|  |  |
| --- | --- |
|  | **Healthcare Appointment Management System**  **Technical Design Document** |
| |  |  |  |  | | --- | --- | --- | --- | |  | **Prepared By / Last Updated By** | **Reviewed By** | **Approved By** | | **Name** | Himanshu Singh |  |  | | **Role** | Developer |  |  | | **Signature** |  |  |  | | **Date** | 27/03/25 |  |  | |
|  |

Table of Contents

[1.0 Introduction 3](#_Toc14171023)

[1.1 Purpose of this document 3](#_Toc14171024)

[1.2 Project overview 3](#_Toc14171025)

[2.0 Solution Summary 3](#_Toc14171026)

[2.1 Scope 3](#_Toc14171027)

2.2 Assumptions 4

2.3 Dependencies 4

2.4 Risks 4

3.0 Schematic Diagram 5

4.0 System Design 6

4.1 Proposed Design 6

4.2 Compoment Inventory 8

5.0 Database Design 9

5.1 Data Model 9

5.2 Table Structure 10

6.0 Appendices 13

6.1 Glossary 13

7.0 Terms and Conditions 13

8.0 Change Log 14

9.0 Deadlines 14

# Introduction

## Purpose of this document

The Healthcare Appointment Management System aims to streamline and simplify the process of booking, managing, and tracking appointments for healthcare providers and patients. The system ensures efficient scheduling, reduces manual effort, minimizes errors, and improves patient satisfaction by providing a user-friendly platform for appointment management.

## Project overview

The system is a web-based platform designed for hospitals, clinics, and medical practices to manage appointments seamlessly. It allows patients to book appointments online, view available time slots, and receive notifications for confirmations or cancellations. Healthcare providers can access a centralized dashboard to manage schedules, track patient appointments, and optimize resource allocation. The system integrates features such as patient records, reminders, and analytics to enhance operational efficiency and improve healthcare delivery.

# Solution Summary

## Scope

The design document outlines the architecture, features, and functionality of the Healthcare Appointment Management System. It covers the following:

1. User Roles: Patients, Doctors.

2. Core Features:

- Online appointment booking and cancellation.

- Availability management for healthcare providers.

- Notifications and reminders for appointments.

- Patient profile and medical history management.

3. System Requirements:

- Frontend: A user-friendly interface for patients and staff.

- Backend: A robust database for storing user data, schedules, and logs.

- Security: Data encryption, role-based access control, and compliance with healthcare data regulations (e.g., HIPAA).

