Software Requirements Specification

for

Clinical Management System

Prepared by:
Anirudh M K
Hiba Nm
Aashil
Abhinav

Contents

1 Introduction

1.1 Purpose

The Clinical Management System (CMS) is a web-based application designed to enhance operational efficiency in healthcare facilities. It addresses critical challenges such as prolonged patient wait times, manual record-keeping, and administrative overburden by providing an integrated solution for appointment scheduling, patient record management, laboratory test processing, medication dispensing, billing, and website administration. The primary objectives are to reduce average patient wait times (currently averaging 20 minutes beyond scheduled appointments, per industry studies), enable real-time data access for medical staff, and provide administrators with comprehensive oversight of clinic operations. This SRS defines the system's features, interfaces, and constraints, serving as a detailed specification for development teams and a formal proposal for healthcare regulatory approval.

1.2 Intended Audience

This document is intended for the following stakeholders:

- **Clinic Administrators**: Responsible for managing resources, including staff, equipment, and inventory, to ensure seamless clinic operations.
- **Medical Staff**: Physicians and support personnel who will utilize CMS to access patient data, schedule appointments, and order diagnostic tests efficiently.
- **Patients**: End-users who will interact with CMS to book appointments, review test results, and process payments electronically.
- **Development and Testing Teams**: Technical professionals tasked with designing, implementing, and validating the CMS based on this specification.
- **Stakeholders**: Executives, investors, and insurance representatives focused on operational improvements and patient satisfaction metrics.

1.3 Scope

The CMS provides a comprehensive solution for clinic management, encompassing appointment scheduling, patient record maintenance, laboratory test coordination, medication inventory control, billing automation, and website content management. Key functionalities include enabling patients to schedule appointments remotely, allowing medical staff to access patient histories instantly, and permitting administrators to monitor inventory levels and update public-facing clinic information from a centralized interface. The system leverages SQLite databases ('users.db' for authentication, 'clinic.db' for operational data) to ensure data integrity and compliance with HIPAA regulations. Future enhancements may include predictive analytics for appointment adherence and a mobile application, contingent on budget and regulatory considerations.

1.4 Definitions and Acronyms

The following terms and abbreviations are used throughout this document:

- **CMS**: Clinical Management System – A web-based platform for healthcare facility management.
- **Appointment**: A scheduled interaction between a patient and healthcare provider.
- **Patient Records**: Electronic repository of a patient's medical history and treatment details.
- **Lab Tests**: Diagnostic procedures ordered and tracked through CMS.
- **Medicines**: Inventory of pharmaceuticals managed by the system.
- **Website Management**: Tools for updating and monitoring the clinic's online presence.
- **SQLite Databases**: Data storage systems ('users.db' for user credentials, 'clinic.db' for clinic operations).
- **HIPAA**: Health Insurance Portability and Accountability Act – U.S. regulation mandating patient data privacy.

2 Overall Description

2.1 User Needs

The CMS addresses the following user requirements:

- **Patients**: Require efficient appointment booking, access to test results, and simplified payment processes.
- **Medical Staff**: Need rapid access to accurate patient data and streamlined test ordering capabilities.
- **Administrators**: Demand visibility into scheduling, inventory, and operational status for effective resource management.
- **All Users**: Expect robust security, data accuracy, and an intuitive interface requiring minimal training.

2.2 Assumptions and Dependencies

The system's design is based on the following:

- **Assumptions**:
  - Users have reliable internet access for system interaction.

  - Data entry is accurate and consistent to ensure reliable outputs.
  - Compliance with HIPAA regulations is mandatory.

- **Dependencies**:
  - SQLite databases (‘users.db‘ and ‘clinic.db‘) must be stable and secure.
  - Integration with existing clinic systems (e.g., laboratory equipment) is required.
  - Project success relies on sufficient funding and staff adoption.

