Practical 4

Software Requirement Specification (SRS) Development for the Hospital Management System.

Aim: Perform a requirement analysis and develop a Software Requirement Specification (SRS) sheet for the Hospital Management System. The SRS should include the following sections:

- **Functionality:** Describe what the software is supposed to do.
- **External Interfaces:** Explain how the software interacts with people, the system's hardware, other hardware, and other software.
- **Performance:** Outline the expected speed, availability, response time, recovery time, and other performance-related characteristics of the software functions.
- **Attributes:** Define considerations related to portability, correctness, maintainability, security, and other relevant attributes.
- **Design Constraints:** Specify any design constraints imposed on the implementation, such as required standards, implementation languages, and database integrity policies.

4.1 Overview

A Hospital Management System (HMS) is a digital solution designed to streamline and automate hospital operations. It replaces traditional manual record-keeping with an efficient, organized system that enables seamless management of patient records, doctor schedules, billing, and medical history. The HMS aims to enhance the patient experience by providing faster access to medical services, improving accountability, and reducing human errors.

This Software Requirement Specification (SRS) document follows the IEEE SRS standard and outlines the functional and non-functional requirements of the system, as well as external interfaces, constraints, and dependencies.

4.2 Purpose

The primary purpose of this HMS is to provide an automated platform for managing hospital resources, patient appointments, and billing transactions. The system enables users to register patients, schedule doctor appointments, process medical billing, and generate reports while allowing administrators to monitor hospital activities and track patient history efficiently.

4.3 Document Conventions

This document follows structured formatting using numbered sections, bullet points, and industry-standard terminology for clarity and ease of reference. It is intended for:

- **Hospital Administrators** who manage user accounts, patient records, and operational policies.
- **Developers** who implement system requirements.

- **Testers** who validate system functionality and performance.
- **Stakeholders** who review system capabilities and constraints.

This document follows IEEE 830-1998: Recommended Practice for Software Requirements Specifications.

4.4 Overall Description

Product Perspective

The HMS is a standalone software system designed to digitize hospital operations. It integrates with existing hospital infrastructure, replacing traditional record-keeping methods with a centralized database.

Product Functions

The system will provide:

- **Patient Management:** Register and manage patient records.
- **Doctor Management:** Maintain doctor profiles and schedules.
- **Appointment Scheduling:** Book, modify, or cancel appointments.
- **Billing and Payments:** Generate invoices and manage payments.
- **Reports:** Generate reports on patient history, billing, and hospital revenue.
- **Authentication:** Secure login/logout functionality for users and admins.

Users of the HMS

- **Hospital Staff:** Administrators who manage patient records, users, and transactions.
- **Doctors & Nurses:** Registered medical staff who update patient records.
- **System Administrators:** IT personnel responsible for system maintenance.

General Constraints

- The system must comply with data privacy regulations (e.g., HIPAA, GDPR).
- It should support at least 50,000 patient records.
- The system must be operational 24/7 with minimal downtime.

4.5 Specific Requirements

4.5.1 Functional Requirements

1. **User Authentication & Authorization:** Secure login and role-based access.
2. **Patient Registration:** CRUD (Create, Read, Update, Delete) operations for patient records.
3. **Doctor & Staff Management:** Maintain profiles, schedules, and specialties.
4. **Appointment Scheduling:** Track and manage patient appointments.
5. **Billing & Payments:** Automated invoice generation and payment processing.

6. **Search & Filter:** Advanced search options for patient records.
7. **Reports & Analytics:** Generate reports on system usage and patient data.
8. **Notifications:** Email/SMS reminders for appointments and billing updates.

4.5.2 External Interfaces

1. **User Interface (UI):** A web-based interface with an intuitive dashboard for users to interact with the system.
2. **Hardware Interface:** The system must support barcode scanners for patient identification, printers for reports, and mobile devices for remote access.
3. **Software Interfaces:** The system integrates with SQL databases for storage.
4. **Communication Interfaces:** The system supports email, SMS, and push notifications to alert users about appointments and system updates.

4.5.3 Non-Functional Requirements

1. **Performance:** The system must handle maximum concurrent users, with response times under 2 seconds.
2. **Security:** Implements encryption, multi-level authentication, and role-based access control to prevent unauthorized access.
3. **Usability:** The UI should be user-friendly, accessible, and follow WCAG guidelines.
4. **Scalability:** The system must support expansion without performance degradation.
5. **Availability:** Ensures maximum uptime, with automated backups for data recovery.

4.5.4 Design Constraints

1. **Database Management:** The system should support both SQL and NoSQL databases depending on scalability and performance needs.
2. **Technology Stack:** The system should be built using Node.js for the backend, React.js for the frontend, and MongoDB as the primary database.
3. **Security Compliance:** The system must encrypt user data, follow GDPR compliance, and implement multi-factor authentication (MFA).
4. **Performance Optimization:** The system should implement load balancing and caching mechanisms for high availability.

4.5.5 System Attributes

1. **Portability:** The system must be accessible on Windows, macOS, Linux, and mobile devices.
2. **Maintainability:** Modular design to allow easy updates and fixes.
3. **Reliability:** Ensures consistent operation with failover mechanisms.
4. **Scalability:** The system must handle increasing users and patient records efficiently.
5. **Interoperability:** Ability to integrate with third-party systems for extended functionality.
6. **Extensibility:** The system should allow future enhancements and new feature additions with minimal modifications.

4.6 Implementation Details and Hardware Requirements

Implementation Details

- **Programming Language:** The backend will be implemented using PHP while the front end will be built with HTML, CSS, JavaScript.
- **Database:** SQL from XAMPP MySQL.
- **Security Features:** I will implement an encryption algorithm on database.
- **Hosting:** I will host the Hospital Management System Project in infinity host because it's free.
- **Testing:** I will be doing Unit testing, System testing, and Integration testing for my project.
- **Backup & Recovery:** Automatic daily backups with a disaster recovery plan.

Hardware Requirements

- **Server:** Minimum 4-core CPU, 8GB RAM, SSD storage (500GB+), and high-speed internet connectivity.
- **Client Devices:** Only mobile supported applications because patients can't carry windows system & other system, and reason for not in IOS because my client like IOS devices.
- **Networking:** I am now using Marwadi University internet and developing my mini project for submission.
- **Peripheral Devices:** Barcode scanners, receipt printers, and biometric authentication devices.