



**Department of Computer Science**

---

University of Gujrat

## HOSTEL ADMINISTRATION SYSTEM



Hostel Administration system

**Session: BSCS 2019-2023**

**Project Advisor: Dr Nauman Riaz Chaudhry**

**Submitted By**

*Shahzaib Saeed*                      *19011519-071*

*Mohammad Ryan*                      *19011519-133*

*Maaz Hassan*                      *19011519-138*

---

**Department of Computer Science**

## STATEMENT OF SUBMISSION

This is certify that **Shahzaib Saeed** Roll No. **19011519-071** and **Mohammad Ryan** Roll No. **19011519-133** and **Maaz Hassan** Roll No. **19011519-138** has successfully completed the final year project named as “**Hostel Administration System**” at the Department of Computer Science, University of Gujrat, to fulfill the requirement of the degree of **BS in Computer Science**.

---

Project Supervisor

---

Project Coordination Office  
Faculty of C&IT -UOG

---

Chairperson  
Department of Computer Science

## **Acknowledgement**

We truly acknowledge the cooperation and help make by Dr Nauman Riaz Chaudhry, Chairman, Department of Computer Science, University of Gujrat. He has been a constant source of guidance throughout the course of this project. We would also like to thank our project supervisor Dr Nauman Riaz Chaudhry for his help and guidance throughout this project. We are also thankful to our friends and families whose silent support led us to complete our project.

Date:

## **Abstract**

The Hostel Administration System is a comprehensive and user-friendly software solution designed to simplify the process of finding and managing hostel accommodations. In today's dynamic world, students and working professionals often find themselves in need of temporary housing solutions in unfamiliar cities. This system aims to bridge the gap between hostel seekers and hostel administrators, providing a centralized platform for effortless accommodation management. The system offers a range of features to enhance the user experience. Hostel administrators can efficiently register their hostels, inputting essential details such as location, room configurations, and amenities. Prospective boarders can explore these hostels, view detailed information, and make informed decisions based on their preferences. The system also facilitates hassle-free bookings, allowing users to reserve rooms seamlessly and even interact with hostel administrators. Through an intuitive user interface, the Hostel Administration System enables quick searches, filtered results, and access to comprehensive hostel information. Hostel administrators benefit from an admin panel that streamlines management tasks, such as adding or updating hostel details, rooms, and boarder records. In essence, the Hostel Administration System aims to provide a unified solution for hostel seekers and administrators, easing the process of finding suitable accommodations and enhancing the management of hostel facilities. With its interactive features and user-centered design, the system transforms the experience of seeking temporary housing, contributing to a more streamlined and efficient accommodation landscape.

## TABLE OF CONTENTS

<b>CHAPTER 1: PROJECT FEASIBILITY REPORT.....</b>	<b>7</b>
1.1. INTRODUCTION .....	1
1.2. PROJECT/PRODUCT FEASIBILITY REPORT.....	1
1.2.1. Technical Feasibility.....	2
1.2.2. Operational Feasibility.....	2
1.2.3. Economic Feasibility .....	2
1.2.4. Schedule Feasibility.....	2
1.2.5. Specification Feasibility.....	3
1.2.6. Information Feasibility .....	3
1.2.7. Motivational Feasibility.....	3
1.2.8. Legal & Ethical Feasibility.....	3
1.3. PROJECT/PRODUCT SCOPE .....	3
1.4. PROJECT/PRODUCT COSTING .....	3
1.5. TASK DEPENDENCY TABLE .....	4
1.6. CPM - CRITICAL PATH METHOD.....	4
1.7. GANTT CHART .....	5
1.8. ALLOCATION OF MEMBERS TO ACTIVITIES .....	5
1.9. TOOLS AND TECHNOLOGY WITH REASONING .....	6
1.10. VISION DOCUMENT .....	6
1.11. PRODUCT FEATURES/ PRODUCT DECOMPOSITION .....	7
<b>CHAPTER 2: SOFTWARE REQUIREMENT SPECIFICATION (FOR OBJECT ORIENTED APPROACH) .....</b>	<b>9</b>
2.1 INTRODUCTION: .....	10
2.2 SYSTEMS SPECIFICATIONS.....	11
2.2.1. Identifying External Entities .....	11
2.2.2. Context Level Data Flow Diagram: .....	11
2.2.3. Capture "shall" Statements:.....	12
2.2.4 Allocate Requirements:.....	13
2.2.5. Prioritize Requirements (If Any):.....	14
2.3. EXISTING SYSTEMS / LITERATURE REVIEW:.....	16
2.3.1. Existing System .....	16
2.4. USECASE DIAGRAM OF YOUR PROJECT:.....	17
2.4.1. Usecase Description .....	18
<b>CHAPTER 3: DESIGN DOCUMENT (FOR OBJECT ORIENTED APPROACH) .....</b>	<b>20</b>
3.1. INTRODUCTION: .....	21
3.2. DOMAIN MODEL .....	21
3.3. DESIGN CLASS DIAGRAM.....	22
3.4. SEQUENCE DIAGRAM .....	23
3.5. STATE CHART DIAGRAM.....	25
3.6. COLLABORATION DIAGRAM.....	26
a. Contents of Collaboration Diagrams.....	26

<b>CHAPTER 4: USER INTERFACE DESIGN .....</b>	<b>27</b>
4.1. INTRODUCTION.....	28
4.2. SITE MAPS .....	28
4.3. STORY BOARDS .....	29
4.4. NAVIGATIONAL MAPS: .....	30
<b>CHAPTER 5: SOFTWARE TESTING .....</b>	<b>31</b>
5.1 INTRODUCTION: .....	32
5.2. TEST PLAN.....	32
5.2.1. <i>Purpose</i> .....	32
5.2.2. <i>Outline</i> .....	33
5.3. TEST DESIGN SPECIFICATION .....	40
5.3.1. <i>Purpose</i> .....	40
5.3.2. <i>Outline</i> .....	40
5.4. TEST CASE SPECIFICATION.....	43
5.4.1. <i>Purpose</i> .....	43
5.4.2. <i>Outline</i> .....	43
5.5. TEST PROCEDURE SPECIFICATION .....	44
5.5.1. <i>Purpose</i> .....	44
5.5.2 <i>Outline</i> .....	45
5.6. TEST ITEM TRANSMITTAL REPORT .....	46
5.6.1. <i>Purpose</i> .....	46
5.6.2. <i>Outline</i> .....	46
5.7. TEST LOG .....	47
5.7.1. <i>Purpose</i> .....	47
5.7.2. <i>Outline</i> .....	47
5.8. TEST INCIDENT REPORT .....	48
5.8.1. <i>Purpose</i> .....	49
5.8.2. <i>Outline</i> .....	49
5.9. TEST SUMMARY REPORT.....	50
5.9.1. <i>Purpose</i> .....	50
5.9.2. <i>Outline</i> .....	50
<b>CHAPTER 6: USER MANUAL .....</b>	<b>52</b>

## **Chapter 1: Project Feasibility Report**

---

## **1.1. Introduction**

This is a web-based software application (web application) designed to manage and streamline the administrative tasks of a hostel or dormitory. The system allows hostel administrators to efficiently manage various hostel operations such as room allocation, meal management, visitor management, inventory management, and billing.

The Hostel Administration System is an innovative solution for students and working professionals searching for reliable and feasible hostel accommodations in a distant town. With this system, users can easily search for verified hostels in their desired area, featuring all facilities, including price range and services, at their fingertips. Users can sign up and gain access to the system, where they can view and book available hostel accommodations. Hostel owners can also register and post their hostel properties with necessary descriptions, pictures, and services, and keep records of their boarders. Overall, the Hostel Administration System provides an easy and relaxing alternative to the traditional way of finding hostel accommodations, making the process efficient and stress-free.

The admin panel of the Hostel Administration System will serve as the central control hub for the system. It will provide the system administrator with the ability to manage users, hostels, rooms, and bookings. The admin panel will have a clean and user-friendly interface that allows the admin to easily access and manage different modules of the system. It will also feature role-based access control, which means that only authorized users will be able to access certain features of the system. The admin panel will streamline the management process of the system, allowing the admin to efficiently manage the hostels, rooms, and users, and providing a seamless user experience to the end-users.

The system typically includes features such as room allocation and real-time monitoring of hostel tenancy. It also allows hostel administrators to track the daily attendance of hostel native, manage their personal details, and generate reports on various hostel-related activities.

This project helps to streamline the entire hostel management process, eliminates manual data entry and paperwork, and provides an efficient and organized approach to hostel administration. It helps to reduce the workload of the hostel staff, enabling them to focus on other important tasks while ensuring a smooth and hassle-free experience for hostel native.

## **1.2. Project/Product Feasibility Report**

When a project is started the first matter to establish is to assess the feasibility of a project or product. Feasibility means the extent to which appropriate data and information are readily available or can be obtained with available resources such as staff, expertise, time, and equipment. It is basically used as a measure of how practical or beneficial the development of a software system will be to you (or organization). This activity recurs throughout the life cycle.



There are many types of feasibilities:

- Technical
- Operational
- Economic
- Schedule
- Specification
- Information
- Motivational
- Legal and Ethical

#### **1.2.1. Technical Feasibility**

The technical feasibility of the Hostel Administration System is grounded in a well-defined system architecture and design. The project will adopt a microservices-based approach, comprising distinct frontend and backend components that communicate seamlessly through well-defined APIs. This approach ensures modular development, scalability, and flexibility. The technology stack, encompassing Spring Boot for the backend and React for the frontend, has been carefully selected to align with the project's requirements for robustness and extensibility. Database design entails the creation of separate databases for various modules, ensuring data integrity and efficient management. The choice of MySQL as the relational database management system stems from its established reliability and compatibility with the project's goals. By addressing these technical facets, the Hostel Administration System is poised to deliver a reliable, scalable, and secure solution that aligns closely with the project's goals.

#### **1.2.2. Operational Feasibility**

The operational feasibility of the Hostel Administration System is underpinned by its user-centric design and practical implementation strategies. The system's user interfaces have been carefully designed to be intuitive and easy to navigate, fostering user adoption and minimizing the learning curve. The inclusion of comprehensive hostel information, room details, and booking processes ensures a seamless user experience. The system's compatibility with various devices and browsers further enhances accessibility. By focusing on user-friendliness, adaptability, and streamlined administrative operations, the system demonstrates strong operational feasibility.

#### **1.2.3. Economic Feasibility**

By enhancing the efficiency of hostel operations, the system can lead to improved resource allocation and decreased operational expenses. Additionally, the system's user-friendly interface and streamlined booking mechanisms can attract a larger user base, potentially generating revenue through booking fees. This combination of reduced operational costs and increased revenue potential underscores the strong economic feasibility of the project.

#### **1.2.4. Schedule Feasibility**

The project's modular architecture allows for parallel development of various components, facilitating faster implementation. Moreover, potential risks and challenges

have been identified and addressed, contributing to a proactive approach to risk management. With a clear project roadmap and a commitment to meeting milestones, the project exhibits strong schedule feasibility.

#### **1.2.5. Specification Feasibility**

The system's functionalities, including user management, hostel registration, room allocation, and booking processes, correspond precisely to the outlined specifications. Regular communication with stakeholders and iterative development cycles ensure that the system adheres to specified features, fostering a clear and robust specification feasibility.

#### **1.2.6. Information Feasibility**

The system will incorporate verified hostel details, room descriptions, and boarder information, guaranteeing accuracy and relevancy. A robust data management approach, including database normalization and validation, ensures data integrity and reliability. By providing accurate and up-to-date information, the system maintains strong information feasibility.

#### **1.2.7. Motivational Feasibility**

The system streamlines the process of finding accommodations and assists hostel administrators in efficient management. By simplifying booking processes, enhancing user experience, and optimizing hostel operations, the project's goals align effectively with user motivations and needs, emphasizing its motivational feasibility.

#### **1.2.8. Legal & Ethical Feasibility**

The legal and ethical feasibility of the Hostel Administration System is upheld through adherence to data privacy laws, user consent, and industry best practices. User data will be securely managed and protected, complying with relevant data protection regulations. Ethical considerations, including transparency in data usage, user consent, and responsible data handling, are woven into the system's design. By prioritizing legal compliance and ethical considerations, the system demonstrates robust legal and ethical feasibility.

### **1.3. Project/Product Scope**

- The scope of the system consists of managing the boarders record with the admin side.
- A frontend public view which is used to add listing of hostel containing necessary details such as pictures, price etc.
- The system is well maintained and easy to use.
- The system allows hostel administrator to monitor room occupancy and room allocation etc.

### **1.4. Project/Product Costing**

The cost of this project is considered none. The only cost we can consider is the project development cost. The cost of development is primarily the cost of the efforts involved.

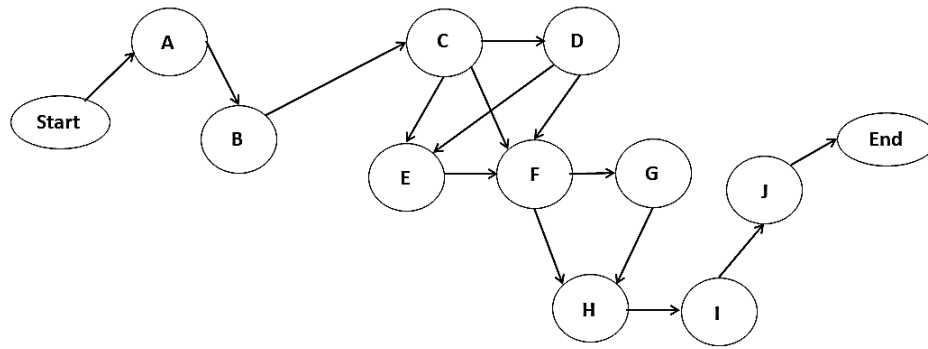
All other costs for the project e.g., documentation, testing and designing cannot be included.

