SOFTWARE REQUIREMENTS SPECIFICATION

For

Hospital Appointment Scheduler

By:

Dharaneswaran V, Dharnesh R, Ganesh Kumar M, Theebeshwar S

|  |  |  |
| --- | --- | --- |
| **TABLE OF CONTENTS** | | |
| **S. No** | **CONTENTS** | |
| **1** | **Introduction** | |
|  | 1.1 | Purpose |
|  | 1.2 | Scope |
|  | 1.3 | Definitions, Acronyms, and Abbreviations |
| **2** | **Overall Description** | |
|  | 2.1 | Product Perspective |
|  | 2.2 | Product Features |
|  | 2.3 | User Classes and Characteristics |
|  | 2.4 | Operating Environment |
|  | 2.5 | Assumptions and Dependencies |
|  | 2.6 | Product Function |
| **3** | **Functional Requirements** | |
|  | 3.1 | Appointment Scheduling |
|  | 3.2 | Appointment Confirmation |
|  | 3.3 | Appointment Cancellation |
|  | 3.4 | Medical Record Access |
|  | 3.5 | Notifications and Reminders |
|  | 3.6 | Security and Access Control |
|  | 3.7 | Emergency SOS Notification (Optional) |
| **4** | **External Interface Requirements** | |
|  | 4.1 | User Interfaces |
|  | 4.2 | Hardware Interfaces |
|  | 4.3 | Software Interfaces |
| **5** | **Non-Functional Requirements** | |

|  |  |  |
| --- | --- | --- |
|  | 5.1 | Performance |
|  | 5.2 | Security |
|  | 5.3 | Reliability |
|  | 5.4 | Scalability |
|  | 5.5 | Usability |
| 6 | **Additional Information** | |

1. Introduction
   1. Purpose

The Hospital Appointment Scheduler's main goal is to give hospital appointment management staff members an effective and user-friendly solution that seamlessly integrates and tracks patients' medical records. The system's objectives are to improve patient experience, lower administrative burden, expedite appointment scheduling, and facilitate effective medical history monitoring.

* 1. Scope

The Patient-Doctor Appointment and Medical Records Management System's functionality and interactions are designed to meet the demands of medical professionals, patients, and administrative personnel in a healthcare setting.Appointment Management.

* + - Medical Records Access and Management
    - Notifications and Reminders
    - Security and Compliance
  1. Definitions, Acronyms, and Abbreviations SRS: Software Requirements Specification GUI: Graphical User Interface

