Software Requirement Specification

For

Hostel Management System

Prepared by Shehraz Sarwar

Table of Contents

1. Introduction

1.1 Purpose

The purpose of the software is to maintain and manage the hostel system and accommodation facility like store records, online reservation and booking of the rooms. In short, the main purpose of this hostel management system is to optimize the overall management of the hostel.

1.2 Scope

The scope of the hostel management system is to provide user an efficient online booking and reservation system, enhancing their overall experience by providing several different services online like room allocations, online payment, reservation, and displaying available rooms and their prices by category.

1.3 Document Conventions

The document includes different sections which tells us what our software should do and what are their requirements like purpose, scope, functional and non-functional requirements with proper top-down numbering. The document is written in "Times new roman" font with size 18 for main headings, 14 for sub-headings and for its description 18 size of font are used with 1.0 line spacing. The headers are also included displaying a title that the SRS is written for with each page number.

1.4 Intended Audience

The Hostel Management System is intended for hostel management team and administrators who are running hostels by using this system they can increase their use experience and easily manage their hostel, guests that are looking to stay for a long time and especially for the students that are come from different cities to pursue their higher education.

2. System Overview

2.1 Description

The hostel management system gives users an online service or system through which they can make online reservations and can be able to book rooms without reaching out physically. They can be able to book their favorite rooms and floor they like while staying at their comfort zone. Following are the key points that our system provides the users and to hostel management team and their administrators.

- Online room bookings
- Manage bookings
- Online payment
- Room allocations

2.2 Features

Here are some features of the Hostel Management System:

- Online Booking: Guests can easily reserve rooms online by selecting dates, room types.
- Management: Staff can keep track of all bookings, make changes, and handle guest details.
- Room Selection: Guests have the option to choose and book available rooms that meet their needs.
- Room Assignment: Rooms are assigned to guests based on availability and their preferences.
- Online Payment: Secure online payment options are available for guests.
- Show Reports: Generate reports on total earnings.

These features can make hostel management easier and provide a better experience for both guests and staff.

2.3 Assumptions

Here are some of the assumptions that may cause our system functionality:

- Internet Issue: The system is almost totally based on the internet, so it is necessary to have a stable internet connection although booking can be done offline if the user is physically present at the hostel but most of the time the bookings and reservations will be done online so the internet will become the major issue if it is not stable, it may also decrease customer satisfaction and their experience.
- System Compatibility: Make sure that the system on which the software is running also called server is in good working condition and have a good capacity to store maximum amount of data, lack of storage may cause systems functionality.
- Security: Hackers or virus may attack the system, so for this case there must be a backup option available for the software in order to recover from hackers or virus attacks.
- Professional Team: The staff or team handling the system should have the proper knowledge of the system and how does it work.
- Accurate Information: Make sure that the user provides the correct/legal information.

3. Functional Requirements

3.1 Use Cases:

Following are the Use Cases in the hostel management system:

3.1.0. User/Guest select room they want and their preference.
3.1.1. System checks availability of rooms.
3.1.2. User/Guest provide personal details.
3.1.3. User/Guest provide payment details to make payment.

3.1.4. System will verify the payment details.
3.1.5. System will assign room and save details of the user.
3.1.6. System will generate confirmation notification for the user.
3.1.7. Manager add new rooms in the system.
3.1.8. Manager remove rooms allocation or add available room if someone leaves the hostel.
3.1.9. Receptionist confirms guest's departure.
3.2.0. System updates room status and guest record.

3.2 Actors

Following are the actors for the above-mentioned use cases:

- User Or Guest
- System
- Manager
- Receptionist

3.3 Pre-Conditions

Follow are the pre-conditions for the above-mentioned use cases:
3.3.1. User/Guest must login into the system.
3.3.2. User/Guest must provide personal details.
3.3.3. User/Guest select room preference.
3.3.4. User/Guest select time and date.
3.3.5. User/Guest could make booking.
3.3.6. User/Guest add their payment details to make payment.
3.3.7. Receptionist must confirm the departure and reservation.
3.3.8. The manager must have the access to the system.
3.3.9. Manager can add new rooms and delete allocated rooms.

