Contents

[**1. Introduction** 2](#_Toc152346496)

[**1.1 Purpose** 2](#_Toc152346497)

[**1.2 Document conventions** 2](#_Toc152346498)

[**1.3 Project scope** 2](#_Toc152346499)

[**1.4 References** 2](#_Toc152346500)

[**2. Overall description** 2](#_Toc152346501)

[**2.1 Product perspective** 2](#_Toc152346502)

[**2.2 User classes and characteristics** 3](#_Toc152346503)

[**2.3 Operating environment** 4](#_Toc152346504)

[**2.4 Design and implementation constraints** 5](#_Toc152346505)

[**2.5 Assumptions and dependencies** 6](#_Toc152346506)

[**3. System features** 7](#_Toc152346507)

[**3.x System feature X** 7](#_Toc152346508)

[**3.x.1 Description** 7](#_Toc152346509)

[**3.x.2 Functional requirements** 7](#_Toc152346510)

[**4. Data requirements** 7](#_Toc152346511)

[**4.1 Logical data model** 7](#_Toc152346512)

[**4.2 Data dictionary** 7](#_Toc152346513)

[**4.3 Reports** 7](#_Toc152346514)

[**4.4 Data acquisition, integrity, retention, and disposal** 7](#_Toc152346515)

[**5. External interface requirements** 8](#_Toc152346516)

[**5.1 User interfaces** 8](#_Toc152346517)

[**5.2 Software interfaces** 8](#_Toc152346518)

[**5.3 Hardware interfaces** 8](#_Toc152346519)

[**5.4 Communications interfaces** 8](#_Toc152346520)

[**6. Quality attributes** 8](#_Toc152346521)

[**6.1 Availability:** 8](#_Toc152346522)

[**6.2 Correctness:** 8](#_Toc152346523)

[**6.3 Flexibility:** 8](#_Toc152346524)

[**6.4 Portability:** 8](#_Toc152346525)

[**6.5 Reliability:** 8](#_Toc152346526)

[**6.6 Reusability:** 8](#_Toc152346527)

[**6.7 Robustness:** 8](#_Toc152346528)

[**6.8 Testability:** 9](#_Toc152346529)

[**7. Internationalization and localization requirements** 10](#_Toc152346530)

[**8. Other requirements** 10](#_Toc152346531)

[**Appendix A: Glossary** 10](#_Toc152346532)

[**Appendix B: Analysis models** 10](#_Toc152346533)

[**Activity diagram** 11](#_Toc152346534)

# **1. Introduction**

## **1.1 Purpose**

This Software Requirements Specification (SRS) document pertains to the Hostel Management System (HMS), version 1.0. The HMS is a comprehensive software solution designed to streamline and enhance the management of hostel operations within a university environment. The scope of this SRS encompasses the entire HMS, including its modules for incident management, room allocation, feedback, staff management, mess operations, fee submission, and user authentication. The document outlines the functional and non-functional requirements necessary for the development and successful deployment of the complete HMS system. It serves as a guide for developers, stakeholders, and end-users to understand the features, constraints, and quality attributes of the Hostel Management System.Top of Form

## **1.2 Document conventions**

## **1.3 Project scope**

## **1.4 References**

# **2. Overall description**

## **2.1 Product perspective**

The Hostel Management System (HMS) is a standalone software designed to efficiently manage and streamline various aspects of hostel operations. It is not part of a larger system but is intended to be an independent solution catering specifically to the needs of hostel management.

*System Components:*

The major components of the Hostel Management System include:

1. User Interface: The interface allows users, including administrators, staff, and residents, to interact with the system seamlessly.
2. Database: The system relies on a robust database to store and manage information related to residents, incidents, room allocations, staff details, mess management, fees, and user accounts.
3. Functional Modules:
   * Incident Management Module: Facilitates the recording and tracking of incidents within the hostel.
   * Room Allocation Module: Manages the allocation of rooms to residents efficiently.
   * Feedback Module: Enables residents to provide feedback on hostel services.
   * Staff Management Module: Administers the details and roles of hostel staff.
   * Mess Management Module: Handles the planning and maintenance of hostel mess operations.
   * Fee Submission Module: Manages the submission and tracking of hostel fees.
   * Authentication Module: Responsible for user registration, login, and logout functionalities.

*Subsystem Interconnections:*

The subsystems are interconnected to ensure a smooth flow of information and processes. For example:

* The Incident Management Module may interact with the User Interface to display incident reports.
* The Room Allocation Module may communicate with the Database to update resident room assignments.
* The Authentication Module is linked to all other modules, ensuring secure access to relevant functionalities.

*External Interfaces:*

The Hostel Management System may interact with external systems for supplementary functionalities, such as financial systems for fee processing.

