Software Requirements Specification

for

Hostel Management System

Version <1.0>

Prepared by

Group Number: 28

Siruvalam Karthik	**B190531CS**	**karthik_b190531cs@nitc.ac.in**
Ambati Sathwik	**B190500CS**	**ambati_b190500cs@nitc.ac.in**
Gaddala Abhinav	**B190366CS**	**abhinav_b190366cs@nitc.ac.in**
Nelluru Keerthi Bhavan	**B191096CS**	**keerthi_b191096cs@nitc.ac.in**
Putta Suman Rao	**B190900CS**	**sumanrao_b190900cs@nitc.ac.in**

Instructor: Dr. Abdul Nazeer K A

Course: CS3002D Database Management Systems

Date: 23-10-2021

HOSTEL MANAGEMENT SYSTEM

Introduction

1.1 Document Purpose

The Software Requirement Specification (SRS) provides a detailed description of requirements and purpose of Hostel Management System(HMS). This includes a detailed description of the project we are going to develop. The clear understanding of the system and its functionality will allow the correct software to be developed for the end user and will be used for development of the project in the future. The SRS helps the project team to understand the expectations of the HMS to construct the appropriate software. The end users can use the SRS to test if the constructing team will be constructing the software to their requirements. The HMS can be designed,constructed and tested using this SRS.

1.2 Product Scope

- HMS is designed for hostels (like schools , universities)
- There are some preset rules for reservation of the hostels.
- HMS checks the submitted application forms obtained from the students and verifies it with the student database.
- If he/she is eligible , then the hostel room is allotted.
- HMS also records the complaints received from the hostel inmates.

1.3 Intended Audience and Document Overview

The intended audience of our project is students. HMS is a web application which aims for computerisation of procedure for allocation of hostel rooms. This system involves receiving applications from the students and allocating rooms , receiving complaints from respective room inmates and recording them for further use. The present format for allocating hostel rooms and resolving the complaints is less efficient.

1.4 Definitions, Acronyms and Abbreviations

SRS : *Software Requirements Specification.*
SyRS : *System Requirements Specification.*
HMS : *Hostel management system.*

1.5 Document Conventions

In this SRS document all the section titles and main title uses Arial font style , Bold and font size of 14. All paragraphs are in Arial font style and font size of 12. We used 1”

margin throughout the document. In section 1.4a, we used Arial font bold style and font size of 12 for abbreviations. We used italics of the same font style for descriptions .

