**HOSPITAL MANAGEMENT SYSTEM**

**PROJECT SCOPE:**

The project is a web application that aims to provide effective management for some of the services offered by hospital such as keeping patient details, doctor appointments, etc.  
Admin, Doctors and Users (Hospital Staff) will use the system.

**ABSTRACT:**

The Hospital Management System (HMS) is designed to streamline and automate the operations of healthcare facilities. This project aims to provide an integrated solution for managing hospital resources, patient information, doctor schedules, and administrative tasks. By leveraging modern software engineering practices, the system ensures efficient and secure handling of sensitive data, such as patient records, appointments, and medical prescriptions.

The HMS includes various modules, each focusing on a critical aspect of hospital management:

* **Employee Management**: Handles the creation, updating, and management of hospital staff, including doctors and administrative personnel. Role-based access control ensures that only authorized users can access specific features.
* **Patient Management**: Manages patient records, including personal details, medical history, and treatment plans. It also facilitates patient registration and ensures the accuracy and privacy of patient data.
* **Appointment Scheduling**: Allows users to book appointments with doctors based on availability. The system tracks doctor schedules, shifts, and ensures conflict-free appointments.
* **Medication and Testing**: Provides doctors with the ability to prescribe medications and order tests for patients. It keeps a detailed record of prescriptions and test results, ensuring continuity of care.

The system is built using robust technologies and follows best practices in software development, including secure coding standards to protect against SQL injection and other vulnerabilities. The HMS is designed to be scalable and adaptable, capable of serving hospitals of various sizes and specialties.

**Software Requirements Specification (SRS) for Hospital Management System**

**1. Introduction**

**1.1 Purpose** The purpose of this document is to outline the software requirements for the Hospital Management System (HMS). The system is intended to streamline and automate the administrative, medical, and financial operations of a healthcare facility, ensuring efficient management of resources and high-quality patient care.

**1.2 Scope**

The Hospital Management System will manage hospital operations, including employee management, patient registration, appointment scheduling. The system will provide role-based access, ensuring that each user (admin, doctor, or user) can access only the functions relevant to their role.

**1.3 Definitions, Acronyms, and Abbreviations**

* **HMS**: Hospital Management System
* **EMR**: Electronic Medical Record
* **SRS**: Software Requirements Specification

**1.4 References**

* Hospital workflows and operational manuals.
* Industry standards for healthcare software (e.g., HIPAA compliance).

**1.5 Overview**

This document details the functional and non-functional requirements for the HMS, including system features, user interfaces, performance criteria, and security measures.

**2. Overall Description**

**2.1 Product Perspective**

The HMS is a stand-alone system that will be deployed within hospital premises. It will interact with external systems like insurance providers, lab systems, and pharmacies for data exchange.

**2.2 Product Functions**

* **Employee Management**: Manage hospital staff information, including roles, shifts, and access levels.
* **Patient Management**: Register new patients, maintain EMRs, and track patient history.
* **Appointment Scheduling**: Allow patients to book appointments and manage doctor schedules.
* **Reporting**: Generate various reports, including patient history, doctor’s report and user’s report.

**2.3 User Characteristics**

* **Admin**: Manage the entire system, including user roles, permissions, and hospital resources.
* **Doctors**: Access patient records, manage appointments, prescribe medications, and order tests.
* User: Book appointments, view appointments and register new patients..

**2.4 Constraints**

* Compliance with healthcare regulations.
* Data security and privacy must be ensured.
* System should be scalable to accommodate growth.

**2.5 Assumptions and Dependencies**

* Users will have basic computer literacy.
* The system will require regular updates to accommodate changing healthcare regulations and hospital needs.

**3. Functional Requirements**

**3.1 User Management**

* The system shall allow the creation, modification, and deletion of user accounts.
* The system shall enforce role-based access control to ensure users can only access the functions relevant to their role.

