**Software Requirements Specification (SRS) for Healthcare System Using MySQL**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to provide a detailed Software Requirements Specification (SRS) for the development of a Healthcare Management System (HMS) using MySQL as the backend database. This system is intended to streamline healthcare services by managing patient records, appointments, medical staff, billing, and inventory management of medical supplies.

**##### 1.2 Scope**

The HMS will cover:

- Patient management (registration, medical history, treatments)

- Appointment scheduling and tracking

- Healthcare professional (doctors, nurses, etc.) management

- Medical billing and financial tracking

- Inventory management for medical supplies

- Reporting and analytics

##### 1.3 Definitions, Acronyms, and Abbreviations

- HMS: Healthcare Management System

- DBMS: Database Management System

- MySQL: Relational database management system

- EMR: Electronic Medical Record

##### 1.4 References

- MySQL Documentation

- HL7 (Health Level Seven) Standards for healthcare software

- HIPAA Compliance Guidelines

---

#### 2. Overall Description

##### 2.1 Product Perspective

The HMS will act as a centralized system where healthcare providers can manage their patients, medical staff, inventory, and financials. MySQL will be used as the database to ensure efficient, secure, and scalable storage of data.

##### 2.2 Product Features

Key features of the system include:

1. \*\*Patient Management\*\*: Registration, view, and update patient records.

2. \*\*Appointment Scheduling\*\*: Managing patient appointments, notifications, and tracking.

3. \*\*Staff Management\*\*: Assigning roles to doctors, nurses, etc., and managing their profiles.

4. \*\*Billing System\*\*: Generation and management of invoices for treatments and services.

5. \*\*Inventory Management\*\*: Tracking stock of medicines, medical devices, and supplies.

6. \*\*Analytics and Reports\*\*: Customizable reports for patient history, financials, and operational statistics.

##### 2.3 User Classes and Characteristics

- \*\*Patients\*\*: Can schedule appointments and view their medical records.

- \*\*Doctors/Nurses\*\*: Can access patient medical histories, update treatments, and add prescriptions.

- \*\*Administrative Staff\*\*: Manage appointments, billing, and inventory.

- \*\*System Administrator\*\*: Maintain the database and provide technical support.

##### 2.4 Operating Environment

- \*\*Database\*\*: MySQL 8.0 or higher

- \*\*Operating System\*\*: Linux, Windows, or macOS

- \*\*Server Requirements\*\*: PHP/Java/Python as the backend server connected with MySQL

- \*\*Client\*\*: Web browser (Google Chrome, Firefox, Safari)

##### 2.5 Design and Implementation Constraints

- The system must comply with HIPAA regulations for storing patient data.

- All data should be encrypted in transit (SSL/TLS) and at rest.

- The system must provide role-based access control to ensure data privacy.

##### 2.6 Assumptions and Dependencies

- Users must have a stable internet connection.

- Medical data will be input manually by staff members.

- System upgrades will follow MySQL release cycles.

---

#### 3. System Features

##### 3.1 Patient Management

- \*\*Description\*\*: Allows the registration, update, and deletion of patient records. It also maintains the patient's medical history, allergies, ongoing treatments, and prescriptions.

- \*\*Inputs\*\*: Patient name, contact information, medical history.

- \*\*Outputs\*\*: Registered patients list, individual patient records.

- \*\*MySQL Tables\*\*:

- `patients`: Stores personal data of the patient.

- `medical\_history`: Stores historical data of treatments and diagnoses.

##### 3.2 Appointment Scheduling

- \*\*Description\*\*: Doctors and patients can manage appointments, reschedule, and cancel bookings.

- \*\*Inputs\*\*: Patient ID, doctor ID, appointment date, reason for the visit.

- \*\*Outputs\*\*: Appointment confirmation or reschedule notices.

- \*\*MySQL Tables\*\*:

- `appointments`: Stores all appointments.

- `notifications`: Stores patient notifications.

##### 3.3 Healthcare Professional Management

- \*\*Description\*\*: This module handles the registration, update, and tracking of all healthcare professionals.

- \*\*Inputs\*\*: Name, role, specialization, shift timings.

- \*\*Outputs\*\*: Doctors' availability, staff schedules.