### 1.5. Task Dependency Table

Activity	Immediate Predecessor	Duration (Weeks)
Idea hunting (A)	None	1
Proposal defense (B)	A	2
Requirement Gathering & Analysis (C)	B	2
Software Requirements Specification (D)	C	1
Consultation with supervisor (E)	C, D	1
Enlisting database tables and entities (F)	C, D, E	3
Consultation with supervisor (G)	F	1
Database development (H)	F, G	5
UI & Coding (I)	H	10
Deploy and Testing (J)	I	5

### 1.6. CPM - Critical Path Method

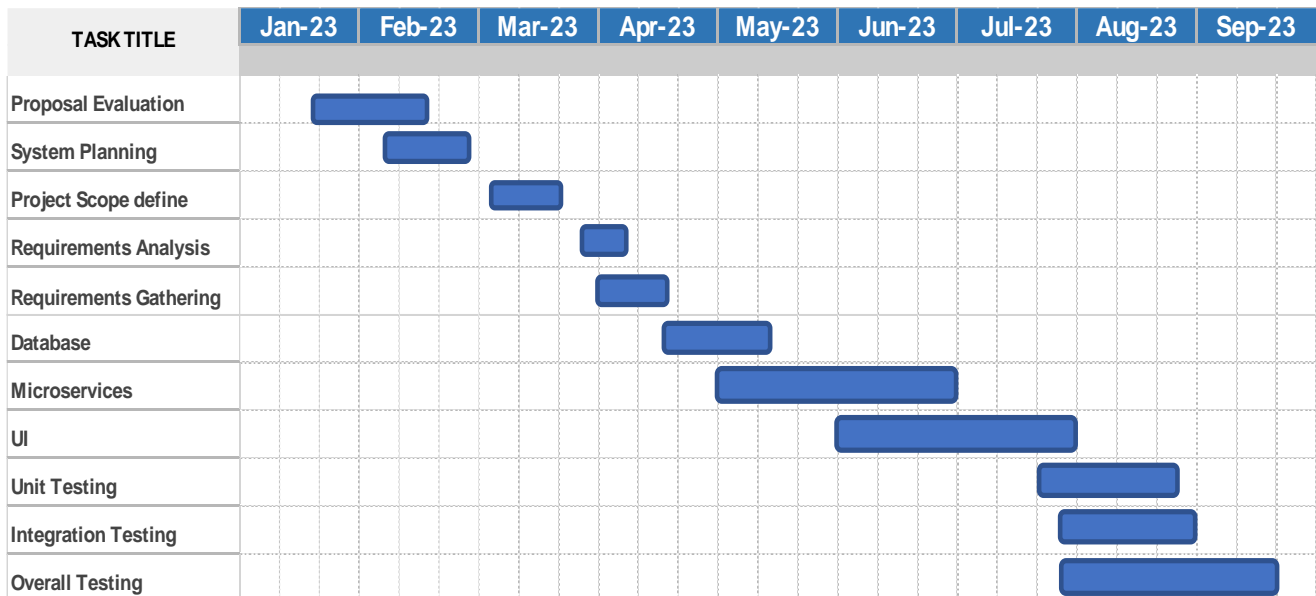
Activity	Immediate Predecessor	Duration (Weeks)
Idea hunting (A)	None	1
Proposal defense (B)	A	2
Requirement Gathering & Analysis (C)	B	2
Software Requirements Specification (D)	C	1
Consultation with supervisor (E)	C, D	1
Enlisting database tables and entities (F)	C, D, E	3
Consultation with supervisor (G)	F	1
Database development (H)	F, G	5
UI & Coding (I)	H	10
Deploy and Testing (J)	I	5



Network Diagram of above activities

## 1.7. Gantt chart

The Gantt chart enumerates the activities to be performed on the vertical axis and their corresponding duration on the horizontal axis.



## 1.8. Allocation of Members to Activities

Task	Member	Duration (Days)	Dependencies
Idea hunting (A)	Maaz, Ryan, Shahzaib	7	None
Proposal defense (B)	//	12	A
Requirement Gathering & Analysis (C)	//	11	B
Software Requirements Specification (D)	//	6	C

Consultation with supervisor (E)	//	6	C, D
Enlisting database tables and entities (F)	//	20	C, D, E
Consultation with supervisor (G)	//	7	F
Database development (H)	//	35	F, G
UI & Coding (I)	//	74	H
Deploy and Testing (J)	//	25	I
Documentation (K)	//	15	All

## 1.9. Tools and Technology with reasoning

### Tools

Visual Studio Code: Code editor used for coding

Spring Tool Suite: For micro services development (Backend)

Postman: For API's Testing

MySQL Workbench: Tool for database

### Technologies

Spring Framework: Develop Micro-services

Angular/React Framework: To develop frontend

- Spring Boot is a framework and tool suite for developing and deploying Spring-based applications quickly and with very little configuration.
- The Spring Tool Suite 3 supports application targeting to local, virtual, and cloud-based servers.

The programming language(s) to be used are: Angular is one of the most popular JavaScript frameworks for building web applications. Angular is not only the best solution, but the only scalable one that can be used no matter how large you predict your application is developed. Html, CSS and JavaScript will be under use.

## 1.10. Vision Document

### **Vision Statement:**

To create a transformative Hostel Administration System that revolutionizes the way individuals find and manage hostel accommodations, offering a seamless and user-centric experience for both users and hostel administrators.

### **Objectives:**

1. **Simplify Accommodation:** Our vision is to simplify the process of finding, booking, and managing hostel accommodations, making it as effortless as possible for students, working professionals, and hostel administrators.
2. **Enhance User Experience:** We aim to provide an intuitive and user-friendly platform that allows users to explore and book hostels with ease, reducing the stress associated with hostel hunting.

3. **Empower Hostel Administrators:** We aspire to empower hostel administrators with efficient tools and insights to streamline their operations, optimize room allocation, and enhance the overall hostel experience for boarders.
4. **Data Security:** We are committed to ensuring the security and privacy of user data, adhering to legal and ethical standards, and safeguarding sensitive information.
5. **Scalability:** Our system is designed to scale with the growing needs of both users and hostel administrators, accommodating future expansions and enhancements.
6. **Innovation:** We aim to continually innovate and integrate emerging technologies to stay at the forefront of the hostel accommodation industry.

#### **Benefits:**

- For Users: A hassle-free and transparent experience in finding and booking hostels, saving time and reducing stress.
- For Hostel Administrators: Streamlined hostel management, optimized room allocation, and improved occupancy rates.
- For Society: Enhancing the overall accommodation landscape, promoting ease of access, and fostering trust and transparency.

#### **Key Principles:**

- User-Centric Design: Prioritizing the needs and preferences of users.
- Efficiency: Streamlining processes and operations for maximum productivity.
- Data Privacy: Ensuring the highest standards of data security and user privacy.
- Innovation: Embracing technological advancements for continuous improvement.

#### **Checkpoints:**

- Have you fully explored what the "problem behind the problem" is? YES
- Is the problem statement correctly formulated? YES
- Is the list of stakeholders complete, and correct? YES
- Does everyone agree on the definition of the system boundaries? YES
- If system boundaries have been expressed using actors, have all actors been defined and correctly described? YES
- Have you sufficiently explored constraints to be put on the system? YES
- Have you covered all kinds of constraints - for example political, economic, and environmental? YES
- Have all key features of the system been identified and defined? YES
- Will the features solve the problems that are identified? YES
- Are the features consistent with constraints that are identified? YES

### **1.11. Product Features/ Product Decomposition**

Our system will be useful and users will be able to easily search for hostel accommodations in their desired area with all the necessary details available at their fingertips. This saves time and effort compared to traditional methods of searching for accommodations. The system will only list verified hostels with all the necessary information, ensuring that users have access to reliable and safe accommodations. The system provides an interactive and engaging user experience, making it easier and more convenient for users to find hostel accommodations. By digitizing the hostel

administration process, the system can increase the efficiency of hostel management, reducing the workload for hostel owners and managers. The common features are listed below:

- Management of registered hostels and their properties.
- Managing individual rooms within hostels.
- This system will reduce the paper usage.
- Hostel owners can add and edit their hostel properties through the admin panel.

A user-friendly interface for easy searching and filtering of hostels.

## **Chapter 2: Software Requirement Specification (For Object Oriented Approach)**

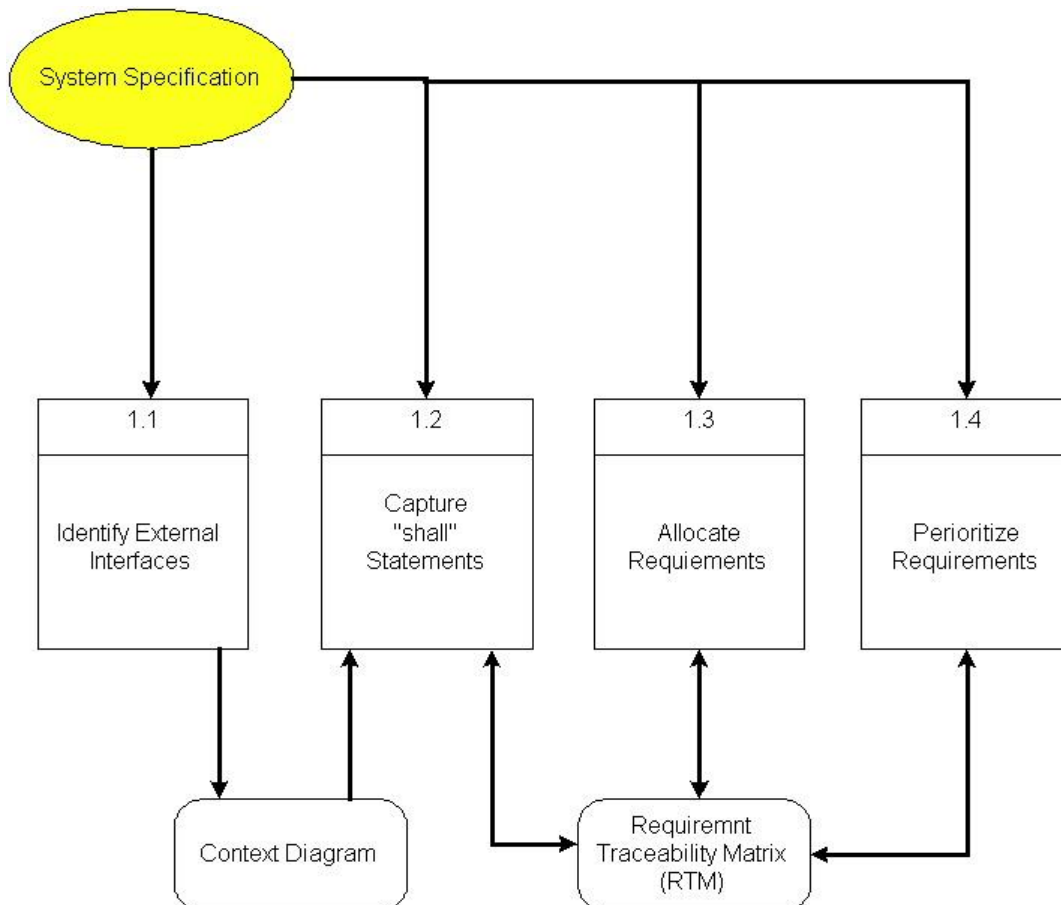
---



## 2.1 Introduction:

Requirements engineering process provides the appropriate mechanism for understanding what the customer wants, analyzing need, assessing feasibility, negotiating a reasonable solution, specifying the solution unambiguously, validating the specification and managing the requirements as they are transformed into an operational system. The task of capturing, structuring, and accurately representing the user's requirements so that they can be correctly embodied in systems which meet those requirements (i.e. are of good quality).

- Requirements elicitation
- Requirements analysis and negotiation
- Requirements specification
- System modeling
- Requirements validation
- Requirements management



Here, requirements specification is to be discussed. Requirement's specification would lead to the following four steps:

- Identify external interfaces

- Development of context diagram
- Capture “shall statements
- Allocate requirements
- Prioritize requirements

Development of requirements traceability matrix

## 2.2 Systems Specifications

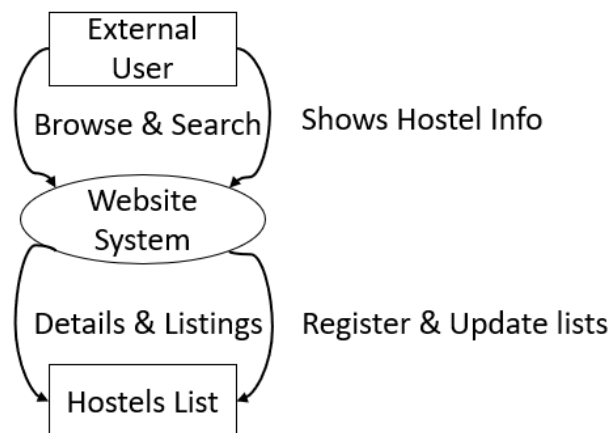
The following are the clauses that must be included while describing the system specifications.