**3.2 Patient Management**

* The system shall allow the registration of new patients and maintain a detailed electronic medical record (EMR) for each patient.
* The system shall allow doctors and authorized staff to update patient records with new information, including diagnoses, treatments, and test results.

**3.3 Appointment Scheduling**

* The system shall allow patients to book, cancel, and reschedule appointments with available doctors.
* The system shall ensure that appointments do not overlap and are confirmed by both the patient and the doctor.

**3.6 Reporting**

* The system shall generate various reports, including patient histories, doctor’s work report and user’s work report.

**4. Non-Functional Requirements**

**4.1 Performance Requirements**

* The system shall support at least 100 concurrent users.
* The system should process requests within 2 seconds.

**4.2 Security Requirements**

* The system shall protect sensitive data, including patient records and payment information.
* The users of all roles shall have login credentials to utilise the system.

**4.3 Usability Requirements**

* The system shall have a user-friendly interface that is intuitive for users with varying levels of computer literacy.
* The system shall provide help documentation and support for users.

**4.4 Reliability and Availability**

* The system shall have a good uptime.

**4.5 Scalability**

* The system shall be scalable to accommodate a growing number of users and patient records without degradation in performance.

**4.6 Compliance**

* The system shall comply with healthcare regulations, including data privacy and security standards.

**5. System Interfaces**

**5.1 User Interfaces**

* Web-based interfaces for different user roles (admin, doctor, user [Hospital Staff]).
* Responsive design to ensure accessibility from various devices.

**5.2 Hardware Interfaces**

* The system shall interact with hospital hardware such as printers.

**5.3 Software Interfaces**

* The system shall integrate with third-party systems, including insurance providers, laboratory systems, and pharmacies.

**5.4 Communication Interfaces**

* The system shall use HTTPS for secure communication over the network.
* The system shall support API integration for data exchange with external systems.

**6. Other Requirements**

**6.1 Data Migration**

* The system shall include tools to migrate existing patient records and other relevant data into the new system.

**6.2 Testing**

* The system shall undergo rigorous testing, including unit testing.

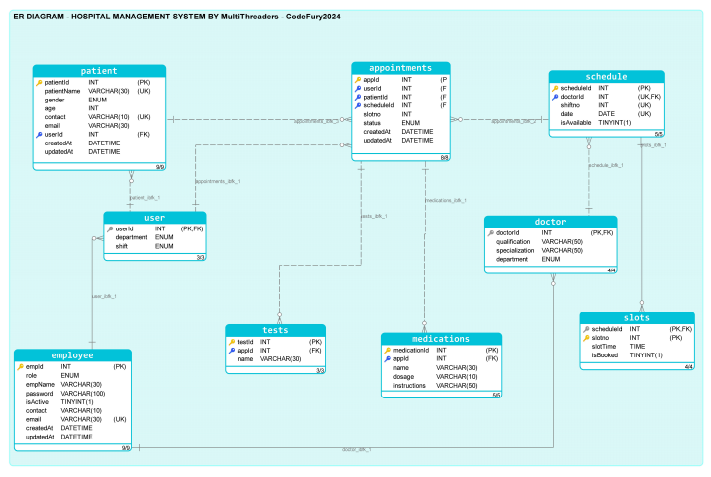
**6.3 Deployment**

* The system shall be deployed on the hospital's local servers with provisions for future cloud deployment.