3.4 Flows

Follow are the flows for the above-mentioned use cases:
3.4.1. User/Guest must have access to the system to make booking.
3.4.2. System checks availability of the rooms.
3.4.3. User/Guest must provide personal details.
3.4.4. User/Guest select room preference with time and date.
3.4.5. User/Guest add their payment details.
3.4.6. Receptionist confirms the details are valid if not then deletes the record of that guest/user.
3.4.7. System generates the confirmation message if details are valid.
3.4.8. Receptionist confirms the guest departure.
3.4.9. System assigns rooms to guest/user and updates the records of guests and rooms

3.5 Post-Conditions

Follow are the pre-conditions for the above-mentioned use cases:
3.5.1. Verification messages should be generated by the system.
3.5.2. User/Guest details must be updated by system.
3.5.3. Booking record should be saved in the system.
3.5.4. The room availability status must be updated.
3.5.5. The guest should assign a room.

3.5.6. The room details must be updated.

4. Non-Functional Requirements

4.1 Performance

Here are some things it needs to do:

4.1.1. The system should respond quickly when users interact with it, like booking a room.
4.1.2. It should be able to handle at least 100 users at the same time without slowing down.
4.1.3. As more people start using the system, it should handle them without any major problem.
4.1.4. Users should be able to use the system almost all the time.
4.1.5. If something goes wrong, like the system crashes, it should be back up and running in 5 minutes or less, and it shouldn't lose any important information.
4.1.6. The system should use computer resources like memory and processing power wisely, without using too much.
4.1.7. It should be able to keep information like reservations and guest details for a long time, at least 5 years, without slowing down.
4.1.8. If something goes wrong, like trying to book a room that's already taken, the system should let users know.
4.1.9. Users' information should be safe in the system, and it should let users in quickly when they log in.

4.2 Security

Following are security points that the system must have:

4.2.1. The system must store password securely.
4.2.2. The system should do backups every week so that every detail should not be deleted in case of virus or hacker's attack.
4.2.3. The system should implement not a robot feature by asking them several easy question in order to avoid scam.
4.2.4. Have disaster recovery plans.
4.2.5. The system should force user or suggest user to write strong password and apply two factor authentication for strong account security.

4.3 Reliability

This is about making sure the system works all the time. Even if something goes wrong, like a power outage or a computer glitch, the system should be able to fix itself quickly without losing any of our important data. It's like having a superhero ready to save the day!

4.3.1. The system should be available almost all the time.
4.3.2. If something goes wrong, like a computer crashing or a problem with the internet, the system should fix itself automatically without losing any information.
4.3.3. All the information stored in the system should stay safe and not get messed up or lost.
4.3.4. The system should make copies of all the important information regularly
4.3.5. The most important parts of the system, like the main computer and the database where all the information is stored, should have backups in case they stop working.

4.3.6. As more people start using the system, it should be able to handle all of them without slowing down or crashing.

4.4 Usability

Following are the Usability points our system should have:

4.4.1. The system should be easy to use with simple menus and buttons.
4.4.2. It should work for everyone, including people with disabilities, following the rules for making websites easy to use.
4.4.3. Tips and guides should be available to help users understand how the system works.
4.4.4. Doing common tasks should be quick and easy.
4.4.5. Users should be able to change things like language and colors to suit their preferences.
4.4.6. The system should help users avoid making mistakes and let them know if something goes wrong.
4.4.7. Users should get feedback quickly when they do something, like seeing a message when they complete a task.

4.5 Compatibility

The system should work on different types of devices like mobile and as well as on computers and laptops. It should also be work on different types of operating systems like on Linux, android, Iphone and windows. It should also be able to use on different web browsers like Chrome, Firefox and Edge.

4.6 Documentation

This means it should be easy for people to fix and update the system if something goes wrong or if we want to add new features. The instructions on how to use and change the system should be clear in the documentation of the system. If the developer who is new to the team working to update the software should be able to understand the requirements documentation and it also be helpful for the client to understand the software.