4. Integration: Compatibility with hospital management systems and third-party tools like payment gateways.

## Assumptions

1. All users will have access to the internet and compatible devices to use the system.

2. Healthcare providers and administrators will be trained to use the system effectively.

3. The system will comply with regional healthcare data privacy laws.

4. Appointment duration and scheduling rules will be predefined by healthcare providers.

5. Emergency cases and walk-in appointments will be managed separately from the system.

6. The system will support scalability for additional users or features as required in the future.

## Dependencies

- Spring Boot Starter Web

- Spring Boot DevTools

- Spring Data JPA

- Spring Security

- OAuth2 Client

- Oracle SQL Driver

- Java Mail API

- Lombok

- JUnit

- Mockito

## Risks

**1. Data Breach and Privacy Risks** – Unauthorized access to sensitive patient information.

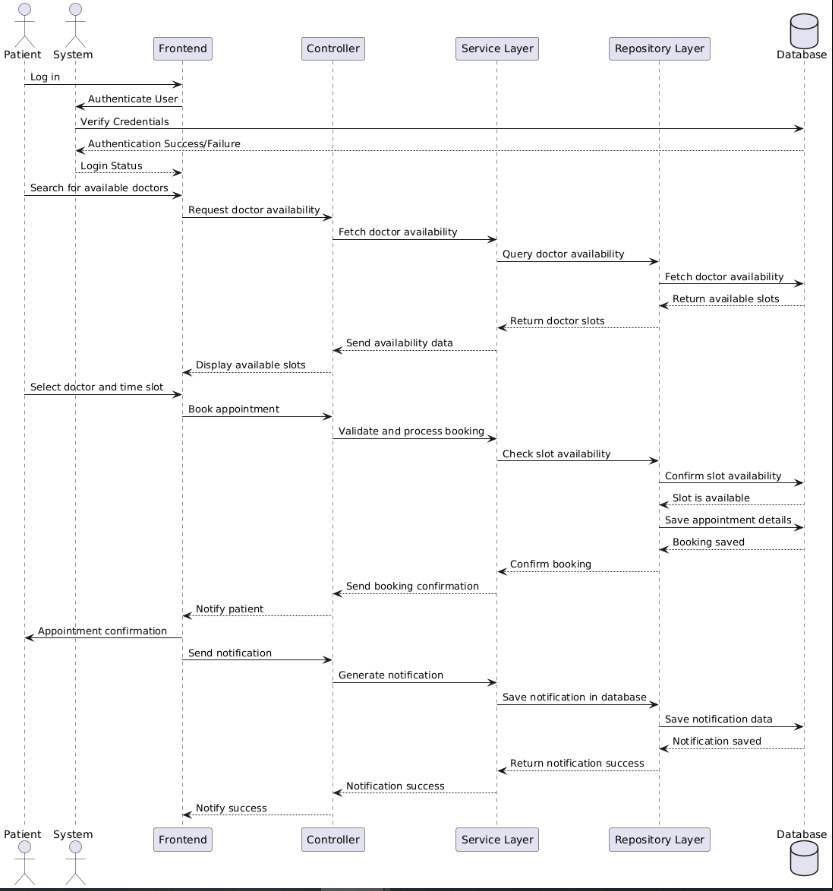
**2. System Downtime** – Service unavailability affecting appointments and user trust.

**3. Scalability Issues** – Poor performance under high user load.

**4. Incorrect Appointment Scheduling** – Errors leading to double bookings or missed slots.

**5. Cybersecurity Threats** – Vulnerabilities to attacks like SQL injection or DDoS.

# Schematic Diagram



# System Design

## Proposed design

The proposed design of the Healthcare Appointment Management System is a modular, scalable, and secure architecture. It involves multiple layers and components to ensure efficient and seamless interaction between users (patients and doctors) and the system. The design will adhere to best practices in security, maintainability, and performance optimization.

**1. Presentation Layer (Frontend):**

- Technologies: Bootstrap CSS, jQuery.

- Provides user interfaces for patients and doctors to interact with the system.

- Consumes RESTful APIs for all functionalities.

**2. Application Layer (Backend):**

- Technology: Java Spring Boot.

- Implements business logic and exposes RESTful APIs.

- Components:

- Controllers: Handle API requests and responses.

- Services: Contain business logic and validation.

- Repositories: Handle database interactions.

- Security: Spring Security for authentication and role-based access.

**3. Data Layer (Database):**

- Technology: Oracle SQL.

- Stores and manages data such as users, appointments, notifications, and prescriptions.

- Indexed tables for optimized queries.

**Core Modules in the System**

1. User Management:

- Handles registration, login, and role-based access control.

- Supports two roles: Patient and Doctor.

- Provides secure authentication and authorization using Spring Security.

- Being developed by Aryan.

2. Doctor Management:

- Manages doctor profiles, including specialization, availability, and experience.

- Allows doctors to define and update availability (recurring or one-time).

- Being developed by Aryan.

3. Patient Management:

- Manages patient profiles, including medical history, gender, and date of birth.

- Tracks appointment history and notifications.

- Being developed by Himanshu.

4. Appointment Management:

- Enables patients to search for available doctors and book appointments.

- Provides a rescheduling and cancellation feature.

- Tracks the appointment status: Scheduled, Completed, Canceled, or Rescheduled.

- Being developed by Himanshu.

5. Notification System:

- Sends appointment confirmations, reminders, and updates to patients and doctors.

- Tracks notification statuses (e.g., Read, Unread).

- Being developed by Souradeep.

6. Prescription Management:

- Generates and stores prescriptions after a consultation.

- Links prescriptions with appointments for traceability.

- Being developed by Devendra.

## Component inventory

**1. Frontend Components:**

- Login/Register Page

- Dashboard for Patients and Doctors

- Appointment Booking and Management UI

- Notification Center

- Profile Management (Patients/Doctors)

**2. Backend Components:**

- Authentication Service:

- Login/registration functionality.

- User Service:

- Manages user roles and profiles.

- Doctor Service:

- Handles doctor availability and profile management.