JAVA -> platform independence

SQL-> Structured query Language

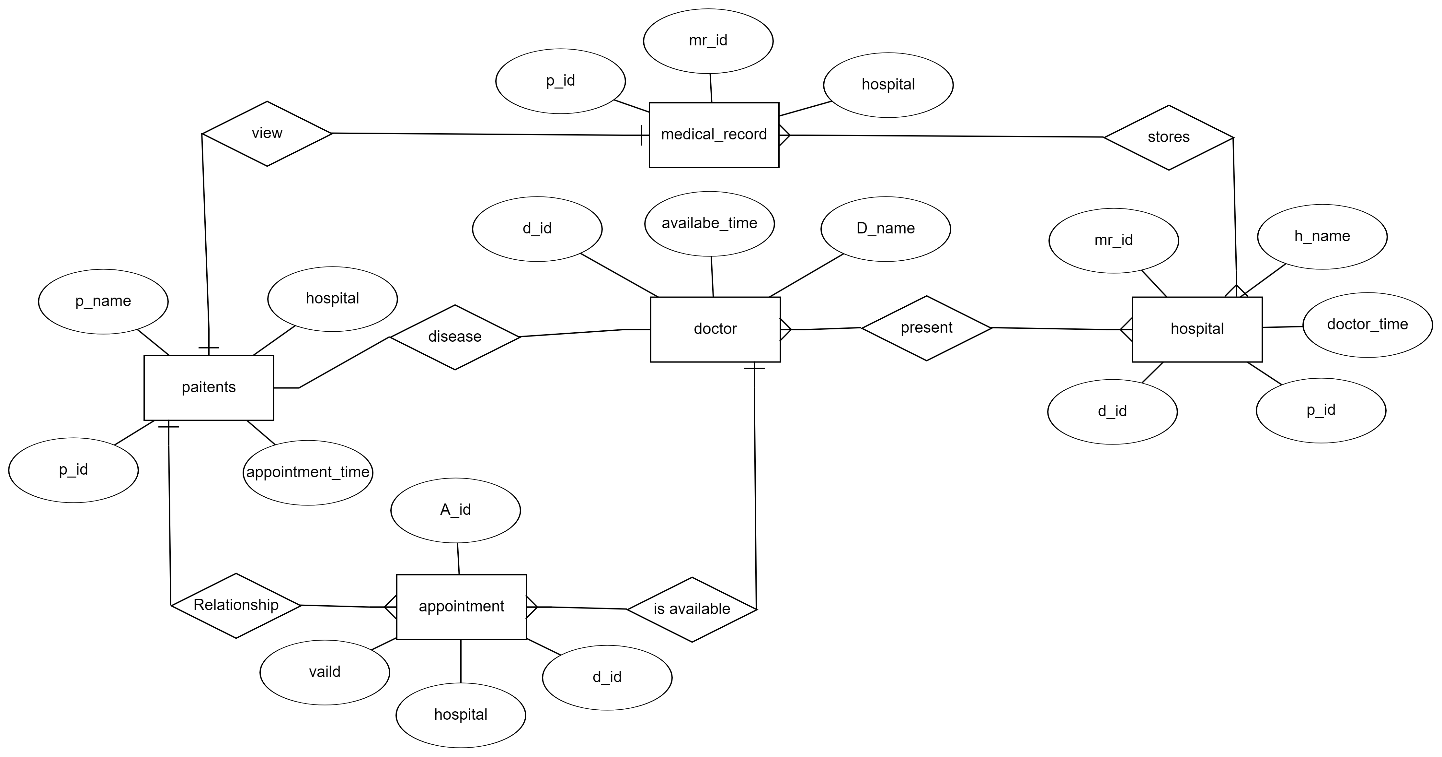
ER-> Entity Relationship

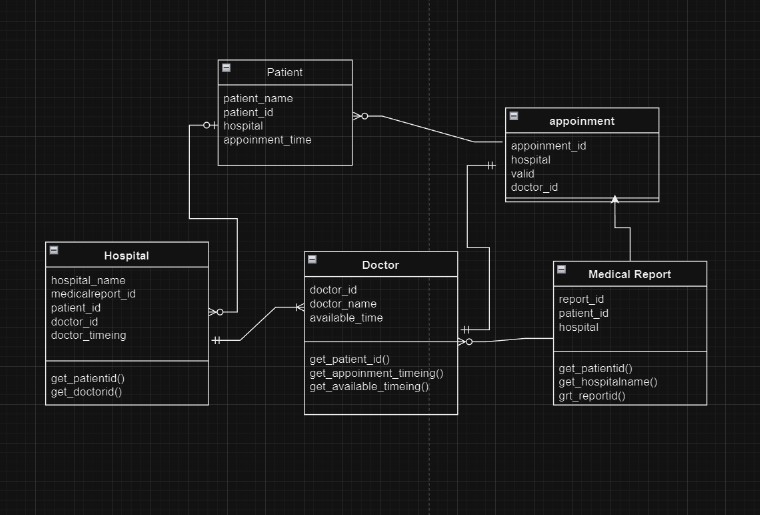
UML -> Unified Modeling Language

IDE-> Integrated Development Environment

SRS-> Software Requirement Specification

1. Overall Description
   1. Product Perspective



* 1. In order to improve and expedite the appointment scheduling procedure inside the hospital's healthcare ecosystem, the Hospital Appointment Scheduler is intended to be a stand-alone, self-contained system. In order to access and update patient information, the standalone system functions independently while integrating smoothly with the hospital's current patient management infrastructure. It contains an external SOS notification feature for hospital emergencies.
  2. Product Features
     1. Patien t Features
        + Account Creation: Patients can create accounts to manage their appointments.
        + Appointment Scheduling: Patients can schedule, reschedule, or cancel appointments.
        + Notification: Patients receive notifications for appointment confirmation, reminders, and updates.
     2. Appointment Management
        + Patients: Can request appointments, specifying preferred dates, times, and reasons.
        + Doctors: Can view and manage their schedules, confirm or reschedule appointments.
     3. Medical Record Access
        + Patients: Can access their medical records securely.
        + Doctors: Can access patient records they are treating, with appropriate permissions.
  3. User Classes and Characteristics
     + Patients: End users who schedule and manage their appointments.
     + Hospital Staff: Can make changes in schedule as per availability of doctor.
  4. Operating Environment

The system will be web-based, accessible through popular web browsers (Chrome, Firefox, Safari) on desktop and mobile devices.

* 1. Assumptions and Dependencies
     1. *Assumptions*

Internet Connectivity:

* + - * The system assumes that users will have a reliable internet connection for accessing the web-based application.

User Device Compatibility:

* + - * It is assumed that users will have devices (desktops or mobile devices) with compatible web browsers (Chrome, Firefox, Safari) to access the system.

Hospital Infrastructure:

* + - * The system assumes the availability and proper functioning of the hospital's

existing patient management infrastructure, which the scheduler will integrate with for accessing and updating patient information.