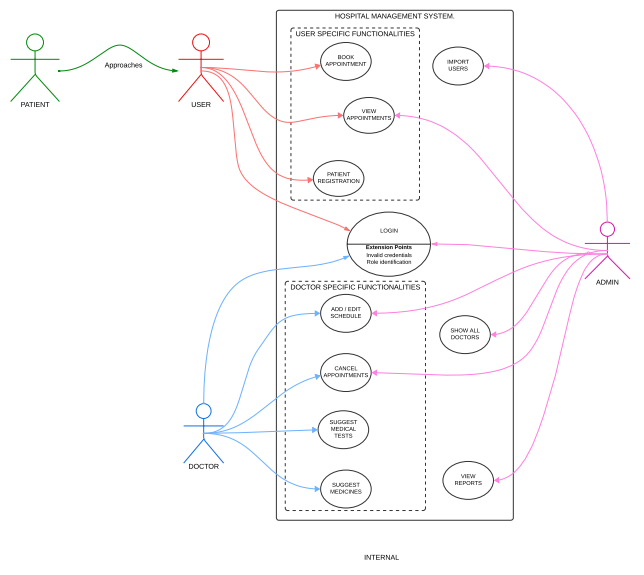
**6.4 Maintenance**

* The system shall include maintenance plans, including regular updates and security patches.