3 System Requirements

3.1 Functional Requirements

The CMS shall provide the following capabilities:

- **User Registration and Profile Management**:
  - Enable registration for patients, medical staff, and administrators with name, email, and phone number, verified via SMS.
  - Allow profile updates, such as adding certifications or changing contact details.
- **Appointment Scheduling**:
  - Facilitate patient-initiated appointment booking with confirmation notifications.
  - Provide staff with visibility into schedules and the ability to adjust as needed.
- **Lab Test Ordering**:
  - Support requests for diagnostic tests with automated lab notifications.
  - Deliver test results to the system for review and action by medical staff.
- **Record Management**:
  - Enable medical staff to document patient visits and treatments.
  - Generate administrative reports, such as patient visit summaries.
- **Medicine Dispensing**:
  - Track medication inventory levels and update stock records.
  - Automate dispensing records and trigger reorder alerts at predefined thresholds.
- **Billing and Payment**:
  - Calculate charges for services and generate electronic invoices.
  - Process payments via credit card or insurance integration.
- **Website Administration**:
  - Allow updates to clinic website content, such as announcements or service details.
  - Provide analytics on website traffic for administrative review.

3.2 External Interface Requirements

The CMS shall integrate with:

- **User Interface**: A web-based portal displaying appointment details, patient lists, and administrative controls.
- **Clinic Systems Integration**: Compatibility with laboratory equipment and existing patient databases.
- **Lab Integration**: Automated communication for test orders and result retrieval.

3.3 System Features

The CMS shall include:

- **User Authentication**: Role-based access control with secure login credentials.
- **Record Management Module**: Searchable database for patient records.
- **Scheduling Engine**: Conflict-free appointment allocation system.
- **Database Management**: Secure SQLite databases (‘users.db‘, ‘clinic.db‘) for data storage.
- **Transaction Processing**: Automated billing and payment workflows.

3.4 Subsystem Requirements

The CMS shall meet the following subsystem specifications:

- **Hardware Requirements**:
  - **Server**: Minimum 8 GB RAM, 4-core CPU, 500 GB SSD storage for primary operations; additional 1 TB HDD for backups.
  - **Client Devices**: Devices with web browsers (e.g., desktops, laptops, tablets) supporting 1920x1080 resolution.
  - **Network**: 10 Mbps minimum bandwidth, with redundant 5 Mbps backup connection for uninterrupted service.
  - **Peripherals**: Barcode scanners for medication dispensing and printers for billing receipts.
- **Software Requirements**:
  - **Operating System**: Server support for Linux (Ubuntu 20.04 or later) or Windows Server 2019; clients on Windows 10+, macOS 11+, or equivalent.
  - **Web Server**: Apache 2.4+ or Nginx 1.18+ with SSL/TLS certification for secure data transmission.

  - **Database**: SQLite 3.x with automated backup scheduling and integrity checks.
  - **Browser Support**: Chrome 90+, Firefox 85+, Edge 90+ with JavaScript and cookies enabled.
  - **Security Software**: Firewall and intrusion detection system (e.g., pfSense, Snort) for network protection.

- **Development Tools**:
  - **Languages**: Python 3.8+ (backend), HTML5/CSS3/JavaScript ES6 (frontend).
  - **Frameworks**: Flask 2.0+ or Django 4.0+ with REST API support for extensibility.
  - **Version Control**: Git with GitHub or GitLab for collaborative development and rollback capabilities.
  - **Testing Tools**: PyTest for unit testing, Selenium for browser automation, and JMeter for load testing.

- **Subsystem Integration**:
  - **Authentication Subsystem**: Multi-factor authentication (MFA) support using SMS or authenticator apps.
  - **Backup Subsystem**: Daily incremental backups to a secondary server or cloud storage (e.g., AWS S3).
  - **Scalability Subsystem**: Load balancer (e.g., HAProxy) to distribute traffic across multiple servers as clinic size grows.

3.5 Nonfunctional Requirements

The CMS shall meet the following standards:

- **Performance**: Process 100 simultaneous bookings with responses under 2 seconds.
- **Security**: Encrypt data and comply with HIPAA to prevent unauthorized access.
- **Scalability**: Support expansion from small clinics to large healthcare networks.
- **Reliability**: Achieve 99.9% uptime, with no more than 8 minutes of annual downtime.
- **Usability**: Feature an intuitive interface requiring minimal training.