### 2.2.1. Identifying External Entities

Here are some potential external entities:

1. **Users:** These are individuals who access the system to search for hostels, make bookings, and manage their profiles. Users include students, working professionals, and anyone seeking hostel accommodations.
2. **Hostel Administrators:** Hostel administrators are individuals responsible for managing and maintaining the hostel facilities. They use the system to register their hostels, manage room information, and keep records of boarders.
3. **External Email Services (future):** Our system may use external email services (e.g., SMTP servers) to send email notifications and confirmations to users and administrators.
4. **External Messaging Services (future):** Our system includes messaging or notification features, external messaging services (e.g., SMS gateways) can be used to send alerts and messages to users.

### 2.2.2. Context Level Data Flow Diagram:



- **External User:** A person who want to search for hostels.
- **Hostels List:** Listing of available hostels.
- **Website System:** The central system (process) that connects external users with local businesses. It displays business information to users and provides notifications & stats to the registered businesses.

### 2.2.3. Capture "shall" Statements:

Identify “shall” statements, as they would be all functional requirements.

Para #	Initial Requirements
1.0	A customer “shall” browse the listings
1.0	A customer “shall” register himself to the system
1.0	The system “shall” provide two types of registration process, normal and privileged
1.0	CA “shall” accept, reject and temporarily waive the requests on the basis of credentials provided.
1.0	A customer “shall” login to the system and can change his password
1.0	System “shall” update the customer’s Request
1.0	System “shall” process different types of updating
1.0	A customer “shall” view his details for verification purposes
1.0	CA “shall” accept, reject and temporarily waive the requests on the basis of credentials provided.
1.0	System “shall” search any customer details
2.0	Both registered and privileged customers “will” browse for listings
2.0	Customer “shall” make reservation request
3.0	An action event "shall" be generated for a corresponding administrator when a request is placed for reservation of hostel
3.0	Corresponding administrator "shall" view his Action List containing different actions, and correspondingly process these pending actions
3.0	When the action processing is completed or if the action is just a notification message, then administrator "shall" delete these actions from the action list

#### 2.2.4 Allocate Requirements:

Allocate the requirements in the use cases.

Para #	Initial Requirements	Name
1.0	A customer “will” make reservation request	UC_Reservation_Request
1.0	A customer “shall” register himself to the system	UC_Registration_Request
1.0	The system “shall” provide two types of registration process, normal and privileged	UC_Place_Order_Request
1.0	CA “shall” accept, reject and temporarily waive the requests on the basis of credentials provided.	UC_Process_Customer_Request
1.0	A customer “shall” login to the system and can change his password	UC_Login
1.0	System “shall” update the customer’s Request	UC_Update_Request
1.0	System “shall” process different types of updating	UC_Change_Status
1.0	A customer “shall” view his details for verification purposes	UC_View_Customer_Details
1.0	System “shall” search any customer details	UC_Search_Customer
2.0	Both registered and privileged customers “will” browse for listings	UC_Reservation_Request_Privileged
2.0	User “shall” view the status of their reservation requests	UC_Search_Reservations
2.0	Privileged customers “shall” place the request for the updating of their orders if the reservation request is not approved.	UC_Update_Request
3.0	The System “shall” generate an action event for a corresponding administrator when a request is placed for updating of reservation or customer details	UC_Create_Action,
3.0	Corresponding administrator “shall” view his Action List containing different actions, and correspondingly process these pending actions	UC_View_Action,

### 2.2.5. Prioritize Requirements (If Any):

Para #	Rank	Initial Requirements	Use Case ID	Use Case Name
1.0	Highest	A customer “will” make reservation request	UC_1	UC_Reservation_Request
1.0	Highest	A customer “shall” register himself to the system	UC_2	UC_Registration_Request
2.0	Medium	Both registered and privileged customers “will” browse for listings	UC_4	UC_Reservation_Request_Privileged
1.0	Medium	The system “shall” provide two types of registration process, normal and privileged	UC_5	UC_Reservation_Request_Request
3.0	Medium	The System “shall” generate an action event for a corresponding administrator when a request is placed for updating request	UC_6	UC_Create_Action
1.0	Medium	CA “shall” accept, reject and temporarily waive the requests on the basis of credentials provided.	UC_7 UC_8 UC_9	UC_Accept_Customer_Request UC_Reject_Customer_Request UC_View_Customer_Request
1.0	Medium	System “shall” update the customers Request	UC_10	UC_Update_Request

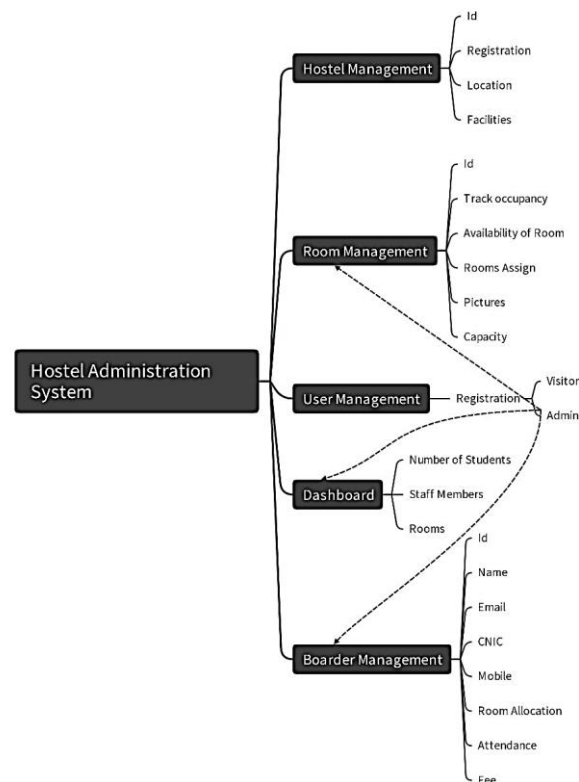
1.0	Medium	System “shall” process different types of updating e.g. updating of his personal/shipping details, or upgrading of his status from registered to privileged customer	UC_11 UC_12	UC_Change_Status, UC_Change_Personal_Details
1.0	Medium	A customer “shall” view his details for verification purposes	UC_13	UC_View_Customer_Details
1.0	Medium	System “shall” search any customer details	UC_14	UC_Search_Customer
2.0	Medium	User “shall” view the status of their orders by providing the reserve num	UC_15	UC_Serach_Requests
2.0	Medium	Privileged customers “shall” place the request for the updating of their orders if the request is not approved	UC_16	UC_Update_Request
1.0	Lowest	A customer “shall” login to the system and can change his password	UC_17 UC_18	UC_Login,
3.0	Lowest	Corresponding administrator “shall” view his Action List containing different actions, and correspondingly process these pending actions	UC_19	UC_View_Action,

3.0	Lowest	When the action processing is completed or if the action is just a notification message then administrator “shall” delete these actions from the action list	UC_20	UC_Delete_Action
-----	--------	--	-------	------------------

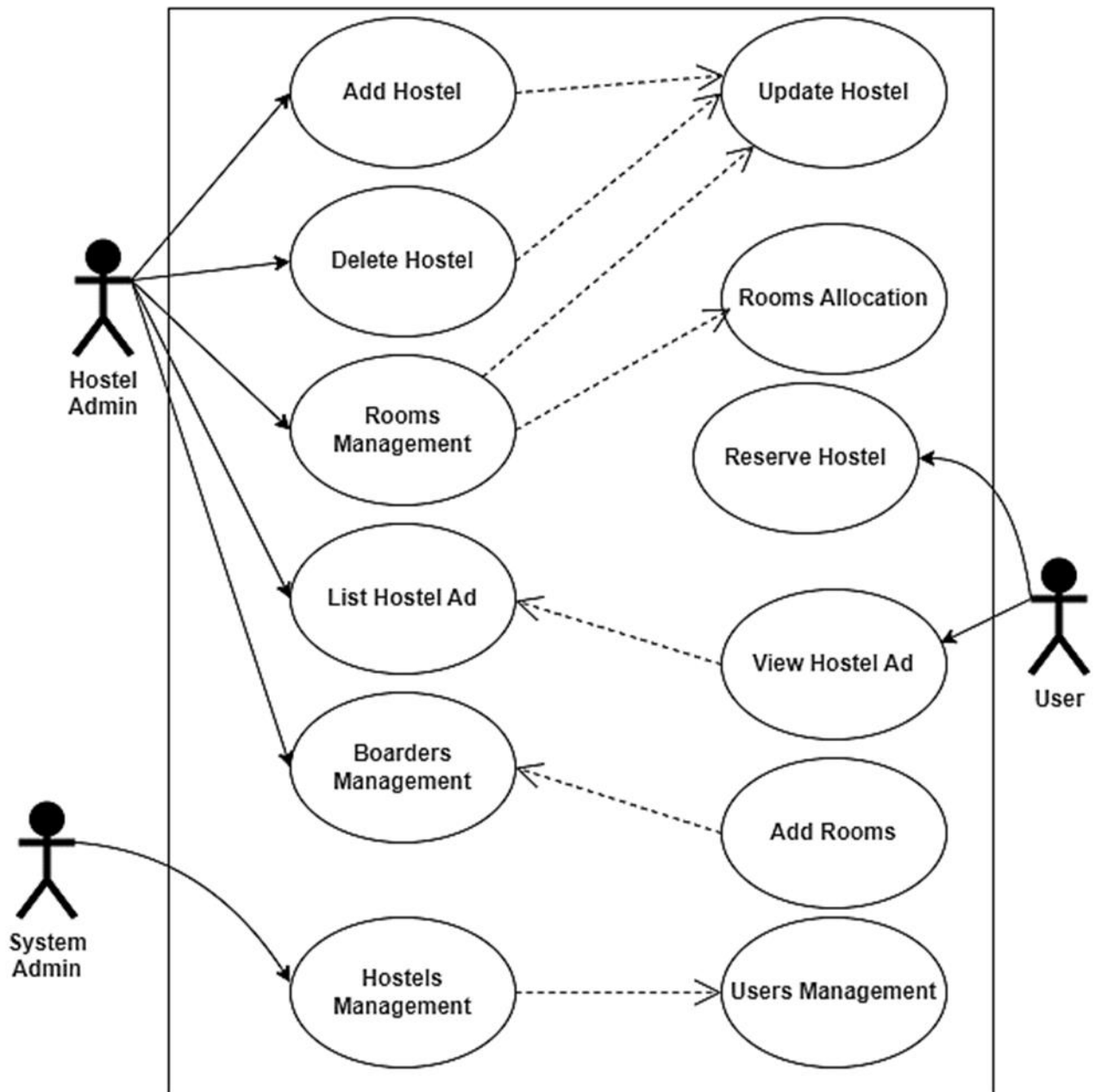
## 2.3. Existing Systems / Literature Review:

Here is an example to explain all the above. We are taking the system of Green Wood Company.

### 2.3.1. Existing System



## 2.4. Usecase Diagram of Your Project:





#### 2.4.1. Usecase Description

UC_Login	
Use Case Name	UC_Login
Use Case ID	UC_01
Primary Actor	General User, Hostel Admin, System Admin
Secondary Actor	System
Dependency	None
Description	This Use Case is used by user to login to system.
Basic Flow	<ul style="list-style-type: none"><li>• User selects log in</li><li>• User enters his username and password and clicks on submit button.</li><li>• The account will be verified. User will be redirected to its profile.</li></ul>
Pre-Conditions	User should have an account.
Post-Conditions	User will be shown his main page where he can post job.
Extension	The account is not a valid.

UC_Signup	
Use Case Name	UC_Signup
Use Case ID	UC_02
Primary Actor	General User, Hostel Admin, System Admin
Secondary Actor	System
Dependency	None
Description	This Use Case is used by user to Sign up in system
Basic Flow	<ul style="list-style-type: none"><li>• User selects Sign up</li><li>• User enters his data and clicks on submit button.</li><li>• The account will be created. User will be redirected to login screen.</li></ul>
Pre-Conditions	System is working perfectly.

Post-Conditions	User will be redirected to login screen.
Extension	The account is not created.