1.6 References and Acknowledgments

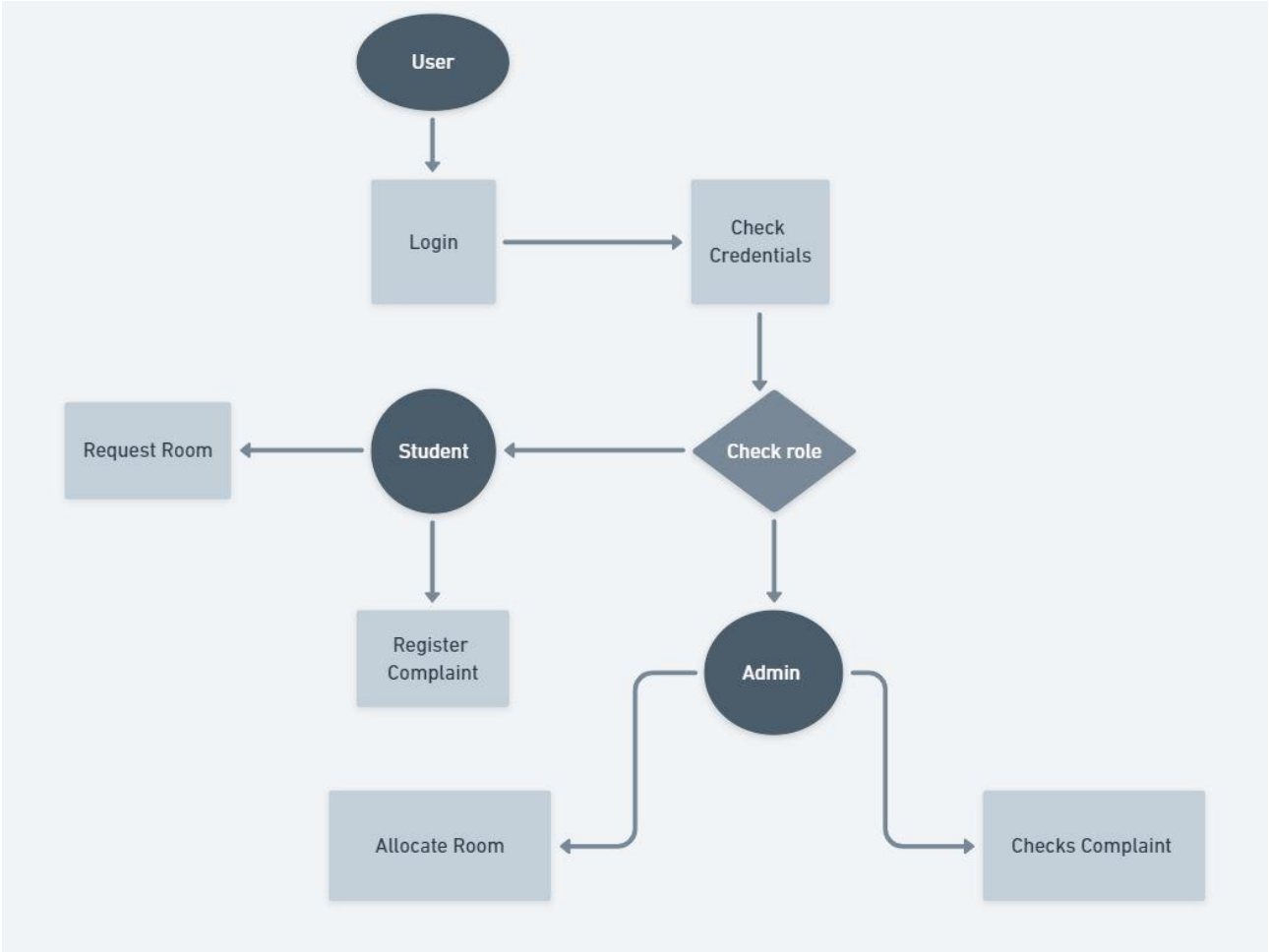
This document follows the interface style provided by the following link. We followed IEEE format in this document.

http://www.cse.msu.edu/~cse870/IEEEXplore-SRS-template.pdf .

2 Overall Description

2.1 Product Overview

This is a self-contained product used for managing a hostel. This software will have the potential for allocating rooms for the students and can receive complaints from the hostel inmates. This software is a subsystem of Institute management software. The details of students will be collected from the Institute's student database.



2.2 Product Functionality

Functionality of User:

- Users must create an account before accessing the software.
- Users must fill the respective forms to register a complaint, requesting for room.

- Users must login using their NITC mail.

Functionality of Software:

- Provides a different interface for admin and users.
- The admin's interface shows all the complaints and room requests from students.
- The student's interface provides respective forms for registering complaints and requesting room.

2.3 Design and Implementation Constraints

.

- The website runs 24 X 7.
- Implementation of databases using centralized database systems.
- This project will also contain a user interface built using javascript, html,css.
- Users must have a minimum of 512mb RAM and the latest version of the web browser.
- We use gunicorn with multiple workers for parallel operation.
- User’s web browser must support javascript for rendering the software.
- The challenges in developing the system are the required number of users.The expected number of users once launched will be around 5000 at the first point of implementation.
- Safety and Security : Users are given access to the website so that they can register into the website using NITC mail id.
- We use a hash function to store passwords securely in the database

2.4 Assumptions and Dependencies

- User must be a NITC student.
- Users must be allocated a room only once.
- Users must use the Linux operating system to interact with the system.
- Server must be running in the Linux Operating system.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

Welcome Interface:

This is the first interface that user encounters when they opens the application. This interface contains the login option. There are multiple login options like admin and student. Users can choose the login option from these options which redirect to the login page.

Login Page:

This page contains two fields for username and password. If you are admin then, after successful login you will be redirected to admin interface and if you are student then, after successful login you will be redirected to your respective user interface.

Admin Interface:

This page contains the complaint list registered by students along with room requesting forms. Admin can either reject this request or accept this request. If the admin accepts the request our software will automatically provide a room number to the student.

User Interface:

This page contains multiple options for the student to choose, if he didn't request for room then requisition room option will be available in this interface or if he is already allocated the room then request room option will be disabled. There will be another option for registering any difficulties he faces in the hostel or in his room.

