<br>3. **edit_profile():** update profile<br>4. **announce ():** used to post notices and announcements.<br>5. **updateComplaint():** used to update the status of a complaint.<br>6. **bookRoom()**: used to accept/reject guest room booking requests.<br>7. **updateDues():** to update fee-dues of a student<br>8. **updateCatalog():** update catalog of the hall's general store<br>9. **viewRoom_Cleaning_Status():** access room cleaning status of any room |

**Complaint Class:**

| **Element Name** | **Description** | **Attributes** | **Operations** |
|---|---|---|---|
| Complaint | The complaints that have been lodged by the users. Each complaint has a Title, Description, no. of upvotes and Status. | 1. name: string<br>2. img1: image<br>3. img2: image<br>4. description string<br>5. location: integer<br>6. lodge_date: datetime<br>7. upvotes: integer<br>8. status: string | 1. **lodgeComplaint():** used to lodge a new complaint<br>2**.upvote ():** used to upvote an existing complaint<br>3.**checkStatus ():** used to check status of an existing complaint |

**Guest_Room_Class**

| **Element Name** | **Description** | **Attributes** | **Operations** |
|---|---|---|---|
| room | User can check availability of guest rooms and their tariffs and can raise a booking request | 1. name: string<br>2. img1: image<br>3. tariff: integer | 1. **update_availability(): used** to update availability of the given room<br>2. **update_price()**: used to update tariffs of the guest room |

**Notice_Class**

| **Element Name** | **Description** | **Attributes** | **Operations** |
|---|---|---|---|

| | | | |
|---|---|---|---|
| Notice | Important notices posted by the administration to address all hall residents | 1. name: string<br>2. description string<br>3. date: datetime | 1. **update_notice():** used to post new/update existing notices |

**Lost/Found_Class**

| **Element Name** | **Description** | **Attributes** | **Operations** |
|---|---|---|---|
| Lost/Found | Various lost/found objects posted by residents | 1. name: string<br>2. img1: image<br>3. description: string<br>4. date: datetime | 1. **update_lost/found ():** used to post a new lost/found object |

**Room_Cleaning_Class**

| **Element Name** | **Description** | **Attributes** | **Operations** |
|---|---|---|---|

| | | | |
|---|---|---|---|
| Cleaning_Log | Record of room-cleaning status of each room of the hostel | 1. room_name: string<br>2. date: datetime<br>3. status | 1. **update_cleaning_status():** used to update room-cleaning status of a room<br>2. **history ():** fetch room cleaning history |

# Appendix B - Group Log

Our entire team has been extremely enthusiastic about the project since its inception.

We have formed a WA grp and a Discord server to support a communication and collaborative development.

We divided tasks amongst ourselves and through constant discussion we ensured that everyone was in sync with each other to maintain consistency of the document.

| Meeting Minutes | Agenda |
|---|---|
| 16/01/24 | Meeting between all 10 members to SRS document, team was divided into two sub-groups of five each and work was divided within each sub-group. |
| 19/01/24 | Short meet to check-up on each other's contribution and make suggestions |
| 22/01/24 | First Meeting with TA: Discussed the SRS document with TA and incorporated his suggestions. |
| 24/01/24 | Team meeting of all 10 members for proof-reading and final modifications. |
| 25/01/24 | Online Meet with TA and to review the doc |