## **2.2 User classes and characteristics**

1. **Administrator:**
   * **Characteristics:** University staff with high privileges and technical expertise.
   * **Functions:** Comprehensive access for overall hostel management.
2. **Staff:**
   * **Characteristics:** University employees managing hostel operations.
   * **Functions:** Access to specific modules aligned with their roles.
3. **Residents (University Students):**
   * **Characteristics:** Students with varying technical skills and limited access.
   * **Functions:** Personal modules for feedback, room details, and fee submission.
4. **System Guests:**
   * **Characteristics:** Visitors with basic technical proficiency and minimal access.
   * **Functions:** Limited access for viewing general hostel information.
5. **Support/IT Personnel:**
   * **Characteristics:** Technical support staff ensuring system functionality.
   * **Functions:** High-level access for maintenance, troubleshooting, and support.

## **2.3 Operating environment**

The Hostel Management System (HMS) is tailored for operation within the university environment, specifically catering to the needs of students and hostel administrators. In terms of hardware, the system requires a robust server hosted by the university to manage centralized data and access control efficiently. This server should run a compatible server operating system such as Windows Server, Linux (e.g., Ubuntu Server, CentOS), or another suitable server OS.

End-users, including residents, staff, and administrators, can access the HMS through various client devices. Access to the HMS is facilitated through modern web browsers, including Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge.

Moreover, the HMS may need to integrate with other university systems, such as student databases or financial systems, to ensure seamless data flow and accuracy. Regular updates and compatibility checks will be conducted to maintain a versatile and secure operating environment, aligning with the dynamic nature of university systems and technologies.

## **2.4 Design and implementation constraints**

Several items and issues will shape the design and implementation of the Hostel Management System (HMS), imposing specific constraints on the development process. These constraints include:

1. **Regulatory Compliance:**

The HMS must adhere to relevant regulatory policies and standards governing data privacy and security, particularly as it involves sensitive information related to university students and staff.

1. **Hardware Limitations:**

The design and implementation must consider hardware limitations, including timing and memory requirements. The system should be optimized for efficient performance on a variety of devices, especially those used by students and staff.

1. **Database Technology:**

The selection of a specific database management system (DBMS) is a constraint, requiring compatibility with university standards and existing infrastructure. The chosen DBMS (e.g., MySQL, PostgreSQL) should align with the university's technology stack.

1. **Integration with Existing Systems:**

Integration with other university systems, such as student databases or financial systems, may be constrained by existing technologies and protocols. The HMS development must align with these systems to ensure seamless data flow.

1. **Design Conventions and Programming Standards:**

The development team must follow established design conventions and programming standards. If the customer's organization will be responsible for maintaining the software, adherence to their coding practices is essential.

1. **Parallel Operations:**

The development and deployment of the HMS must be conducted with minimal disruption to ongoing hostel operations. This constraint necessitates careful planning and coordination to avoid conflicts during system integration.

1. **Language Requirements:**

The system must support multiple languages to accommodate a diverse user base. Language localization should be a consideration during the design and implementation phases.

These constraints provide a framework for the development team, ensuring that the HMS aligns with university policies, technological infrastructure, and operational requirements while delivering a secure and effective hostel management solution.

## **2.5 Assumptions and dependencies**

Several assumptions and dependencies influence the requirements and successful implementation of the Hostel Management System (HMS):

**Assumptions:**

1. Third-Party Components: It is assumed that third-party components, such as database management systems, will function as expected and integrate seamlessly with the HMS. Any changes or issues with these components may impact development timelines.
2. Regulatory Compliance: Assumption that the university's regulatory environment remains stable, and any changes in data privacy or security regulations will be communicated promptly to ensure compliance.
3. Hardware Stability: It is assumed that the university's hardware infrastructure remains stable and can support the requirements of the HMS, including server capacities and network capabilities.
4. User Availability: Assumption that users, including administrators, staff, and students, have reliable internet access and compatible devices to interact with the HMS.
5. Availability of Development Resources: Assuming that necessary development resources, including skilled personnel and development tools, are consistently available throughout the project lifecycle.

**Dependencies:**

1. External System Integration: The HMS is dependent on successful integration with existing university systems, such as student databases and financial systems. Delays or changes in these external systems may impact the project timeline.
2. Web Browser Compatibility: Dependencies on web browser compatibility for user access. Any changes in browser standards or updates may affect the user experience.
3. University Policies: The project depends on adherence to university policies, including security protocols, coding standards, and technology preferences. Changes in these policies may require adjustments to the HMS.
4. External Software Updates: Dependencies on updates or changes in third-party software components (e.g., web frameworks, databases). Changes to these components may require adjustments to the HMS.

These assumptions and dependencies underscore the importance of ongoing communication, monitoring, and flexibility throughout the development process to address any unforeseen changes or challenges.