3.1.2 Hardware Interfaces

Server side:

The web application will be hosted on a web server which is listening on the web standard port.

Client side:

- Processor: Pentium or greater

- RAM: 512MB
- Keyboard
- Monitor

3.1.3 Software Interfaces

Server side:

A javascript, html based interface accepts requests from client and forward information accordingly. The database will be hosted on postgresql.

Client side:

A web-browser which allows javascript and HTML.

3.2 Functional Requirements

< Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. This section is the direct continuation of section 2.2 where you have specified the general functional requirements. Here, you should list in detail the different product functions.

3.2.1 Student Functionalities :

- **Login:** Students can log in using the credentials(NITC email and password) provided
- **Register:** Registering as a member in the hostel
- **Request for a Room:** Request to allocate a room
- **File a Complaint:** Can make complaints regarding the hostel or his room.

3.2.2 Admin Functionalities :

- **Login:** Admin can log in using his/her NITC mail id and password.
- **Check Complaints:** Admin can check the complaints received in his/her hostel and can work to resolve them.
- **Get Student List:** Admin can get all the residents of the hostel.
- **Vacancy List:** Admin can get all the vacant beds in the hostel.
- **Add Worker:** Admin can append a worker to the worker's list.

3.3 Use Case Model

3.3.1 Use Case #1 (Student Login)

Author: Nelluru Keerthi Bhavan

Purpose - Check whether the provided credentials are correct.

Priority - High.

Preconditions - Student must have an existing account

Postconditions - The user must be a student and password provided by the user must match with the respective password for the given username in the database

Actors – System

Flow of Events

- Collects information provided by the student from frontend .
- Compare the credentials with the existing details in the database and if it matches redirects to corresponding student page interface.
- If the entered details do not match with the existing details in the database then the page reloads and asks the student to login again.

3.3.2 Use Case #2 (Admin Login)

Author: Ambati Sathwik

Purpose - Check whether the provided credentials are correct.

Priority - High.

Preconditions - Admin must have an existing account

Postconditions - The user must have an admin role and password provided by the user must match with the respective password for the given username in the database.

Actors – System

Flow of Events

- Collects information provided by the admin from frontend .
- Compare the credentials with the existing details in the database and if it matches redirects to corresponding student page interface.
- If the entered details do not match with the existing details in the database then the page reloads and asks the admin to login again.

3.3.3 Use Case #3 (Register Student) :

Author : Putta Suman Rao

Purpose - To register students in the hostel database.

Priority - High.

Preconditions - Must be a student of NITC, Must not have already registered.

Postconditions - Nil

Actors – System

Flow of Events

- Collects information provided by the student from frontend .
- Compare the credentials with the student database in Institute management system
- if it does not match it raises an error while registering

3.3.4 Use Case #4 (Request for Room)

Author: Siruvalam Karthik

Purpose - Student requests the system to allocate a room.

Priority - High.

Preconditions - Students must have an existing account and must be logged in.

Postconditions - The System will assign a room/ vacant bed according to availability.

Actors – System, Student

Flow of Events

- Student logs in using NITC mail id and password
- The student requests the system to assign a room/vacant bed
- The system will assign a room/vacant bed to the student.

3.3.5 Use Case #5 (File a Complaint) :

Author : Gaddala Abhinav

Purpose - To file complaints regarding any hostel issues of the student.

Priority - High.

Preconditions - Must have an account.

Postconditions - Nil

Actors – System

Flow of Events

- Students log in using NITC mail id and respective password.
- Registers the complaint.

3.3.6 Use Case #6 (Check Complaint) :

Author : Gaddala Abhinav

Purpose - To check and resolve the complaints received

Priority - High.

Preconditions - Must be an admin

Postconditions - Nil

Actors – Admin

Flow of Events

- Admin logs in to the system
- Check the complaints.

3.3.7 Use Case #7 (Get Student list) :

Author : Ambati Sathwik

Purpose - To get the list of the students in the hostel by the admin.

Priority - High.

Preconditions - Must have an account with an admin role.

Postconditions - Nil

Actors – System

Flow of Events

- Admin logs in using the existing account.
- Can get the list of the students in respective rooms of the hostel.