<b>UC_PostPoperty</b>	
Use Case Name	UC_PostPoperty
Use Case ID	UC_03
Primary Actor	Hostel Admin
Secondary Actor	System
Dependency	None
Description	This Use Case is used by Hostel Admin to post the hostel property.
Basic Flow	<ul style="list-style-type: none"> <li>Hostel Admin logged in the system.</li> </ul>
Pre-Conditions	Hostel Admin must be logged in.
Post-Conditions	Hostel Admin will be shown his listing.
Extension	

Name of Use Case:	UC_Data_Storage_Request
Description:	This module will make sure user data I
Actors:	1. User 2. System
Preconditions:	1. User is logged in
Post conditions:	<ul style="list-style-type: none"> <li>User is logged in</li> </ul>
Flow:	<ul style="list-style-type: none"> <li>User is displayed dashboard upon login</li> <li>User selects profile info</li> <li>Profile info is shown on screen</li> </ul>
Exceptions:	<ul style="list-style-type: none"> <li>No Exception</li> </ul>

## **Chapter 3: Design Document (For Object Oriented Approach)**

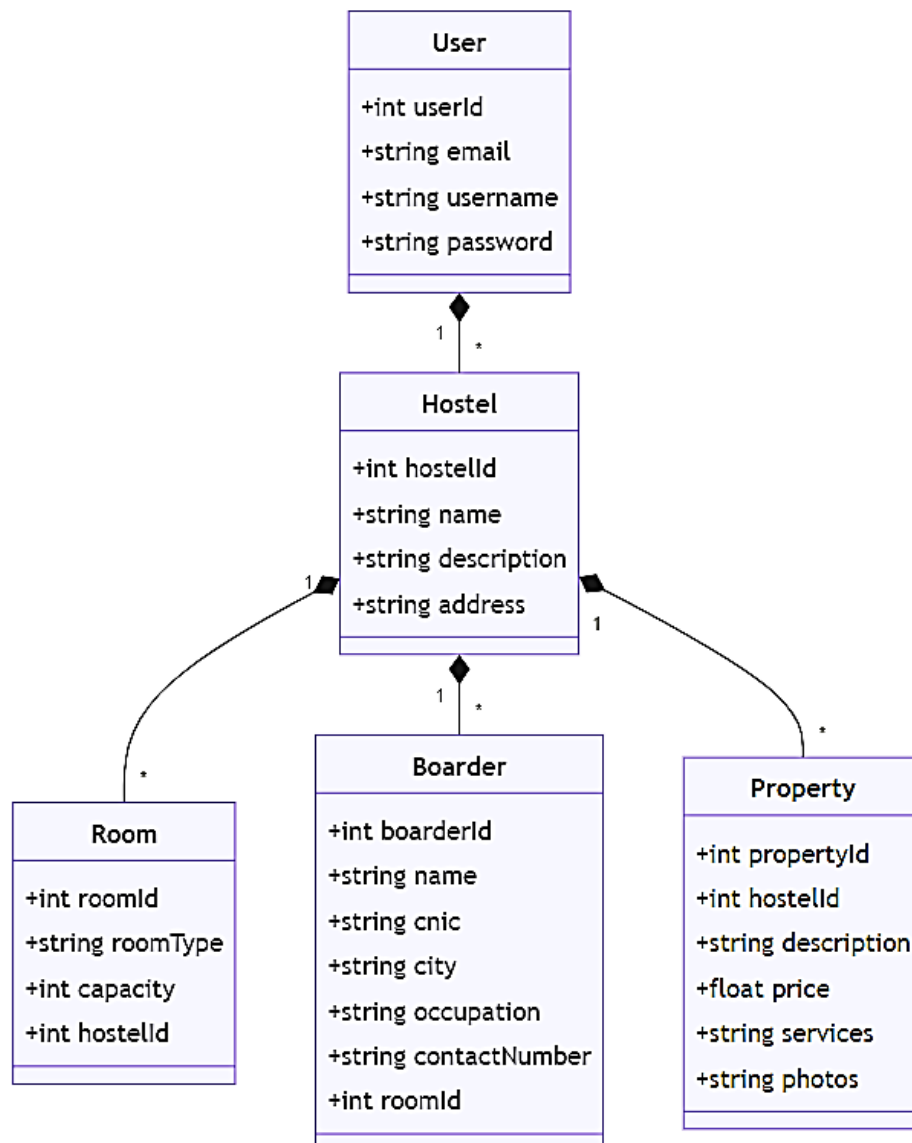
---

### 3.1. Introduction:

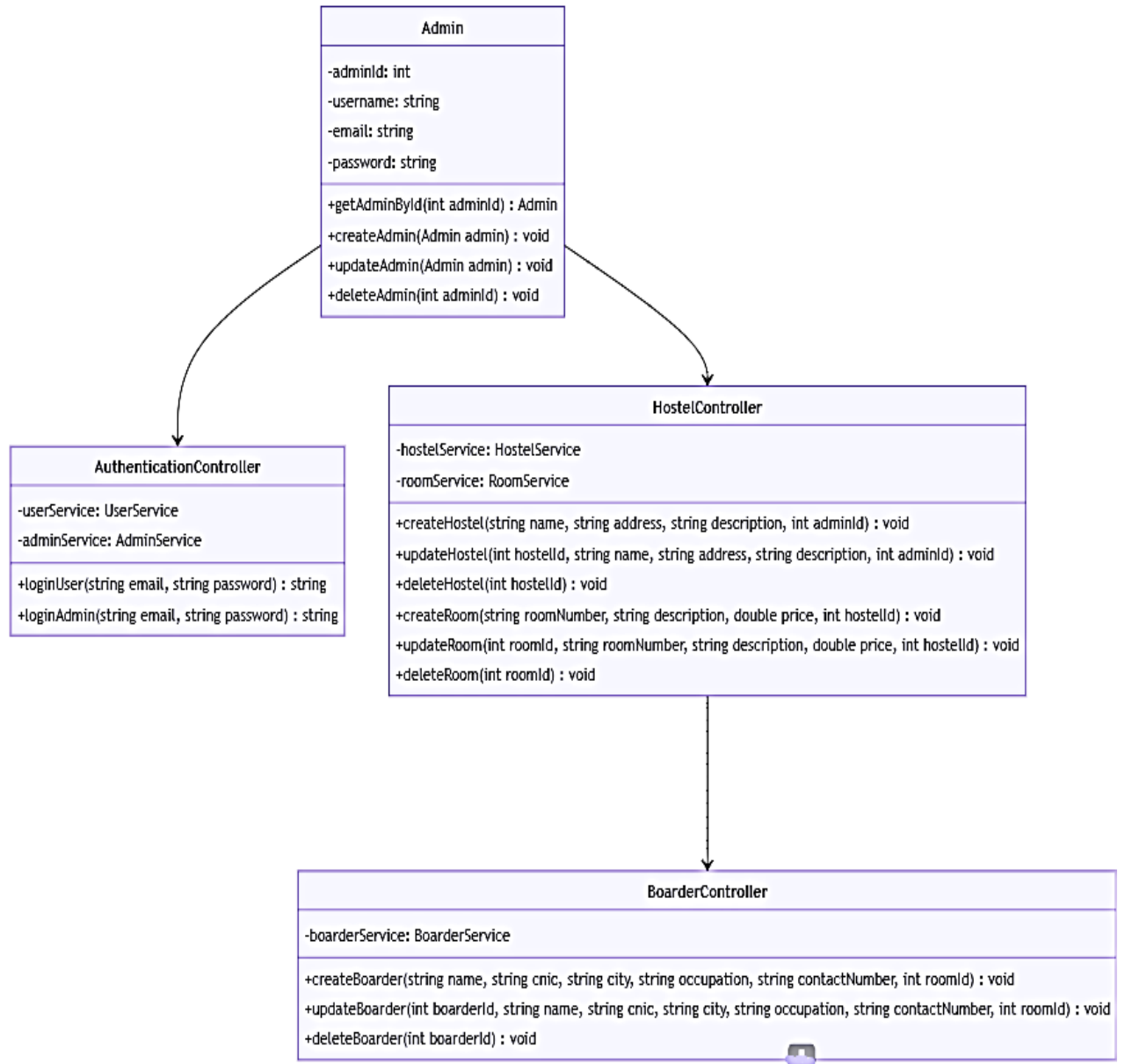
Domain model is a visual representation of the entities and relationships involved in a particular domain or subject area. It is used to describe the various concepts, terms, and relationships within a particular system or business. In the context of the hostel administration system, the domain model represents the different entities involved in managing the hostels, such as hostels, rooms, boarders, admins, and bookings.

Now we discuss these artifacts one by one as follows:

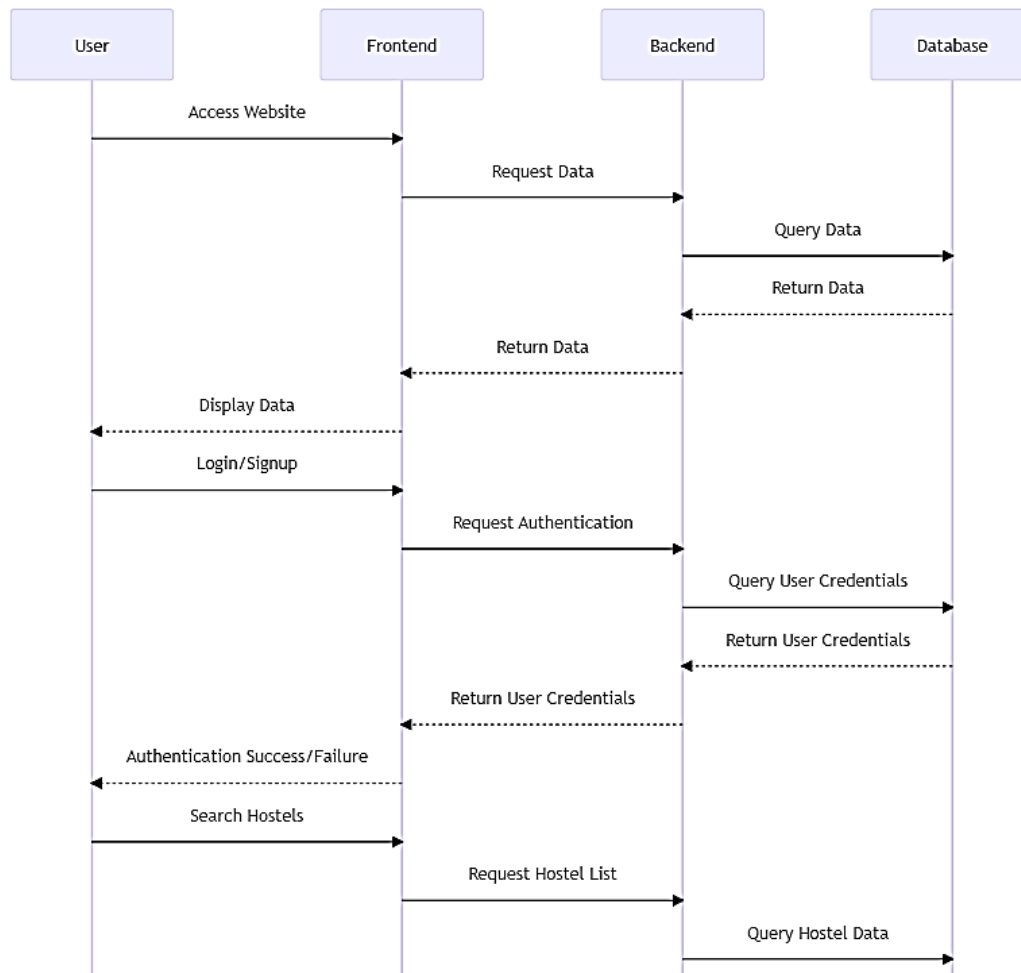
### 3.2. Domain Model

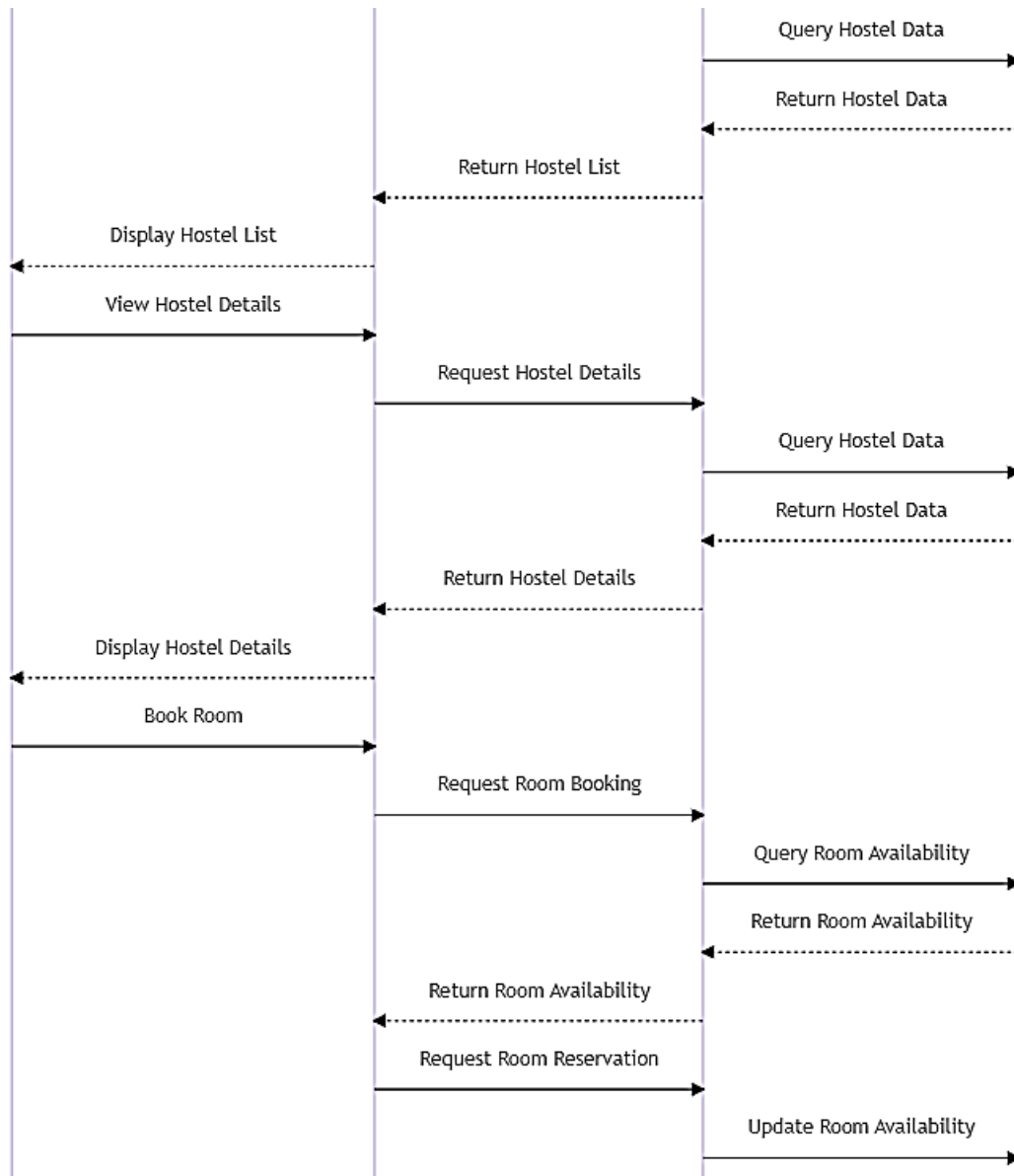


### 3.3. Design Class Diagram

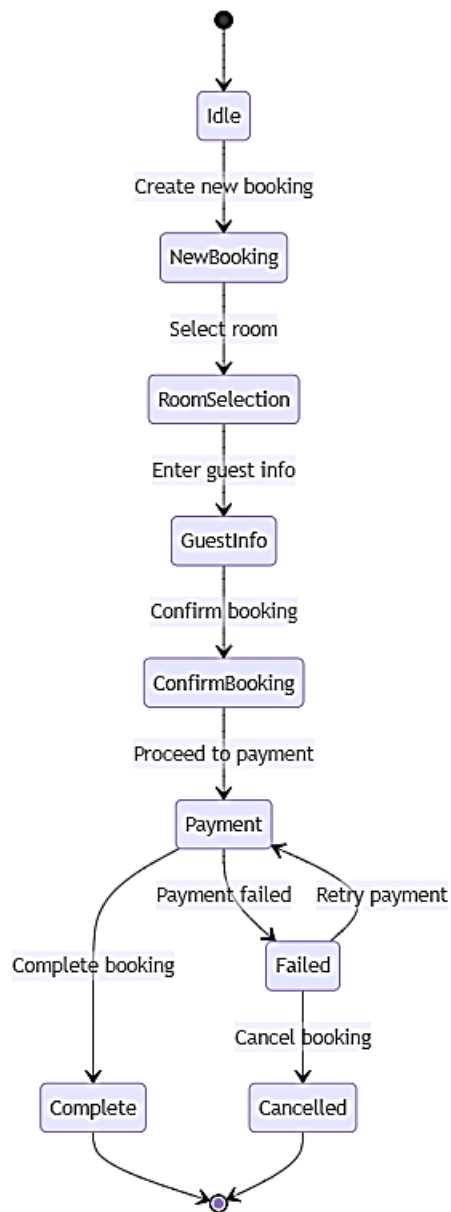


### 3.4. Sequence Diagram





### 3.5. State chart diagram



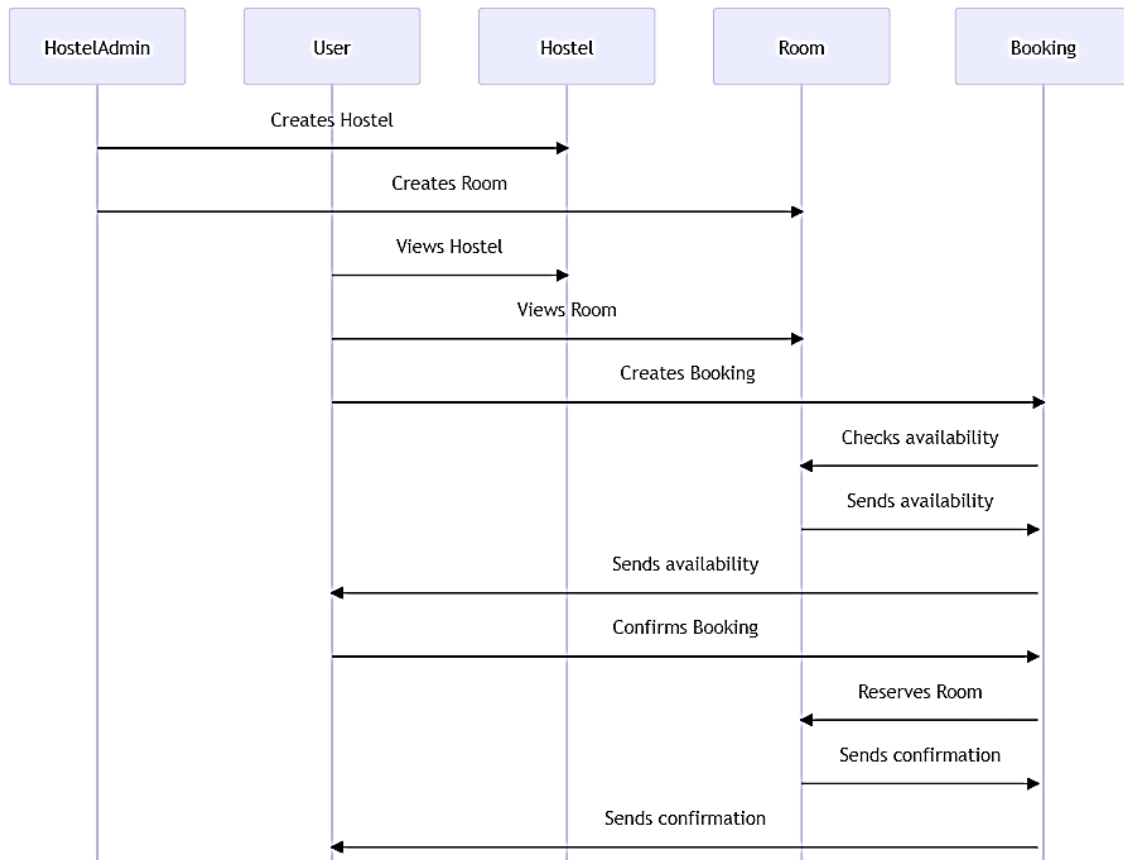


### 3.6. Collaboration Diagram

A collaboration diagram describes a pattern of interaction among objects; it shows the objects participating in the interaction by their links to each other and the messages that they send to each other.

#### a. Contents of Collaboration Diagrams

You can have objects and actor instances in collaboration diagrams, together with links and messages describing how they are related and how they interact. The diagram describes what takes place in the participating objects



## **Chapter 4: User Interface Design**

---

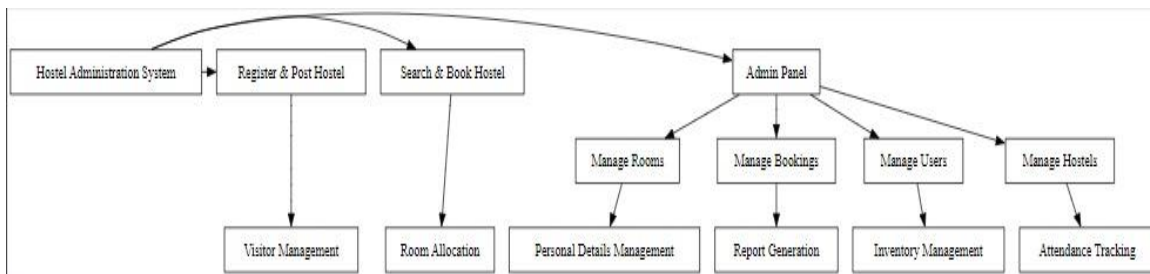
## 4.1. Introduction

A user interface design consists of three main parts:

Page elements should be visualized on paper before building them in the computer. Just as you draw a site map to plan the site, use cartoons and storyboards to begin blocking out the site's appearance and navigational scheme.

1. Site maps
2. Storyboards
3. Navigational maps

## 4.2. Site Maps



### Explanation:

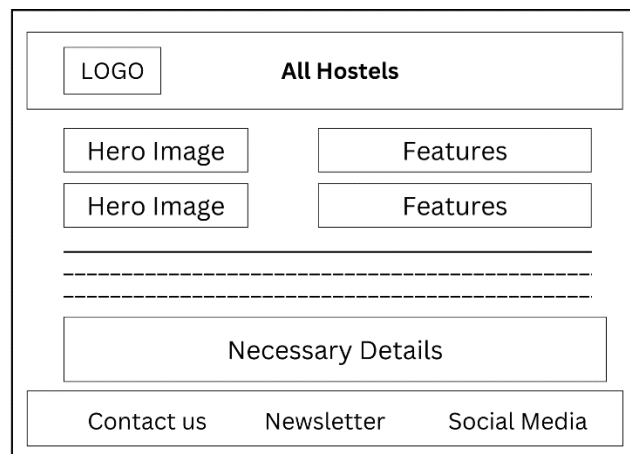
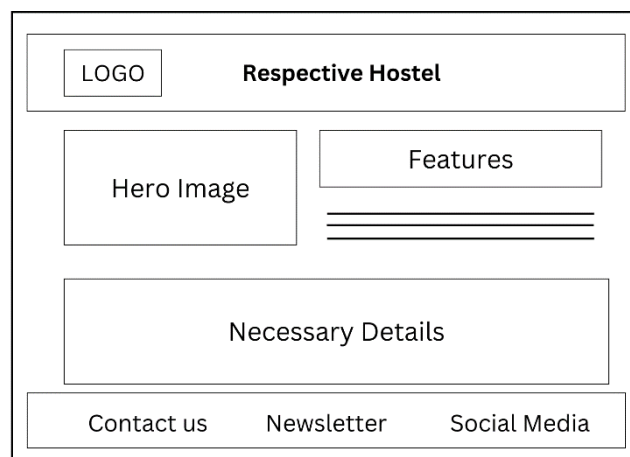
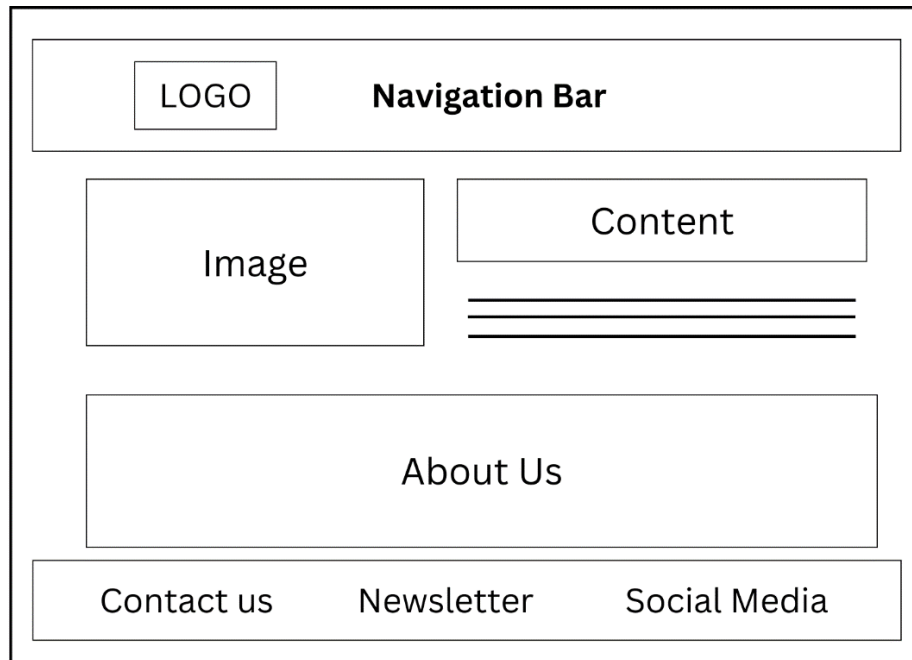
**Hostel Registration System (Home):** The main home Page where other options are available.