- Patient Service:

- Manages patient profiles and medical history.

- Appointment Service:

- Handles booking, rescheduling, and cancellation.

- Notification Service:

- Sends notifications for appointment updates and reminders.

- Prescription Service:

- Generates and manages prescriptions for completed appointments.

- Audit Logging:

- Tracks and stores system events for debugging and compliance.

**3. Database Components:**

- Users Table

- Doctors Table

- Patients Table

- Appointment Table

- Notifications Table

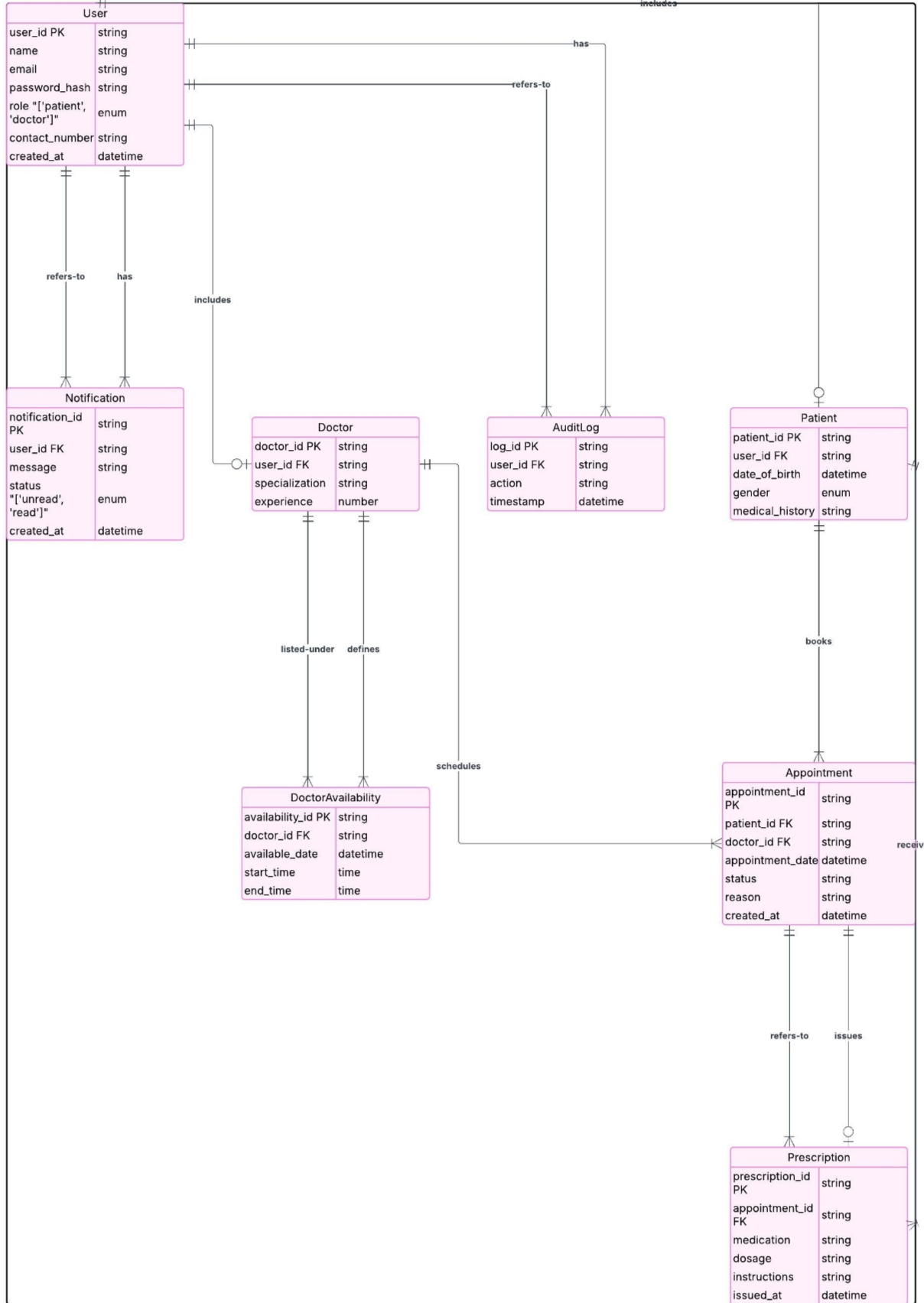
- Prescriptions Table

- Consultation Table

- Doctor Availability Table.