Security Compliance:

* + - * It is assumed that the hospital's existing patient management system adheres to security and compliance standards, ensuring the secure handling of patient data.

Availability of Java Runtime Environment (JRE):

* + - * For deployment, it is assumed that end-users will have the Java Runtime Environment (JRE) installed on their machines to run the JavaFX application.
    1. *Dependencies*

JavaFX and JDK Compatibility:

* + - * The system is dependent on the compatibility between the JavaFX version used for development and the corresponding version of the Java Development Kit (JDK).

MySQL Database Compatibility:

* + - * The system is dependent on the compatibility between the version of the

MySQL database used for development and the target environment where the system will be deployed.

External Web Browsers:

* + - * The system is dependent on external web browsers (Chrome, Firefox, Safari) for users to access the web-based application.

Hospital Emergency SOS Notification System:

* + - * If the system includes a feature for hospital emergency SOS notification, it depends on the proper integration and functioning of this external system.

JDBC Driver for MySQL:

* + - * The system depends on the availability and proper configuration of the JDBC driver for MySQL to facilitate communication between the Java application and the MySQL database.

Proper Configuration of IDE:

* + - * During development, the system depends on the proper configuration of the Integrated Development Environment (IDE) to support JavaFX and FXML for UI design.

Hospital's Patient Management System:

* + - * The system is dependent on the hospital's patient management system for retrieving and updating patient information. Any changes or updates to the patient management system may impact the functionality of the appointment scheduler.
  1. Product Function

The primary functions of the Hospital Appointment Scheduler revolve around enhancing the efficiency and convenience of managing appointments and medical records within a healthcare environment. Key functionalities include:

Appointment Management:

* + - *Patient-Side Functionality:* Enables patients to create accounts, schedule, reschedule, or cancel appointments based on their preferences.
    - *Doctor-Side Functionality:* Allows doctors to view and manage their schedules, confirming or rescheduling appointments as needed.

Medical Record Access and Management:

* + - *Patient Access:* Provides a secure platform for patients to access their medical records, promoting transparency and involvement in their healthcare.
    - *Doctor Access:* Allows doctors to securely access and manage the medical records of patients they are treating, with appropriate permissions.

Notifications and Reminders:

* + - Sends notifications to patients for appointment confirmation, reminders, and updates, improving communication and reducing the likelihood of missed

appointments.

Security and Compliance:

* + - Ensures the security of patient data by implementing access controls and

encryption, maintaining compliance with healthcare regulations and standards.

Emergency SOS Notification Feature:

* + - Provides an external feature for emergency SOS notifications, enhancing the hospital's ability to respond swiftly to critical situations.

1. Functional Requirements
   1. *Appointment Scheduling*
      1. *Description*

Patients can schedule appointments based on available time slots for specific doctors or departments.

* + 1. *Inputs*
       - Patient details (name, contact information, etc.)
       - Preferred date and time for the appointment
       - Selection of a specific doctor or department
    2. *Outputs*
       - Confirmation message indicating the successful scheduling of the appointment
       - Update to the patient's calendar within the system
  1. *Appointment Confirmation*
     1. *Description*

Hospital staff can confirm or reschedule appointments based on resource availability.

* + 1. *Inputs*
       - Appointment details (patient details, date, time)
       - Resource availability information
    2. *Outputs*
       - Confirmation message for the scheduled appointment
       - Updated schedule reflecting the confirmed or rescheduled appointment
  1. *Appointment Cancellation*
     1. *Description*

Patients can cancel their scheduled appointments if needed.

* + 1. *Inputs*
       - Patient identification details
       - Appointment details (date, time)
    2. *Outputs*
       - Confirmation message for the canceled appointment
       - Updated schedule reflecting the canceled appointment
  1. *Medical Record Access*
     1. *Description*

Patients and authorized doctors can securely access and manage medical records.

* + 1. *Inputs*
       - Patient login credentials for access
       - Doctor login credentials with appropriate permissions
    2. *Outputs*
       - Secure access to patient medical records
       - Ability for doctors to view, update, and manage patient records
  1. *Notifications and Reminders*
     1. *Description*

The system sends notifications to patients for appointment confirmation, reminders, and updates.

* + 1. *Inputs*
       - Scheduled appointment details
       - Patient contact information
    2. *Outputs*
       - Appointment confirmation notification
       - Appointment reminder notification
       - Updates or changes notification
  1. *Security and Access Control*
     1. *Description*

The system ensures the security of patient data and restricts access based on user roles.

* + 1. *Inputs*
       - User login credentials
       - Role-based permissions
    2. *Outputs*
       - Secure storage and retrieval of patient data
       - Access restricted based on user roles
  1. *Emergency SOS Notification (Optional)*
     1. *Description*

The system interfaces with an external emergency SOS notification system for critical situations.