**Register and post hostel:** Where hostel admin can post the listings of his hostel.

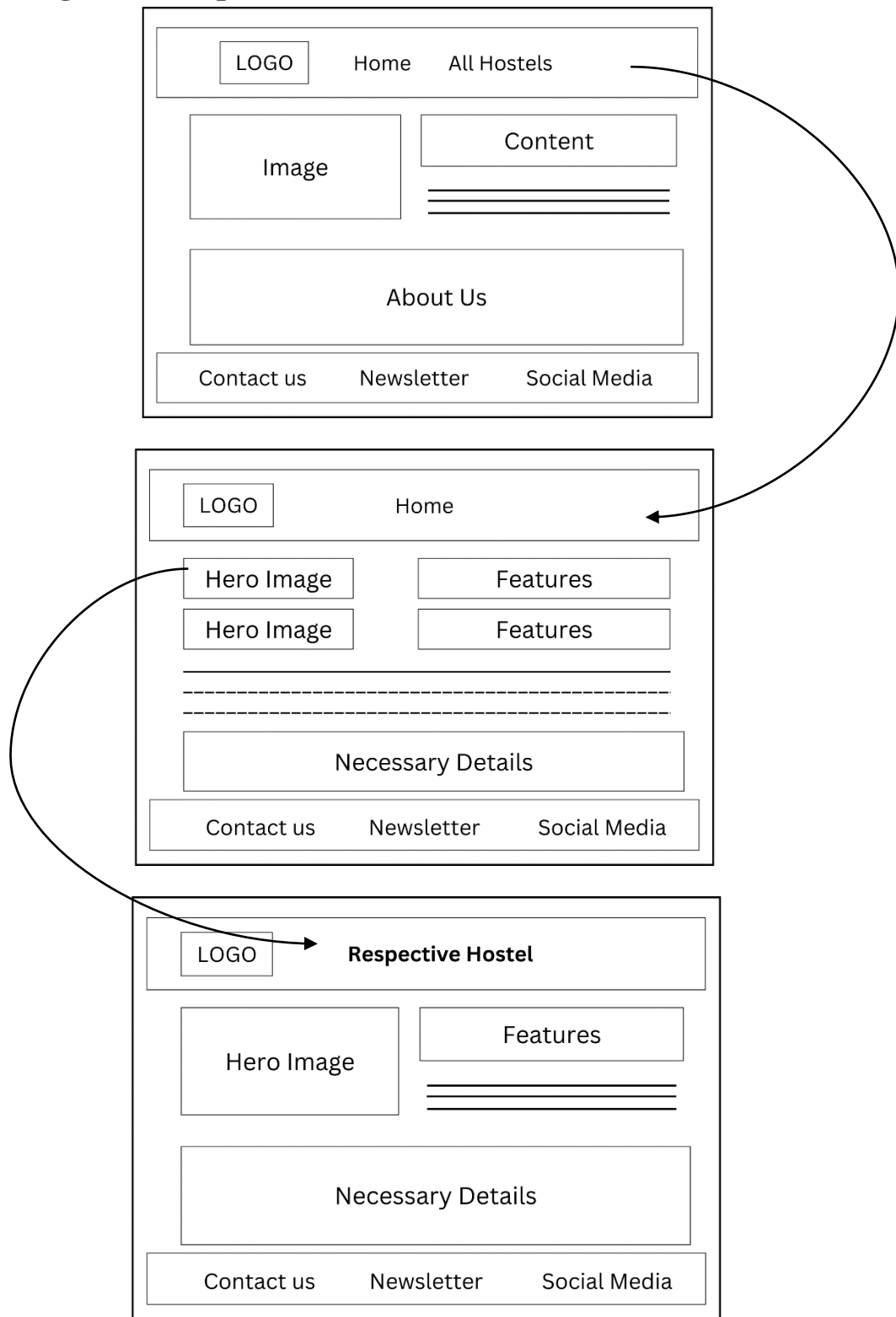
**Search and Book Hostel:** User can search for hostels and can make reservation request.

**Admin Panel:** A complete admin panel for management where all access controls are present.

### 4.3. Story boards



#### 4.4. Navigational maps:



## **Chapter 5: Software Testing**

---

## 5.1 Introduction:

This deliverable is based on the IEEE standard of software testing i.e. IEEE SOFTWARE TEST DOCUMENTATION Std 829-1998. This standard describes a set of basic test documents that are associated with the dynamic aspects of software testing (i.e., the execution of procedures and code). The standard defines the purpose, outline, and content of each basic document. While the documents described in the standard focus on dynamic testing, several of them may be applicable to other testing activities (e.g., the test plan and test incident report may be used for design and code reviews). This standard may be applied to commercial, scientific, or military software that runs on any digital computer. Applicability is not restricted by the size, complexity, or criticality of the software. However, the standard does not specify any class of software to which it must be applied. The standard addresses the documentation of both initial development testing and the testing of subsequent software releases. For a particular software release, it may be applied to all phases of testing from module testing through user acceptance. However, since all of the basic test documents may not be useful in each test phase, the particular documents to be used in a phase are not specified. Each organization using the standard will need to specify the classes of software to which it applies and the specific documents required for a particular test phase.

The standard does not call for specific testing methodologies, approaches, techniques, facilities, or tools, and does not specify the documentation of their use. Additional test documentation may be required (e.g., code inspection checklists and reports). The standard also does not imply or impose specific methodologies for documentation control, configuration management, or quality assurance. Additional documentation (e.g., a quality assurance plan) may be needed depending on the particular methodologies used.

Following are standard artifacts, which must be included in this deliverable:

1. Test Plan
2. Test Design Specification
3. Test Case Specification
4. Test Procedure Specification
5. Test Item Transmittal Report
6. Test Log
7. Test Incident Report
8. Test Summary Report

## 5.2. Test plan

### 5.2.1. Purpose

To prescribe the scope, approach, resources, and schedule of the testing activities. To identify the items being tested, the features to be tested, the testing tasks to be performed, the personnel responsible for each task, and the risks associated with this plan.

### 5.2.2. Outline

A test plan for **Hostel Administration System (HAS)**:

1. **HAS Test Plan Identifier:** A unique identifier for Hostel Administration System's (HAS) test plan, ensuring easy tracking and referencing.
2. **HAS Introduction:** An overview of HAS's objectives for testing and the significance of this test plan in ensuring the website's functionality and user experience.
3. **HAS Test Items:** List of specific components, modules, or features of the HAS website that will be subjected to testing.
4. **HAS Features to be Tested:** Detailed outline of the specific features of the HAS website that will undergo testing.
5. **HAS Features Not to be Tested:** Features of the HAS website that are excluded from this testing cycle.
6. **HAS Approach:** The methodology HAS will employ for testing, whether it's manual, automated, or a hybrid approach.
7. **HAS Item Pass/Fail Criteria:** Criteria that will determine the success or failure of tested items on the HAS website.
8. **HAS Suspension Criteria and Resumption Requirements:** Conditions under which HAS's testing will be halted and the prerequisites for resuming the tests.
9. **HAS Test Deliverables:** Outputs from HAS's testing process, such as detailed reports, logs, and lists of identified issues.
10. **HAS Testing Tasks:** Sequential tasks or steps that will be undertaken during the testing of the HAS website.
11. **HAS Environmental Needs:** Requirements for HAS's testing environment, including specific configurations and setups.
12. **HAS Responsibilities:** Roles and duties assigned to HAS team members involved in the testing process.
13. **HAS Staffing and Training Needs:** Identification of personnel and training requirements for HAS's testing phase.
14. **HAS Schedule:** Timelines detailing when each testing activity for HAS will be conducted.
15. **HAS Risks and Contingencies:** Potential challenges HAS might face during testing and the backup plans in place to address them.

**HAS Approvals:** Section for obtaining validation and sign-offs from HAS stakeholders to commence with the test plan.

#### 5.2.2.1. Test plan identifier

##### **Identifier: HAS-TP-2023-1001**

This identifier, **HAS-TP-2023-1001**, is a unique code assigned to this specific test plan for Hostel Administration System. The format suggests:

- **HAS:** Denotes that this test plan is specifically for the HAS website.
- **TP:** Stands for "Test Plan".
- **2023:** Represents the year the test plan is created or intended for.
- **1001:** A sequential number, indicating that this is the first test plan for that year. This number will increment with each subsequent test plan created within the same year.



#### 5.2.2.2. Introduction

##### Software Items and Features to Be Tested:

1. **User Registration and Login:**
  - **Need:** To allow users to create accounts, log in securely, and access personalized features.
  - **History:** Developed as a fundamental user management feature for system access.
2. **Search for Hostels:**
  - **Need:** To enable users to find hostels based on various criteria, streamlining the hostel search process.
  - **History:** Introduced to enhance the user experience and simplify hostel discovery.
3. **View Hostel Details:**
  - **Need:** To provide users with comprehensive information about hostels, including room types, rates, and amenities.
  - **History:** Implemented to improve user decision-making when selecting accommodations.
4. **Make a Booking:**
  - **Need:** To allow users to reserve rooms, providing a convenient and efficient booking process.
  - **History:** Developed to facilitate secure and straightforward room reservations.
5. **Hostel Registration:**
  - **Need:** To enable hostel administrators to register their properties and manage hostel details.
  - **History:** Introduced to empower hostel administrators and expand the system's database of hostels.

##### Software Features to be Tested:

1. **Responsive Design:** Ensures the HAS web app is accessible and user-friendly on various devices, including mobiles and tablets.
2. **Post Property feature:** Allowing hostel admins to list the hostel property for customers.
3. **Public End:** Ensures the visibility of listed properties to customers.

##### References:

- **Project Authorization:** Document that provides the formal go-ahead for the Hostel Administration System project.
- **Project Plan:** Outlines the timeline, milestones, resources, and objectives for the Hostel Administration System project.
- **Quality Assurance Plan:** Details the quality standards, testing procedures, and criteria for the Hostel Administration System platform.
- **Configuration Management Plan:** Describes how changes to the Hostel Administration System web app will be managed and documented.
- **Relevant Policies:** Includes data protection policy, user privacy policy, and terms of service for Hostel Administration System.
- **Relevant Standards:** Standards that the Hostel Administration System platform adheres to, such as ISO/IEC 25010 for software quality.

For multilevel test plans, this test plan (Level 1) will serve as a reference for subsequent lower-level test plans (Level 2, Level 3, etc.), ensuring a cohesive and comprehensive testing approach for the PNC platform.

#### 5.2.2.3. Test items

##### 1. User Registration and Login:

- **Version:** 1.0
- **Transmittal Media Characteristics:** This component enables user registration and login.

##### 2. Search for Hostels:

- **Version:** 1.0
- **Transmittal Media Characteristics:** This feature allows users to search for hostels based on various criteria.

##### 3. View Hostel Details:

- **Version:** 1.0
- **Transmittal Media Characteristics:** Users can view detailed information about hostels.

##### 4. Hostel Registration:

- **Version:** 1.0
- **Transmittal Media Characteristics:** Hostel administrators can register their properties.

#### References to Test Item Documentation:

a) **Requirements Specification:** Document detailing the functional and non-functional requirements of the Hostel Administration System platform. It provides a clear understanding of the expected behavior of the system.

b) **Design Specification:** Provides a comprehensive overview of the architectural and design aspects of the Hostel Administration System platform, including database design, user interface design, and system architecture.

c) **Users Guide:** A manual that guides end-users on how to navigate and utilize the features of the Hostel Administration System platform effectively.

d) **Operations Guide:** Document tailored for system administrators, detailing the operational aspects of the Hostel Administration System platform, including maintenance, backup, and troubleshooting.

e) **Installation Guide:** Provides step-by-step instructions on how to install and set up the Hostel Administration System platform on various environments.

#### Incident Reports:

- **IR-123:** Incident related to the User Registration Module where certain fields were not saving correctly.
- **IR-456:** Incident where the search functionality was returning incorrect results for specific queries.
- **IR-789:** Incident regarding the reservation request where users were unable to edit the request.

#### Exclusions from Testing:

- **Online Payment Module:** This module is planned for future and will not be part of this testing cycle.

- **Newsletter System:** Due to third-party integration changes, this system will be tested in a separate cycle.

By identifying the test items, their versions, and the necessary documentation, we ensure a structured and comprehensive approach to testing the Hostel Administration System platform.

#### 5.2.2.4. Features to be tested

##### 1. User Registration Module:

- **Features:**
  - User sign-up
  - Profile creation
  - Email verification
- **Test Design Specification: TDS-URM-1001**

This specification will detail the test cases for user input validation, successful registration, error handling, and email verification process.

##### 2. Search Functionality:

- **Features:**
  - Keyword-based search
  - Category filtering
  - Location-based search
- **Test Design Specification: TDS-SF-1002**

This specification will outline the test scenarios for search accuracy, response time, and filtering capabilities.

##### 3. Post Property System:

- **Features:**
  - Posting Hostel Property
  - Adding specific details
  - Editing and updating listings
- **Test Design Specification: TDS-RRS-1003**

This specification will cover test cases related to review submission, rating calculations, and user interactions with their reviews.

#### 5.2.2.5. Features not to be tested

##### 1. Online Payments Module:

- **Reason:** This module is planned for future and will not be part of this testing cycle.

##### 2. Newsletter System:

- **Reason:** Due to third-party integration changes, this system will be tested in a separate cycle.

#### 5.2.2.6. Approach

**Overall Testing Strategy:** Our approach to testing the Hostel Administration System platform will be systematic and iterative. We will employ a combination of manual and automated testing techniques to ensure comprehensive coverage. The testing will be divided into different phases, including unit testing, integration testing, system testing, and user acceptance testing.

#### **5.2.2.7. Item pass/fail criteria**

All core functionality of the systems should function as expected and outlined in the individual test cases. There must be no critical defects found and an end user must be able to complete a purchase cycle successfully and initiate a refund without any errors. 95% of all test cases should pass and no failed cases should be crucial to the end-user's ability to use the application.

#### **5.2.2.8. Suspension criteria and resumption requirements**

**Suspension criteria:** In this section we must specify when to stop testing. If any major functionalities are not functional or system experiences login issues then testing should suspend. Testing should be paused immediately if either system experiences login issues or failure in any basic CRUD (Create, Read, Update and Delete) actions.

**Resumption Requirement:** The test cases which are not dependent on the case where the bug is reported will be executed in parallel with the bug fixing. Once the failed test case has been taken note of and has been identified and fixed then the testing for the failed test case will resume.