4 Other Non-functional Requirements

4.1 Performance Requirements

The performance of hardware components such as CPUs, hard disks, disk controllers, RAM, and network interfaces will significantly affect how fast our database performs. The workload equals the total demand from the DBMS, and it varies over time. The total workload is a combination of user queries, applications, transactions, and system commands directed through the DBMS at any given time. It can increase during admission time because all students try to request room. Workload strongly influences database performance. DBMS throughput is closely related to the processing capacity of the underlying systems.

Hence it is planned for:

- Processing of the largest possible workload.
- Minimum latency is possible.
- Keep all the login credentials safe and only authenticated users can enter into sensitive logins on UI which are mentioned earlier.

4.2 Safety and Security Requirements

There is always this possibility of Data breaching and Malware attacks. The information from the database may be accessed by outsiders or unauthorized users. Which could result in loss of data or originality of data, which creates chaos or misinterpretation of data when we unknowingly access our data. To not allow this to happen different techniques should be implemented at various levels of accessing data before performing the actual query on the database.

The types of control measures to deal with these problems: access control, inference control, flow control, and encryption. We also pass a password through a hash function before entering it in the database.

In addition to making every effort to prevent attacks and detecting attacks when an incident occurs, the DBMS should also be able to perform the following actions:

- **Restrictions**: Take immediate action to eliminate attackers' access to the system, and isolate or contain the problem to prevent further spread.
- **Damage assessmen**t: Determine the extent of the problem, including functional failure and data corruption.
- **Reconfiguration**: Reconfiguration to allow the operation to continue in degraded mode while recovery is in progress.

- **Repair**: restore damaged or lost data, and repair or reinstall failed system functions to restore normal operating levels.
- **Troubleshooting**: Find out the weaknesses used in the attack as much as possible and take measures to prevent a recurrence.

4.3 Software Quality Attributes

4.3.1 Reliability

As there will be only two types of users(Student, Admin). There will be no priority for any specific student. Hence, our software will be consistent while providing service to any student.

4.3.2 Adaptability

The software is implemented in such a way that most of institutions can adapt to this easily.

4.3.3 Availability

The software is available for anyone for free of cost as long as they are part of the institute.

4.3.4 Efficiency

The software uses gunicorn with multiple workers, so it will use its resources fully to complete the task.

4.3.5 Maintainability

This software just requires a database to function other than that it requires almost no cost and is easy to maintain.

4.3.6 Usability

This software contains a user-friendly interface. A user with a basic knowledge of English can handle this website with ease.

Appendix B - Group Log

Before conducting any meetings, each team member did the individual task of gathering some prerequisites about the topic. We analyzed the need of such an application and how it can be useful to NITCians. Everyone had their inputs on the features that must be added to the app in order to meet the requirements of the users.

Meeting Details :

1) ***Oct 22 : 6:00pm - 8:30pm***

In this meeting, we discussed the requirements and specifications of the software that we are going to build. We discussed the roles and responsibilities of each user. We finally decided to have an admin role to manage this software. We also concluded the functionalities of both admin and user. Each team member gave valuable input on various functionalities. we also discussed the inclusion of certain use cases. We concluded the meeting by splitting the workload equally among us.

2) ***Oct 23 : 2:30pm - 5:00pm***

First, we divided work among team members mainly the introduction part of the document was covered by Sathwik (B190500CS) and KeerthiBhavan (B1991096CS), the overall description was handled by Karthik(B190531CS) ,Suman (B190900CS), and Abhinav(1903656CS). Each team member's job was to fill the assigned section to them. A part of interfaces and functional requirements was also completed. Each team member came up with each product's functionalities.

3) ***Oct 24 : 2:00pm - 4:00pm***

***I**n this meeting, we decided to complete use cases. we split use cases among us as follows.*

- ➔ *Use Case1 : (Student Login) - KeerthiBhavan (B1901096CS)*
- ➔ *Use Case2 : (Admin Login) - Sathwik (B190500CS)*
- ➔ *Use Case 3 : (Register Student) - Suman (B190900CS)*
- ➔ *Use Case 4 : (Request Room) - Karthik (B190531CS)*
- ➔ *Use Case 5 : (File a complaint) - Abhinav (B190366CS)*
- ➔ *Use Case 6 : (Get Student List) - Sathwik (B190500CS)*
- ➔ *Use Case 7 : (Check Complaint) - Abhinav (B190366CS)*

Other non functional requirements were completed by all the team members together.