MeDoc Software Design Document

Introduction

- Overview:

MeDoc is a software development that is designed to facilitate the process of reaching healthcare services easing the patient doctor interaction

- Purpose:

The primary objective of this software is to enhance the patient-doctor interaction experience. This document aims to provide comprehensive details about the project's features and our proposed implementation strategies.

1. Objectives:

The healthcare system application aims to provide a comprehensive platform for managing and improving healthcare services, with a primary focus on enhancing patient care and streamlining administrative and clinical processes. The key objectives include:

- Efficient patient record management, including registration, medical history, and treatment information.
- Facilitating secure and seamless communication between healthcare providers, staff, and patients.
- Assisting healthcare professionals in diagnosis, treatment planning, and decision support.
- Enhancing patient experience through easy access to their health records and appointment scheduling.
- Ensuring compliance with relevant healthcare data security standards (e.g., HIPAA).
- Supporting efficient administrative tasks, such as billing and insurance claims processing.

2. Inclusions:

The healthcare system application will encompass the following features and functionalities:

- Patient registration and record management.
- Appointment scheduling and reminders.
- Clinical data entry and retrieval.
- Communication tools for healthcare providers and patients.
- Reporting and analytics for healthcare decision-making.
- Data security and compliance measures.
- Administrative functions, including billing and insurance management.

3. Exclusions:

The following items are explicitly excluded from the scope of the healthcare system application:

- Direct medical services or medical devices (the application is a management and information tool, not a medical device).
- In-depth clinical decision support, which may require specialized medical software.
- Implementation of hardware components (e.g., medical devices) outside of the software application.

4. Constraints:

- The application will include examples of healthcare data.
- Hardware, infrastructure, and people limitations will be considered during development and deployment.

5. Assumptions:

- It is assumed that the application will be used by healthcare professionals, administrative staff, and patients.
- Access to necessary hardware and infrastructure components will be available as required.

6. Dependencies:

- The successful implementation of the application is dependent on the availability of skilled development and support teams.
- Integration with external systems and databases may be necessary.

7. Risks:

- Risks associated with data security, privacy, and regulatory compliance must be addressed.

8. Acceptance Criteria:

The healthcare system application will be considered successful when it meets the project objectives and delivers the specified features while complying with relevant healthcare regulations.

    - Document Conventions: - we use camelCase naming convention
    - Intended Audience: customer, system maintain developer
- System Architecture
    - Technology Stack:
        - HTML, CSS, JAVASCRIPT, MYSQL, PYTHON, DJANGO
- Data Model
    - Data Entities and attributes:
- Patient:
    - PatientID (Primary Key)
    - FirstName
    - LastName
    - DateOfBirth
    - Gender
    - Contact Information (Address, Phone, Email)
    - Insurance Information
- Medical Provider:
    - ProviderID (Primary Key)
    - FirstName
    - LastName
    - Specialty
    - Contact Information (Address, Phone, Email)
- Appointment:
    - AppointmentID (Primary Key)
    - PatientID (Foreign Key)
    - ProviderID (Foreign Key)
    - AppointmentDateTime
    - Notes
- Medical Record:
    - RecordID (Primary Key)

    - PatientID (Foreign Key)
    - ProviderID (Foreign Key)
    - RecordDateTime
    - Diagnosis
    - Treatment
    - Medications
    - Test Results
    - Follow-up Information
- Prescription:
    - PrescriptionID (Primary Key)
    - RecordID (Foreign Key)
    - Medication Name
    - Dosage
    - Frequency
    - Start Date
    - End Date
- Test Result:
    - TestResultID (Primary Key)
    - RecordID (Foreign Key)
    - Test Name
    - Test Date
    - Test Result Details
- Billing and Claims:
    - BillingID (Primary Key)
    - PatientID (Foreign Key)
    - ProviderID (Foreign Key)
    - Service Date
    - CPT (Current Procedural Terminology) Codes
    - Amount
    - Insurance Information
    - Claim Status
        -
    - Relationships:
        - A Patient can have multiple Appointments.
        - A Medical Provider can have multiple Appointments.
        - An Appointment is associated with one Patient and one Medical Provider.
        - A Patient can have multiple Medical Records.
        - A Medical Provider can create multiple Medical Records.
        - A Medical Record is associated with one Patient and one Medical Provider.
        - A Medical Record can have multiple Prescriptions and Test Results.
        - A Prescription is associated with one Medical Record.
        - A Test Result is associated with one Medical Record.
        - Billing and Claims data are related to Patient and Provider entities.
        -
    - Database Schema:

        - Provide a schema for the database design.
    - Data Security:
        - Explain data encryption, access control, and compliance with healthcare data standards (e.g., HIPAA).

- User Interface Design
    - User Roles:
        - Define user roles (e.g., doctors, nurses, administrators) and their permissions.
    - User Interface Mockups:
        - Include wireframes and design elements.
    - User Workflow:
        - Describe how users will interact with the system.
- Functional Requirements
    - Use Cases:
        - List and describe major use cases (e.g., patient registration, diagnosis entry, report generation).
    - Functional Flows:
        - Provide flowcharts or diagrams for critical functions.
    - Business Logic:
        - Explain algorithms and business rules used in the application.
- Security and Compliance
    - Authentication and Authorization:
        - Describe authentication mechanisms and user access control.
    - Compliance (e.g., HIPAA):
        - Explain how the system ensures compliance with healthcare data security standards.
    - Audit Trail:
        - Detail how system activities are logged for auditing purposes.
- Performance and Scalability
    - Performance Goals:
        - Specify performance objectives such as response times and system load capacity.
    - Scalability Measures:
        - Describe how the application can scale to handle increased user loads.
- Testing and Quality Assurance
    - Test Plans:
        - Describe the testing approach, including unit tests, integration tests, and acceptance tests.
    - Quality Assurance Processes:
        - Explain how quality and compliance checks are enforced.
- Deployment and Configuration
    - System Requirements:
        - List hardware and software requirements.
    - Deployment Instructions:
        - Provide installation and configuration guidelines.
    - Disaster Recovery:
        - Describe backup and recovery procedures.
- Maintenance and Updates

    - Version Control:
        - Explain versioning and source code management.
    - Change Management:
        - Describe how changes are documented, reviewed, and implemented.
    - Documentation Maintenance:
        - Explain how design documentation is updated in parallel with the system.
- References and Appendices
    - Include any external references, standards, and any additional information.

Review and Approval

Reviewers:

- Technical Team:

Backend Engineer: Manpreet
Frontend Engineer: Pang
Backend / AI Engineer: Saurabh
Backend / AIEngineer: Duaa

- Healthcare Experts: (In progress)
- Usability and User Experience (UX) Experts: (In progress)

Approvers:

- Project Manager:

Liya

- Stakeholders and End Users: (In progress)