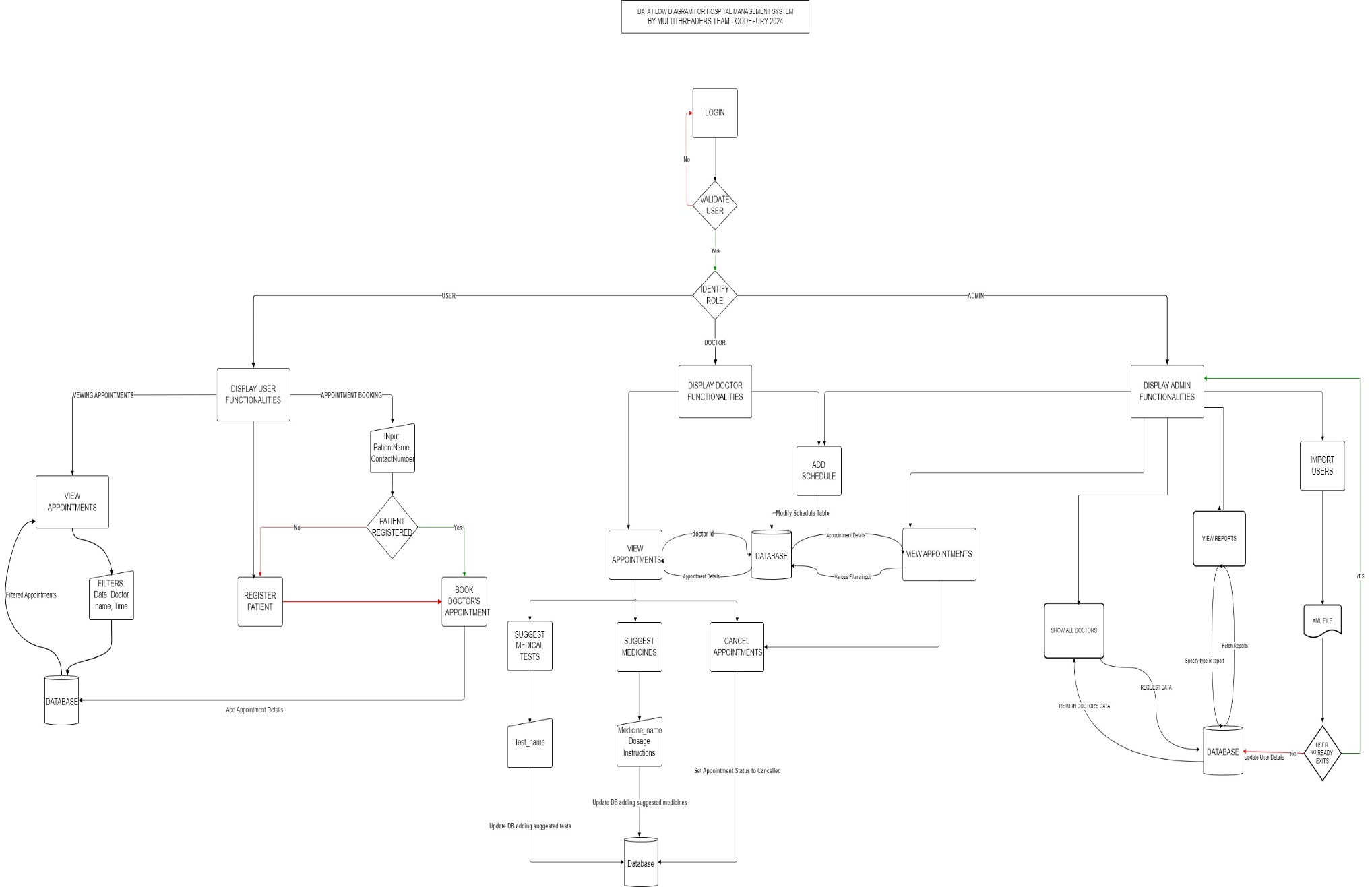
**SCHEMA DIAGRAM:**



**USE CASE DIAGRAM:**



**DATA FLOW DIAGRAM:**



**Team Learnings from the Hospital Management System Project**

Throughout the development of the Hospital Management System (HMS), our team has gained significant insights and skills that have enriched our knowledge and experience in software development. Here are some of the key learnings from the project:

1. **Best Coding Practices**:
   * We have developed a strong understanding of best coding practices, including code modularity, readability, and maintainability. By following design patterns and principles like SOLID, we ensured that our codebase is scalable and easy to manage. Code reviews and pair programming helped us consistently apply these practices, leading to a cleaner and more efficient codebase.
2. **Java Concepts Mastery**:
   * The project provided us with hands-on experience in various Java concepts, including Classes, Interfaces, JDBC, JUnit, and more. We learned how to effectively use object-oriented principles to design our system, implement database connectivity using JDBC, and write unit tests to ensure the reliability of our code. This solidified our understanding of Java as a powerful language for backend development.
3. **UI Development with HTML, Bootstrap, and JavaScript**:
   * We learned how to create user-friendly and aesthetically pleasing interfaces using HTML, Bootstrap, and JavaScript. Understanding how to structure HTML, apply responsive design using Bootstrap, and add dynamic behavior with JavaScript allowed us to deliver a polished user experience. We also became proficient in customizing Bootstrap components and utilizing JavaScript libraries to enhance UI interactions.
4. **XML and JSON Data Handling**:
   * A crucial aspect of our project was learning how to read and manipulate data from XML and JSON files using JavaScript. We gained practical experience in parsing these data formats and displaying the information on our UI, which is essential for integrating various data sources into our system. This skill is valuable for any project that involves data exchange and API integration.
5. **Database Management with MySQL**:
   * Managing the database was a critical component of the HMS, and we learned how to design, implement, and manage databases using MySQL. We gained expertise in writing efficient SQL queries, designing relational database schemas, and ensuring data integrity. Our experience with MySQL has also taught us about database optimization and the importance of indexing and normalization.
6. **Team Collaboration and Coordination**:
   * One of the most valuable lessons from this project was learning how to collaborate effectively as a team. We developed strong communication skills, learned how to divide tasks based on team members' strengths, and coordinated our efforts to meet deadlines. We utilized tools like version control (Git), project management software, and regular meetings to ensure everyone was aligned with the project goals. This experience has prepared us to work effectively in a team environment in future projects.

**CONCLUSION:**

The Hospital Management System (HMS) project has been a comprehensive and rewarding endeavor for our team. Through this project, we have successfully developed a robust system that addresses the complex needs of hospital management, from patient registration and appointment scheduling and more.

This project has allowed us to apply and deepen our understanding of essential software development principles and technologies. By integrating Java, MySQL, HTML, Bootstrap, and JavaScript, we created a scalable and user-friendly application that can streamline hospital operations and improve patient care.

Moreover, the project has fostered our ability to work collaboratively as a team, honing our skills in communication, task management, and deadline adherence. The challenges we faced and overcame have not only enhanced our technical capabilities but also prepared us for the collaborative nature of real-world software development.