### 5.2.2.9. Test deliverables

#### **Test Plan:**

A document that outlines the scope, approach, resources, and schedule of the testing activities. It defines the test items, features to be tested, and the personnel responsible for each task.

#### **Test Design Specifications:**

Detailed documents that describe the test conditions, the data to be used for testing, and the expected results (often including expected response times).

#### **Test Case Specifications:**

Documents that provide a detailed description of what specific input data will be used, the expected outcomes, and the specific criteria for test pass or fail.

#### **Test Procedure Specifications:**

Step-by-step instructions on how to execute the test cases, including any setup prerequisites and the order in which tests should be executed.

#### **Test Item Transmittal Reports:**

Reports that document the test items that have been transmitted (e.g., software builds or releases) to the testing team, including their version and revision levels.

#### **Test Logs:**

Detailed records of the test execution process, capturing which tests were run, when they were run, who ran them, and the results of each test.

#### **Test Incident Reports:**

Documents that capture any anomalies or issues found during testing. This includes details about the defect, steps to reproduce, severity, and any other relevant information.

#### **Test Summary Reports:**

A comprehensive report that summarizes the testing activities, results, statistics, and any outstanding defects. It provides an overview of the test coverage, pass/fail rates, and the overall quality of the test item.

#### **Test Input Data:**

The specific data sets or values used during testing to execute the test cases.

#### **Test Output Data:**

The results produced by the test item when it is tested using the test input data. This includes any generated files, logs, or other outputs.

#### **Test Tools:**

Any software or utilities used to facilitate the testing process. This can include module drivers (software that initiates a module's functions for testing) and stubs (simplified implementations that simulate the behavior of complex modules during testing).

By ensuring that all these deliverables are produced and maintained, PNC can ensure a thorough and well-documented testing process, which is crucial for understanding the quality of the software, addressing any issues, and meeting compliance and audit requirements.

### 5.2.2.10. Testing tasks

Specify the list of testing tasks we need to complete in the current project. Test environment should be ready prior to test execution phase. The following activities must be completed:

- Test plan prepared.
- Functional specifications written and delivered to the testing team

- Environment should be ready for testing (test data, test logins, test payment information, etc.).
- Perform the tests
- Prepare test summary report

#### **5.2.2.11. Environmental needs**

- Postman
- Live microservices
- Stable internet
- Live database

#### **5.2.2.12. Responsibilities**

Test plan should be prepared by test lead. Preparation and execution of tests should be carried out by testers. The test manager is responsible for facilitating the testing project, coordinating availability and schedule of testers and training them as needed. Each tester should understand the expectations on completion date and level of quality. The Test Manager should also communicate any risks to the team.

#### **5.2.2.13 Staffing and training needs**

Testing should be done by both testers. Testers should conduct testing on each system. It is preferred that there will be at least one of the testers assigned to the project for the system/integration and unit testing phases of the project. This will require assignment of a person in the full project from the beginning of the project to participate in reviews, development etc. other member will be assigned more into testing. If a separate test person is not available the test lead will assume this role.

#### **5.2.2.14. Schedule**

Complete details on when to start, finish and how much time each task should take place. Testing will take place 4 weeks prior to the launch date. The first round of testing should be completed in 1 week.

#### **5.2.2.15. Risks and contingencies**

In case of a wrong budget estimation, the cost may overrun. Contingency Plan establish the scope before beginning the testing tasks and pay attention in the project planning and also track the budget estimates constantly. If the first component testing is not completed within a day it can be delay. After bug fixes the testing will be performed. The various risks should be considered and the system should be tested properly.

#### **5.2.2.16 Approvals**

**Name:** Dr. Nauman Riaz Chaudhry

**Title:** “Hostel Administration System”

**Date:**

**Signature:**

## **5.3. Test design specification**

### **5.3.1. Purpose**

To prescribe the scope, approach, resources, and schedule of the testing activities. To identify the items being tested, the features to be tested, the testing tasks to be performed, the personnel responsible for each task, and the risks associated with this plan.

### **5.3.2. Outline**

A test plan shall have the following structure:

- a. Test plan identifier;
- b. Introduction;
- c. Test items;
- d. Features to be tested;
- e. Features not to be tested;
- f. Approach;
- g. Item pass/fail criteria;

The sections shall be ordered in the specified sequence. Additional sections may be included immediately prior to Approvals. If some or all of the content of a section is in another document, then a reference to that material may be listed in place of the corresponding content. The referenced material must be attached to the test plan or available to users of the plan.

Details on the content of each section are contained in the following sub-clauses.

#### **5.3.2.1 Test plan identifier**

Specify the unique identifier assigned to this test plan.

#### **5.3.2.2. Introduction**

Summarize the software items and software features to be tested. The need for each item and its history may be included. References to the following documents, when they exist, are required in the highest-level test plan:

- a. Project authorization
- b. Project plan
- c. Quality assurance plan
- d. Configuration management plan
- e. Relevant policies
- f. Relevant standards

In multilevel test plans, each lower-level plan must reference the next higher-level plan.

#### **5.3.2.3. Test items**

Identify the test items including their version/revision level. Also specify characteristics of their transmittal media that impact hardware requirements or indicate the need for

logical or physical transformations before testing can begin (e.g., programs must be transferred from tape to disk). Supply references to the following test item documentation, if it exists:

- a. Requirements specification
- b. Design specification
- c. Users guide
- d. Operations guide
- e. Installation guide

Reference any incident reports relating to the test items. Items that are to be specifically excluded from testing may be identified.

#### **5.3.2.4. Features to be tested**

Identify all software features and combinations of software features to be tested. Identify the test design specification associated with each feature and each combination of features.

#### **5.3.2.5. Features not to be tested**

Identify all features and significant combinations of features that will not be tested and the reasons.

#### **5.3.2.6. Approach**

Describe the overall approach to testing. For each major group of features or feature combinations, specify the approach that will ensure that these feature groups are adequately tested. Specify the major activities, techniques, and tools that are used to test the designated groups of features.

The approach should be described in sufficient detail to permit identification of the major testing tasks and estimation of the time required to do each one. Specify the minimum degree of comprehensiveness desired. Identify the techniques that will be used to judge the comprehensiveness of the testing effort (e.g., determining which statements have been executed at least once).

Specify any additional completion criteria (e.g., error frequency). The techniques to be used to trace requirements should be specified. Identify significant constraints on testing such as test item availability, testing resource availability, and deadlines.

#### **5.3.2.7. Item pass/fail criteria**

Specify the criteria to be used to determine whether each test item has passed or failed testing.

#### **5.3.2.8. Suspension criteria and resumption requirements**

Specify the criteria used to suspend all or a portion of the testing activity on the test items associated with this plan. Specify the testing activities that must be repeated, when testing is resumed.



#### **5.3.2.9. Test deliverables**

Identify the deliverable documents. The following documents should be included:

- a. Test plan
- b. Test design specifications
- c. Test case specifications
- d. Test procedure specifications
- e. Test item transmittal reports
- f. Test logs
- g. Test incident reports
- h. Test summary reports

Test input data and test output data should be identified as deliverables. Test tools (e.g., module drivers and stubs) may also be included.

#### **5.3.2.10. Testing tasks**

Identify the set of tasks necessary to prepare for and perform testing. Identify all inter task dependencies and any special skills required.

#### **5.3.2.11. Environmental needs**

Specify both the necessary and desired properties of the test environment. This specification should contain the physical characteristics of the facilities including the hardware, the communications and system software, the mode of usage (e.g., stand-alone), and any other software or supplies needed to support the test. Also specify the level of security that must be provided for the test facilities, system software, and proprietary components such as software, data, and hardware. Identify special test tools needed.

Identify any other testing needs (e.g., publications or office space). Identify the source for all needs that are not currently available to the test group.

#### **5.3.2.12. Responsibilities**

Identify the groups responsible for managing, designing, preparing, executing, witnessing, checking, and resolving. In addition, identify the groups responsible for providing the test items identified in 7.2.2.3 and the environmental needs identified in 5.3.2.11.

These groups may include the developers, testers, operations staff, user representatives, technical support staff, data administration staff, and quality support staff.

#### **5.3.2.13. Staffing and training needs**

Specify test-staffing needs by skill level. Identify training options for providing necessary skills.

#### **5.3.2.14. Schedule**

Include test milestones identified in the software project schedule as well as all item transmittal events. Define any additional test milestones needed. Estimate the time required to do each testing task. Specify the schedule for each testing task and test milestone. For each testing resource (i.e., facilities, tools, and staff), specify its periods of

use.

#### **5.3.2.15. Risks and contingencies**

Identify the high-risk assumptions of the test plan. Specify contingency plans for each (e.g., delayed delivery of test items might require increased night shift scheduling to meet the delivery date)

#### **5.3.2.16. Approvals**

Specify the names and titles of all persons who must approve this plan. Provide space for the signatures and dates.

### **5.4. Test Case Specification**

#### **5.4.1. Purpose**

To define a test case identified by a test design specification.

#### **5.4.2. Outline**

A test case specification shall have the following structure:

- a. Test case specification identifier
- b. Test items
- c. Input specifications
- d. Output specifications
- e. Environmental needs
- f. Special procedural requirements
- g. Inter case dependencies

The sections shall be ordered in the specified sequence. Additional sections may be included at the end. If some or all of the content of a section is in another document, then a reference to that material may be listed in place of the corresponding content. The referenced material must be attached to the test case specification or available to users of the case specification. Since a test case may be referenced by several test design specifications used by different groups over a long time period, enough specific information must be included in the test case specification to permit reuse.

Details on the content of each section are contained in the following sub-clauses.

##### **5.4.2.1. Test case specification identifier**

Specify the unique identifier assigned to this test case specification.

##### **5.4.2.2 Test items**

Identify and briefly describe the items and features to be exercised by this test case.

For each item, consider supplying references to the following test item documentation:

- a. Requirements specification
- b. Design specification
- c. Users guide

- d. Operations guide
- e. Installation guide

#### **5.4.2.3. Input specifications**

Specify each input required to execute the test case. Some of the inputs will be specified by value (with tolerances where appropriate), while others, such as constant tables or transaction files, will be specified by name. Identify all appropriate databases, files, terminal messages, memory resident areas, and values passed by the operating system. Specify all required relationships between inputs (e.g., timing).

#### **5.4.2.4. Output specifications**

Specify all of the outputs and features (e.g., response time) required of the test items. Provide the exact value (with tolerances where appropriate) for each required output or feature.

#### **5.4.2.5. Environmental needs**

##### **5.4.2.5.1. Hardware**

Specify the characteristics and configurations of the hardware required to execute this test case (e.g., 132 character ´ 24 line CRT).

##### **5.4.2.5.2. Software**

Specify the system and application software required to execute this test case. This may include system software such as operating systems, compilers, simulators, and test tools. In addition, the test item may interact with application software.

##### **5.4.2.5.3. Other**

Specify any other requirements such as unique facility needs or specially trained personnel.

#### **5.4.2.6. Special procedural requirements**

Describe any special constraints on the test procedures that execute this test case. These constraints may involve special set up, operator intervention, output determination procedures, and special wrap up.

#### **5.4.2.7. Inter case dependencies**

List the identifiers of test cases that must be executed prior to this test case. Summarize the nature of the dependencies.

### **5.5. Test procedure specification**

#### **5.5.1. Purpose**

To specify the steps for executing a set of test cases or, more generally, the steps used to

analyze a software item in order to evaluate a set of features.

### **5.5.2 Outline**

A test procedure specification shall have the following structure:

- a. Test procedure specification identifier
- b. Purpose
- c. Special requirements
- d. Procedure steps

The sections shall be ordered in the specified sequence. Additional sections, if required, may be included at the end. If some or all of the content of a section is in another document, then a reference to that material may be listed in place of the corresponding content. The referenced material must be attached to the test procedure specification or available to users of the procedure specification.

Details on the content of each section are contained in the following sub clauses.

#### **5.5.2.1. Test procedure specification identifier**

Specify the unique identifier assigned to this test procedure specification. Supply a reference to the associated test design specification.

#### **5.5.2.2. Purpose**

Describe the purpose of this procedure. If this procedure executes any test cases, provide a reference for each of them. In addition, provide references to relevant sections of the test item documentation (e.g., references to usage procedures).

#### **5.5.2.3. Special requirements**

Identify any special requirements that are necessary for the execution of this procedure. These may include prerequisite procedures, special skills requirements, and special environmental requirements.

#### **5.5.2.4. Procedure steps**

Include the steps in 8.5.2.4.1. through 8.5.2.4.10 as applicable.

##### **5.5.2.4.1. Log**

Describe any special methods or formats for logging the results of test execution, the incidents observed, and any other events pertinent to the test (see Clauses 9 and 10).

##### **5.5.2.4.2. Set up**

Describe the sequence of actions necessary to prepare for execution of the procedure.

##### **5.5.2.4.3. Start**

Describe the actions necessary to begin execution of the procedure.

##### **5.5.2.4.4. Proceed**

Describe any actions necessary during execution of the procedure.

#### **5.5.2.4.5. Measure**

Describe how the test measurements will be made (e.g., describe how remote terminal response time is to be measured using a network simulator).

#### **5.5.2.4.6. Shut down**

Describe the actions necessary to suspend testing, when unscheduled events dictate.