# Database Design

## Data Model



## Tables Structure

**Table: Users**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Nulls** |
| user\_id | NUMBER |  | NO |
| name | VARCHAR | 255 | NO |
| email | VARCHAR | 255 | NO |
| password | VARCHAR | 255 | NO |
| role | ENUM ('PATIENT', 'DOCTOR') |  | NO |
| contact\_number | VARCHAR | 15 | YES |
| created\_at | TIMESTAMP |  | NO |

**Table: Doctors**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Nulls** |
| user\_id | NUMBER |  | NO |
| specialization | VARCHAR | 255 | NO |
| experience | NUMBER |  | NO |

**Table: Patients**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Nulls** |
| user\_id | NUMBER |  | NO |
| date\_of\_birth | DATE |  | YES |
| gender | ENUM ('MALE', 'FEMALE', 'OTHER') |  | YES |
| medical\_history | CLOB |  | YES |

**Table: Appointments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Nulls** |
| appointment\_id | NUMBER |  | NO |
| patient\_id | NUMBER |  | NO |
| doctor\_id | NUMBER |  | NO |
| appointment\_date | TIMESTAMP |  | NO |
| status | ENUM ('SCHEDULED', 'COMPLETED', 'CANCELED', 'RESCHEDULED') |  | NO |
| reason | CLOB |  | YES |
| created\_at | TIMESTAMP |  | NO |

**Table: Doctor Availability**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Nulls** |
| availability\_id | NUMBER |  | NO |
| doctor\_id | NUMBER |  | NO |
| available\_date | DATE |  | NO |
| start\_time | TIME |  | NO |
| end\_time | TIME |  | NO |
| recurring | ENUM ('NONE', 'DAILY', 'WEEKLY') |  | NO |

**Table: Notifications**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Nulls** |
| notification\_id | NUMBER |  | NO |
| user\_id | NUMBER |  | NO |
| message | CLOB |  | NO |
| status | ENUM ('UNREAD', 'READ') |  | NO |
| created\_at | TIMESTAMP |  | NO |

**Table: Prescriptions**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Nulls** |
| prescription\_id | NUMBER |  | NO |
| appointment\_id | NUMBER |  | NO |
| medication | CLOB |  | NO |
| dosage | CLOB |  | NO |
| instructions | CLOB |  | YES |
| issued\_at | TIMESTAMP |  | NO |

**Table: Consultation**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Length** | **Nulls** |
| consultation\_id | NUMBER |  | NO |
| appointment\_id | NUMBER |  | NO |
| doctor\_id | NUMBER |  | NO |
| patient\_id | NUMBER |  | NO |
| patient\_description | CLOB |  | YES |
| diagnosis | CLOB |  |  |
| treatment\_plan | CLOB |  |  |
| issued\_at | TIMESTAMP |  | NO |

# Appendices

## Glossary

|  |  |
| --- | --- |
| **Acronyms** | **Definitions** |
| API | Application Programming Interface: A set of protocols and tools for building software applications. |
| ENUM | Enumeration: A data type consisting of a set of named values (e.g., roles like PATIENT, DOCTOR). |
| TIMESTAMP | A data type used to store date and time values. |
| AUDIT LOG | A record of changes or actions taken in the system for security and traceability. |
| ENTITY | A real-world object or concept represented in the database (e.g., User, Appointment). |
| SQL | Structured Query Language: A language used for managing and querying relational databases. |

# Terms & Conditions

***Disclaimer: Please do not circulate or distribute this document outside of Cognizant Network, We have a Zero Tolerance Policy. Kindly adhere to 100% Compliance at all times.***

# Change Log

*Please note that this table needs to be maintained even if a Configuration Management tool is used.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version Number | Changes made | | | |
| V<n.n> | *<If the change details are not explicitly documented in the table below, reference should be provided here>* | | | |
| Page no | Changed by | Effective date | Changes effected |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Deadlines

|  |  |
| --- | --- |
| **DATE** | **TASK** |
| March 25 | Frontend Development |
| April 22 | Backend Development |
| April 25 | Integration |
| May 1 | Testing |
| May 2 | Final Documentation |