* + 1. *Inputs*
       - Emergency trigger or alert
    2. *Outputs*
       - Swift notification to hospital staff for emergency response

1. External Interface Requirements

User interfaces, hardware interfaces, and software interfaces are just a few examples of the external interfaces that define how the Hospital Appointment Scheduler communicates with the outside world. The particular specifications pertaining to external interfaces are described below:

* 1. User Interfaces

The user interfaces are designed to be intuitive, user-friendly, and tailored to the needs of both patients and hospital staff.

* + 1. *Patient Interface:*
       - The patient interface should provide functionality for account creation, appointment scheduling, access to medical records, and notifications.
       - It must be accessible through standard web browsers such as Chrome, Firefox, and Safari on both desktop and mobile devices.
       - The GUI should be designed with a focus on simplicity and ease of navigation for patients.
    2. *Hospital Staff Interface:*
       - The staff interface is tailored for hospital personnel responsible for managing the appointments.
       - It should provide functionalities for confirming or rescheduling appointments, accessing schedules, and managing patient records.
       - The GUI should be designed to streamline administrative tasks, ensuring efficient workflow for hospital staff.
  1. Hardware Interfaces

The system will run on standard web browsers and require internet connectivity.

* 1. Software Interfaces

The system will interface with the hospital's patient management system to retrieve and update patient information.

1. Non-functional Requirements
   1. *Performance*
      1. *Response Time:*
         * Ensure the system responds to user actions within 3 seconds for a smooth user experience.
      2. *Concurrent User Handling:*
         * Make sure the system can handle multiple users simultaneously without a significant decrease in performance.
   2. *Security*
      1. *Data Encryption:*
         * Safeguard patient information by encrypting data during storage and transmission.
      2. *User Authentication:*
         * Only allow authorized personnel to access the system through a secure user authentication process.
      3. *Role-Based Access Control:*
         * Control access to system features based on user roles within the hospital.
   3. *Reliability*
      1. *System Uptime:*
         * Aim for high availability with minimal downtime to ensure continuous access.
      2. *Data Integrity:*
         * Implement measures to maintain the accuracy and reliability of patient data.
   4. *Scalability*
      1. *User and Data Scalability:*
         * Design the system to grow seamlessly with an increasing number of users and data.
      2. *Performance Scalability:*
         * Ensure the system's performance scales proportionally with growth.
   5. *Usability*
      1. *User Training:*
         * Design the system to be intuitive, reducing the need for extensive user training.
      2. *Accessibility:*
         * Ensure the system is accessible to users with disabilities, following accessibility standards.
2. *Other Requirements*
   1. *Regulatory Compliance:*
      * The system must adhere to healthcare regulations and standards, ensuring compliance with laws governing patient data protection and electronic health records.
   2. *Backup and Recovery:*
      * Implement regular data backups and establish a robust recovery plan to minimize data loss and system downtime in case of unexpected events.
   3. *System Maintenance:*
      * Provide a mechanism for routine system maintenance without disrupting critical services. Notify users in advance of planned maintenance.

*Additional Information:*

Software Requirements:

Java Development Kit (JDK):

* + - * Version: Java 8 or later.
      * Download: Oracle JDK or OpenJDK. JavaFX SDK:
      * Version: Corresponding to your JDK version.
      * Download: Included with Oracle JDK or available separately. Integrated Development Environment (IDE):
      * Use an IDE supporting JavaFX, e.g., IntelliJ IDEA, Eclipse, or NetBeans. MySQL Database:
      * Version: MySQL 5.7 or later.
      * Download: MySQL Community Edition. JDBC Driver for MySQL:
      * Download the JDBC driver from the MySQL website. FXML for UI Design:
      * Ensure IDE supports FXML for UI design. Build Tool (Optional):
      * Consider using Maven or Gradle. Version Control (Optional):
      * Consider using Git.

Hardware Requirements: Processor:

* + - * Multicore processor suitable for development. Memory (RAM):
      * Minimum 8 GB RAM for smooth development. Storage:
      * Disk space for development environment and project files. Graphics Card (Optional):
      * Dedicated graphics card (optional).

Network Requirements:

Internet Connectivity:

* + - * Required for downloads during development. Database Server:
      * Ensure MySQL server accessibility.

Deployment Requirements:

Java Runtime Environment (JRE):

* + - * End-users need a compatible JRE.

MySQL Server:

* + - * End-users should have access to the MySQL server. Operating System:
      * Target machines should run a compatible OS. Security Considerations:

Secure Communication:

* + - * Implement secure communication. Database Security:
      * Set up authentication and authorization for MySQL users. Application Security:
      * Implement secure coding practices.