#### **5.5.2.4.7. Restart**

Identify any procedural restart points and describe the actions necessary to restart the procedure at each of these points.

#### **5.5.2.4.8. Stop**

Describe the actions necessary to bring execution to an orderly halt.

#### **5.5.2.4.9. Wrap up**

Describe the actions necessary to restore the environment.

#### **5.5.2.4.10. Contingencies**

Describe the actions necessary to deal with anomalous events that may occur during execution.

### **5.6. Test item transmittal report**

#### **5.6.1. Purpose**

To identify the test items being transmitted for testing. It includes the person responsible for each item, its physical location, and its status. Any variations from the current item requirements and designs are noted in this report.

#### **5.6.2. Outline**

A test item transmittal report shall have the following structure:

- a. Transmittal report identifier
- b. Transmitted items
- c. Location
- d. Status
- e. Approvals

The sections shall be ordered in the specified sequence. Additional sections may be included just prior to Approvals. If some or all of the content of a section is in another

document, then a reference to that material may be listed in place of the corresponding content. The referenced material must be attached to the test item transmittal report or available to users of the transmittal report.

Details on the content of each section are contained in the following sub clauses.

#### **5.6.2.1. Transmittal report identifier**

Specify the unique identifier assigned to this test item transmittal report.

#### **5.6.2.2. Transmitted items**

Identify the test items being transmitted, including their version/revision level. Supply references to the item documentation and the test plan relating to the transmitted items. Indicate the people responsible for the transmitted items.

#### **5.6.2.3. Location**

Identify the location of the transmitted items. Identify the media that contain the items being transmitted. When appropriate, indicate how specific media are labeled or identified.

#### **5.6.2.4. Status**

Describe the status of the test items being transmitted. Include deviations from the item documentation, from previous transmittals of these items, and from the test plan. List the incident reports that are expected to be resolved by the transmitted items. Indicate if there are pending modifications to item documentation that may affect the items listed in this transmittal report.

#### **5.6.2.5. Approvals**

Specify the names and titles of all persons who must approve this transmittal. Provide space for the signatures and dates.

### **5.7. Test log**

#### **5.7.1. Purpose**

To provide a chronological record of relevant details about the execution of tests.

#### **5.7.2. Outline**

A test log shall have the following structure:

- a. Test log identifier;
- b. Description;
- c. Activity and event entries.

The sections shall be ordered in the specified sequence. Additional sections may be included at the end. If some or all of the content of a section is in another document, then a reference to that material may be listed in place of the corresponding content. The referenced material must be attached to the test log or available to users of the log. Details on the content of each section are contained in the following sub clauses.

#### **5.7.2.1. Test log identifier**

Specify the unique identifier assigned to this test log.

#### **5.7.2.2. Description**

Information that applies to all entries in the log except as specifically noted in a log entry should be included here. The following information should be considered:

- Identify the items being tested including their version/revision levels. For each of these items, supply a reference to its transmittal report, if it exists.
- Identify the attributes of the environments in which the testing is conducted. Include facility identification, hardware being used (e.g., amount of memory being used, CPU model number, and number and model of tape drives, and/or mass storage devices), system software used, and resources available (e.g., the amount of memory available).

#### **5.7.2.3. Activity and event entries**

For each event, including the beginning and end of activities, record the occurrence date and time along with the identity of the author. The information in 9.2.3.1 through 9.2.3.5 should be considered:

##### **5.7.2.3.1. Execution description**

Record the identifier of the test procedure being executed and supply a reference to its specification. Record all personnel present during the execution including testers, operators, and observers. Also indicate the function of each individual.

##### **5.7.2.3.2. Procedure results**

For each execution, record the visually observable results (e.g., error messages generated, aborts, and requests for operator action). Also record the location of any output (e.g., reel number). Record the successful or unsuccessful execution of the test.

##### **5.7.2.3.3. Environmental information**

Record any environmental conditions specific to this entry (e.g., hardware substitutions).

##### **5.7.2.3.4. Anomalous events**

Record what happened before and after an unexpected event occurred (e.g., A summary display was requested and the correct screen displayed, but response seemed unusually long. A repetition produced the same prolonged response). Record circumstances surrounding the inability to begin execution of a test procedure or failure to complete a test procedure (e.g., a power failure or system software problem).

##### **5.7.2.3.5. Incident report identifiers**

Record the identifier of each test incident report, whenever one is generated.

### **5.8. Test incident report**

### **5.8.1. Purpose**

To document any event that occurs during the testing process that requires investigation.

### **5.8.2. Outline**

A test incident report shall have the following structure:

- a. Test incident report identifier
- b. Summary
- c. Incident description
- d. Impact

The sections shall be ordered in the specified sequence. Additional sections may be included at the end. If some or all of the content of a section is in another document, then a reference to that material may be listed in place of the corresponding content. The referenced material must be attached to the test incident report or available to users of the incident report.

Details on the content of each section are contained in the following sub clauses.

#### **5.8.2.1. Test incident report identifier**

Specify the unique identifier assigned to this test incident report.

#### **5.8.2.2. Summary**

Summarize the incident. Identify the test items involved indicating their version/revision level. References to the appropriate test procedure specification, test case specification, and test log should be supplied.

#### **5.8.2.3. Incident description**

Provide a description of the incident. This description should include the following items:

- a. Inputs
- b. Expected results
- c. Actual results
- d. Anomalies
- e. Date and time;
- f. Procedure step;
- g. Environment;
- h. Attempts to repeat;
- i. Testers;
- j. Observers.

Related activities and observations that may help to isolate and correct the cause of the incident should be included (e.g., describe any test case executions that might have a bearing on this particular incident and any variations from the published test procedure).

#### **5.8.2.4. Impact**

If known, indicate what impact this incident will have on test plans, test design



specifications, test procedure specifications, or test case specifications.

## **5.9. Test summary report**

### **5.9.1. Purpose**

To summarize the results of the designated testing activities and to provide evaluations based on these results.

### **5.9.2. Outline**

A test summary report shall have the following structure:

- a. Test summary report identifier
- b. Summary
- c. Variances
- d. Comprehensive assessment
- e. Summary of results
- f. Evaluation
- g. Summary of activities
- h. Approvals

The sections shall be ordered in the specified sequence. Additional sections may be included just prior to Approvals. If some or all of the content of a section is in another document, then a reference to that material may be listed in place of the corresponding content. The referenced material must be attached to the test summary report or available to users of the summary report.

Details on the content of each section are contained in the following sub clauses.

#### **5.9.2.1. Test summary report identifier**

Specify the unique identifier assigned to this test summary report.

#### **5.9.2.2. Summary**

Summarize the evaluation of the test items. Identify the items tested, indicating their version/revision level. Indicate the environment in which the testing activities took place. For each test item, supply references to the following documents if they exist: test plan, test design specifications, test procedure specifications, test item transmittal reports, test logs, and test incident reports.

#### **5.9.2.3. Variances**

Report any variances of the test items from their design specifications. Indicate any variances from the test plan, test designs, or test procedures. Specify the reason for each variance.

#### **5.9.2.4. Comprehensiveness assessment**

Evaluate the comprehensiveness of the testing process against the comprehensiveness criteria specified in the test plan if the plan exists. Identify features or feature combinations that were not sufficiently tested and explain the reasons.

**5.9.2.5. Summary of results**

Summarize the results of testing. Identify all resolved incidents and summarize their resolutions. Identify all unresolved incidents.

**5.9.2.6. Evaluation**

Provide an overall evaluation of each test item including its limitations. This evaluation shall be based upon the test results and the item level pass/fail criteria. An estimate of failure risk may be included.

**5.9.2.7. Summary of activities**

Summarize the major testing activities and events. Summarize resource consumption data, e.g., total staffing level, total machine time, and total elapsed time used for each of the major testing activities.

**5.9.2.8. Approvals**

Specify the names and titles of all persons who must approve this report. Provide space for the signatures and dates.

## **Chapter 6: User Manual**

---

## Signup Form:

**CREATE A NEW ACCOUNT** ✕

---


☐ I agree to the [Terms of Service](#) and [Privacy Policy](#)

[CREATE ACCOUNT](#)


---

Already registered? [Sign in to your account](#)

## Home Screen:



[HOME](#) [ALL HOSTELS](#) [SOFTWARE](#) [CONTACT US](#)

 [MAAZ HASSAN](#) [POST PROPERTY](#)


SEARCH HOSTELS IN A CERTAIN AREA.

**FIND HOSTELS**

Search The Location And Select The Category To Find Hostels Accordingly.

[FIND HOSTELS](#)

## Post Property (General details):

HOME ALL HOSTELS SOFTWARE CONTACT USMAAZ HASSAN POST PROPERTY

1GENERAL DETAILS

2HOSTEL SPECIFICATIONS

3HOSTEL FACILITIES

### HOSTEL DETAILS

Hostel's Name

Hostel's Title Slug for UrlURL Slug

Hostel Phone Number

Select Category

Select Country

Select State

Select City

Hostel's Complete Address

Enter Some Details About Your Hostel

Gujrat, Pakistan

### HOSTEL PICTURES

(SELECT UPTO 6 PICTURES)

## Hostel Specifications:

1GENERAL DETAILS

2HOSTEL SPECIFICATIONS

3HOSTEL FACILITIES

### RENT AND AREA

10000

Per Month

Yes

Monthly

10 Marla

Furnished

Full Building

All Days

Attacheded

Included

Yes

3 Beds

### ROOM PRICE PLANS

Double Room

Yes

10000

Special Facility (optional)

ADD

Previous

Next

## Hostel Facilities:

1

GENERAL DETAILS

2

HOSTEL SPECIFICATIONS

3

HOSTEL FACILITIES

FACILITIES

☒ Electricity Backup

☐ Geyser

☐ Cable / Satellite

☐ Washer

☐ Laundry

☒ security Cameras

☒ Attach Bath

☐ Telephone

☒ Fridge

☐ Heating

☒ Wifi

☐ Cupboard

☐ Security Gurads

☐ Oven

☐ Pool

☐ Breakfast

☒ Parking

☐ Kitchen

☐ Air Cooler

☐ Gym

☒ Lunch

☐ TV

☐ Doorman

☒ Roof Top

☒ Non Smoking

☒ Dinner

☐ Internet

☐ Safety Fire

☐ Outdoor Sitting

☐ Pets Allowed

TAGS FOR SEO

Best Hostel

Add tag

Previous

Submit

## Profile:

Maaz Hassan

Gujrat, Pakistan. Joined in Dec 2022

Email

rjmaaz23@gmail.com

Contact

03484219440

Gender

Male

CHANGE PROFILE PIC

About Me

Location

Gujrat, Pakistan

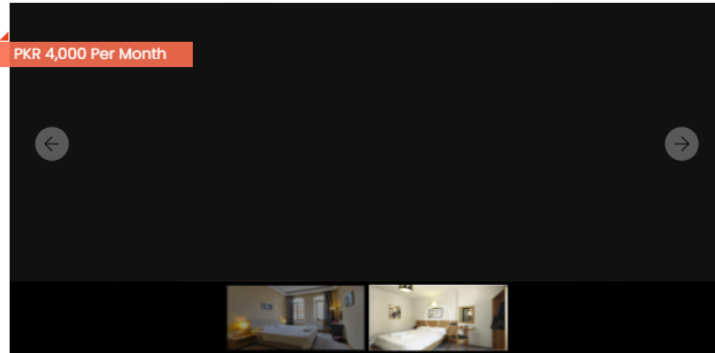
APPLICATIONS

ACCOUNT SETTINGS

You have Not Applied In Any Hostel Yet.

## Listings:

### AL HAFEEZ BOYS HOSTEL IN LAHORE



Please quote our Reference [Cityhostels.com.pk](http://Cityhostels.com.pk) when calling.

Contact No. : \*\*\*\*\* [Show Number](#)

Address : Lahore, Punjab, Pakistan

Office Timing: From 9:00 AM To 9:00 PM

★ Rating  
3.0 (2 Ratings)

👁 Views  
3760

♂ Category  
Boys Hostel

f Messenger  
[Message Us](#)

#### Amenities

- |          |                  |                      |                     |
|----------|------------------|----------------------|---------------------|
| 📶 Wifi   | ✓ Carpeted Rooms | ✓ water cooler       | 🚗 Parking           |
| ✓ Geyser | 🌐 Internet       | ⚙ Fans               | 📡 Cable / Satellite |
| 📺 TV     | 🧺 Washer         | ✓ Electricity Backup | ☎ Telephone         |

[Book Now](#)

You Must **Sign In** To Book Online

#### Features

- |                  |                                    |
|------------------|------------------------------------|
| Condition:       | <a href="#">Furnished</a>          |
| Floor:           | <a href="#">Ground Floor</a>       |
| Bills:           | <a href="#">Bills Not Included</a> |
| Letting Type:    | <a href="#">Long Term</a>          |
| Rent Negotiable: | <a href="#">PKR : 4000</a>         |
| Rent Period:     | <a href="#">Per Month</a>          |
| Schedule:        | <a href="#">All Days</a>           |
| Bathroom:        | <a href="#">Attached</a>           |
| Mess:            | <a href="#">Mess Not-included</a>  |
| Lawn:            | <a href="#">No</a>                 |
| Occupancy:       | <a href="#">3 Beds</a>             |
| Hostel Size:     | <a href="#">10 Maria</a>           |

#### Google Map Location