# **3. System features**

## **3.x System feature X**

## **3.x.1 Description**

## **3.x.2 Functional requirements**

# **4. Data requirements**

## **4.1 Logical data model**

## **4.2 Data dictionary**

## **4.3 Reports**

## **4.4 Data acquisition, integrity, retention, and disposal**

# **5. External interface requirements**

## **5.1 User interfaces**

## **5.2 Software interfaces**

## **5.3 Hardware interfaces**

## **5.4 Communications interfaces**

# **6. Quality attributes**

## **6.1 Availability:**

Aim for 99.9% availability to ensure uninterrupted access for users, with scheduled downtimes communicated in advance.

## **6.2 Correctness:**

Ensure that the system consistently produces correct results for tasks such as fee calculations, room allocations, and data retrieval.

## **6.3 Flexibility:**

Provide customizable settings or configurations to meet varying requirements of different hostels within the university.

## **6.4 Portability:**

Ensure compatibility with major web browsers and support for different operating systems (Windows, macOS, Linux).

## **6.5 Reliability:**

Minimize system failures, errors, and downtime to enhance user trust in the reliability of the HMS.

## **6.6 Reusability:**

Design modular and well-documented code, promoting the reuse of components in future developments.

## **6.7 Robustness:**

Implement error-handling mechanisms and conduct thorough testing to ensure the system gracefully handles unforeseen scenarios.

## **6.8 Testability:**

Design the system with a modular structure to facilitate unit testing, integration testing, and overall system testing.

Note: The criteria associated with each attribute should be used as measurable benchmarks during the development and testing phases to ensure the overall quality and effectiveness of the Hostel Management System.

# **7. Internationalization and localization requirements**

# **8. Other requirements**

## **Appendix A: Glossary**

## **Appendix B: Analysis models**

### **Activity diagram**

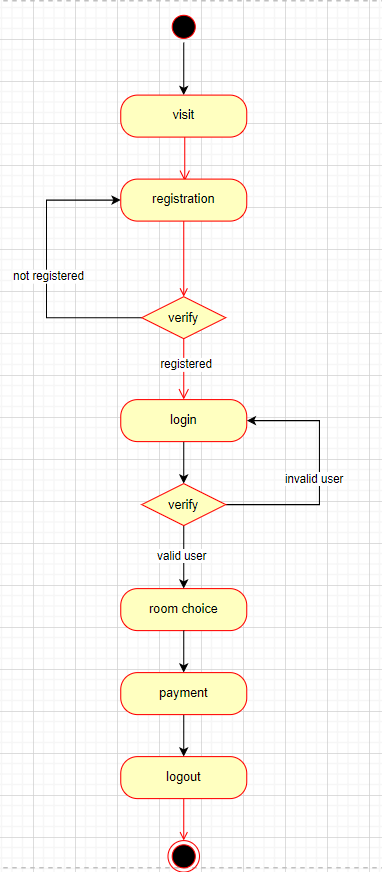


Figure 1 Activity diagram - student activity

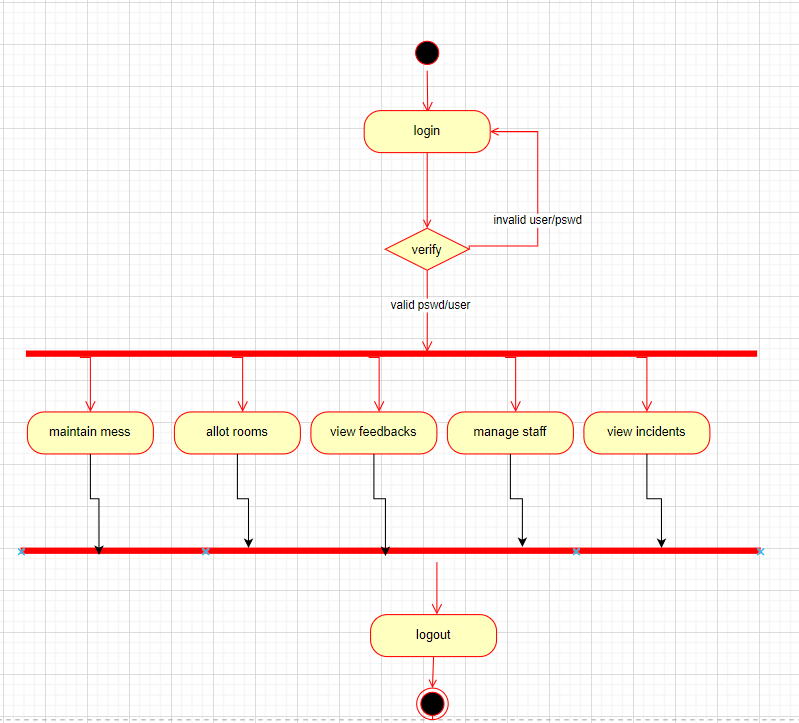


Figure 2 Activity diagram - admin activities