- \*\*MySQL Tables\*\*:

- `staff`: Stores all doctors, nurses, and administrative staff data.

- `shifts`: Manages shifts and schedules of the staff.

##### 3.4 Billing System

- \*\*Description\*\*: Manages the generation of bills for services rendered, including treatments and consultations. It also tracks payments and outstanding dues.

- \*\*Inputs\*\*: Patient ID, services offered, cost, payment method.

- \*\*Outputs\*\*: Generated invoices, payment receipts.

- \*\*MySQL Tables\*\*:

- `invoices`: Contains invoice details.

- `payments`: Stores payment transaction details.

##### 3.5 Inventory Management

- \*\*Description\*\*: Tracks the availability and usage of medical supplies and medicines.

- \*\*Inputs\*\*: Medicine name, quantity, usage date, stock limit.

- \*\*Outputs\*\*: Alerts for low stock, inventory reports.

- \*\*MySQL Tables\*\*:

- `inventory`: Stores details of medical supplies.

- `orders`: Manages reordering of items when stock is low.

##### 3.6 Reporting and Analytics

- \*\*Description\*\*: Generates customized reports and analytics for patient history, financial tracking, and overall hospital performance.

- \*\*Inputs\*\*: Date range, report type (financial, patient history).

- \*\*Outputs\*\*: Generated reports (PDF, Excel, etc.).

- \*\*MySQL Tables\*\*:

- `reports`: Stores the type and date of generated reports.

- Data pulled from other modules as needed.

---

#### 4. External Interface Requirements

##### 4.1 User Interfaces

- Web-based portal with forms for data entry and dashboards for real-time data display.

- Responsive design to work on desktops, tablets, and mobile devices.

##### 4.2 Hardware Interfaces

- Standard web server hosting the MySQL database.

- Backup systems and redundant storage for healthcare data protection.

##### 4.3 Software Interfaces

- Integration with external APIs for medical devices, insurance systems, or patient portals.

- MySQL database will be accessed using server-side scripts written in PHP, Python, or Java.

##### 4.4 Communication Interfaces

- Secure communication between client and server using SSL/TLS.

- RESTful APIs for internal and third-party service integration.

---

#### 5. Performance Requirements

- The system should support up to 1,000 concurrent users without significant performance degradation.

- All database queries should return results within 1 second for up to 100,000 records.

- The system should be scalable to accommodate future expansion and additional modules.

---

#### 6. Security Requirements

- \*\*Encryption\*\*: All sensitive patient data must be encrypted in the database.

- \*\*Authentication\*\*: The system must enforce strong password policies and multi-factor authentication for administrative roles.

- \*\*Authorization\*\*: Role-based access control must be implemented to ensure data access is limited to authorized personnel only.

- \*\*Audit Logs\*\*: The system should log all critical actions performed on patient data, billing, and inventory.

---

#### 7. Database Design

##### 7.1 MySQL Database Structure

- \*\*patients\*\*

- `patient\_id` (PK)

- `name`

- `dob`

- `contact`

- `address`

- `medical\_history\_id` (FK)

- \*\*medical\_history\*\*

- `history\_id` (PK)

- `patient\_id` (FK)

- `disease`

- `treatment`

- `allergies`

- \*\*appointments\*\*

- `appointment\_id` (PK)

- `patient\_id` (FK)

- `doctor\_id` (FK)

- `date`

- `reason`

- \*\*staff\*\*

- `staff\_id` (PK)

- `name`

- `role`

- `specialization`

- \*\*inventory\*\*

- `item\_id` (PK)

- `name`

- `quantity`

- `reorder\_level`

---

#### 8. Other Non-Functional Requirements

##### 8.1 Reliability

- The system must have an uptime of 99.9%, ensuring availability for critical healthcare operations.

##### 8.2 Maintainability

- Regular database backups will be required. The system will allow easy upgrades for both MySQL and application features.

##### 8.3 Portability

- The system should be able to migrate between different hosting platforms with minimal configuration changes.

##### 8.4 Compliance

- The system must comply with HIPAA and other regional healthcare data privacy regulations.

---

This document outlines the high-level requirements for a healthcare system based on MySQL. Detailed specifications of each module and component will be provided during the design and development phases.