Software Requirements Specification (SRS)

For Hostel Management System (Python Django + SQLite Version)

Prepared by:

Nandan Murari SE22UARI110
Krishna Kalyan Thommandru SE22UARI192

Table of Contents

1. Introduction

1.1 Document Purpose

This document provides a detailed Software Requirements Specification (SRS) for the Hostel Management System (HMS) using Python Django and SQLite. The system is designed to manage hostel-related activities such as room allotment, student check-in/check-out, fee management, complaint handling, and staff management efficiently.

1.2 Product Scope

The HMS aims to automate hostel operations through a web-based platform built on Django with SQLite as the backend. It supports room allocation, digital fee tracking, complaint handling, and staff duty management.

1.3 Intended Audience and Document Overview

This document is intended for developers, testers, project managers, and stakeholders to understand the project's scope, features, and technical details.

1.4 Definitions, Acronyms, and Abbreviations

HMS: Hostel Management System
ORM: Object-Relational Mapping
Django: A high-level Python Web framework
SQLite: Lightweight SQL Database

1.5 Document Conventions

Standard IEEE formatting conventions followed.

1.6 References and Acknowledgments

IEEE SRS Template
Django Documentation
University Hostel Policies

2. Overall Description

2.1 Product Overview

The HMS provides a centralized portal using Django for managing hostel operations. SQLite will serve as the default local database.

2.2 Product Functionality

- Student Registration & Profile Management
- Room Allocation & Availability Tracking
- Hostel Fee Management & Online Payments
- Complaint Registration & Tracking
- Staff Duty Roster Management
- Access Control Integration (RFID/Biometrics)

2.3 Design and Implementation Constraints

The application will be developed using Django (Python) with SQLite as the primary database. The solution should be deployable on any platform supporting Python 3.x.

2.4 Assumptions and Dependencies

The system depends on the Django web framework and SQLite database engine. Users must have internet access and browsers to use the web interface.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces: Web-based UI (HTML/CSS) rendered using Django templates.

3.1.2 Hardware Interfaces: Optional RFID/Biometric scanner integration.

3.1.3 Software Interfaces: Django ORM interfacing with SQLite; integration with university student database via APIs.

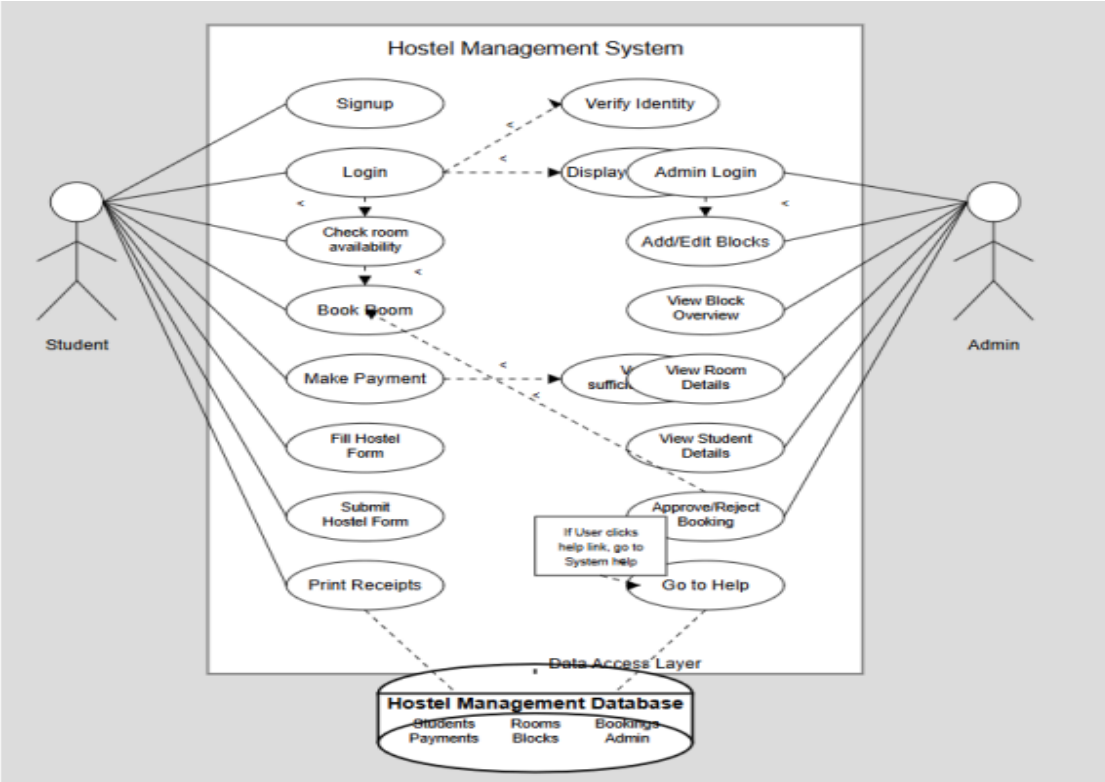
3.2 Functional Requirements

7. FR-1: Student Registration – Register with student ID, name, email, phone.

8. FR-2: Room Allocation – Admin assigns rooms based on availability.
9. FR-3: Complaint Management – Submit/view status of complaints.
10. FR-4: Fee Payment System – Support UPI, card payments (via third-party).
11. FR-5: Security & Access Control – Integration with RFID/Biometric APIs.

3.3 Use Case Model

A UML use case diagram will illustrate system actors and interactions.



4. Other Non-functional Requirements

4.1 Performance Requirements

Support up to 500 concurrent users with sub-3-second response times.

4.2 Safety and Security Requirements

Implement SSL, hashed passwords, and role-based access control.

4.3 Software Quality Attributes

Reliability: 99.5% uptime
Maintainability: Django's modular structure

Portability: Django supports deployment across Windows/Linux/Mac
Scalability: SQLite can be replaced with PostgreSQL/MySQL in future

5. Other Requirements

The solution must adhere to university IT policies on data privacy.

6. Appendices

Appendix A: Data Dictionary

Django ORM will be used to define data models and schema stored in SQLite.

Appendix B: Group Log

Record of weekly development tasks, meetings, and testing milestones.