## Project Development Document: Doctor Appointment Booking System

## 1. Introduction

1.1 Overview

The Doctor Appointment Booking System is a software solution that allows users to efficiently book appointments with doctors, manage patient records, and check doctor availability and user ratings. The system supports three roles: Admin, Doctor, and User (Patients). Each role is designed to simplify interaction and streamline healthcare service management.

1.2 Problem Statement

Traditional appointment booking systems are often time-consuming and inefficient. This project aims to develop an intuitive, easy-to-use system where users can:

* Find doctors based on location and specialization.
* Book and manage appointments.
* Maintain and view patient records.
* Enable doctors to manage their schedules effectively.

1.3 Objectives

* Simplify the appointment booking process.
* Provide real-time doctor availability and user ratings.
* Ensure secure management of patient and doctor data.

---

2. Project Scope

The system is designed for healthcare clinics, hospitals, or standalone doctors looking to digitize their services. It includes the following functionalities:

1. \*\*Admin:\*\*
   * Manage doctor details.
   * Manage user details.
   * Update location information.
2. \*\*Doctor:\*\*
   * Manage appointments (create, edit, delete, view).
   * View and manage patient records.
3. \*\*User:\*\*
   * Search for doctors based on specialization and location.
   * Book appointments online.
   * View appointment history.

---

3. Technologies Used

Current Technologies

1. \*\*Programming Language:\*\* Java (JDK 11+)
2. \*\*Integrated Development Environment (IDE):\*\* Eclipse
3. \*\*Database:\*\* MySQL
4. \*\*Backend Frameworks:\*\*
   * Java Servlets
   * JDBC (Java Database Connectivity)
5. \*\*Frontend (Optional for GUI):\*\*
   * Java Swing (for desktop interface)
6. \*\*Version Control:\*\* Git/GitHub

### Future Technologies

To further enhance and scale the project, the following technologies can be integrated:

1. \*\*Frontend Frameworks:\*\*
   * HTML, CSS, JavaScript for a web-based UI.
   * React.js or Angular for a dynamic, modern interface.
2. \*\*Backend Enhancements:\*\*
   * Spring Boot Framework for robust APIs.
   * Hibernate for advanced ORM (Object-Relational Mapping).
3. \*\*Cloud Integration:\*\*
   * AWS or Azure for hosting.
   * Firebase for real-time database functionality.
4. \*\*Mobile App Development:\*\*
   * Android Studio (Java/Kotlin) for Android apps.
   * Swift for iOS apps.
5. \*\*Security Features:\*\*
   * OAuth2.0 for authentication.
   * SSL Certificates for secure connections.
6. \*\*Notification Systems:\*\*
   * SMS or Email APIs for appointment reminders.

---

4. System Requirements

Hardware Requirements

* + Processor: Intel i3 or above
  + RAM: 4GB or above
  + Hard Disk: Minimum 10GB free space

Software Requirements

* + Operating System: Windows/Linux/Mac
  + JDK 11 or higher
  + MySQL Server
  + Eclipse IDE
  + Git (optional)

---

5. System Design

Flowchart

The following flowchart represents the overall workflow of the system:

1. User selects their role (Admin, Doctor, User).
2. Based on the role:
   * Admin can log in and manage doctors, users, and locations.
   * Doctors can log in and manage appointments and patient records.
   * Users can sign up/log in to book appointments.

Entity-Relationship (ER) Diagram

* + \*\*Tables:\*\*
  + Users: Stores user details.
  + Doctors: Stores doctor details.
  + Appointments: Stores appointment details.
  + Locations: Stores geographical data for doctors.

---

6. Modules and Functionalities

Admin Module

* + Login/Logout functionality.
  + Manage doctor details (add, update, delete).
  + Manage user details (view, delete).
  + Manage locations (add, update, delete).

Doctor Module

* + Create and manage appointments.
  + View appointment schedules.
  + Manage patient details.

User Module

* + Create an account and log in.
  + Search for doctors based on specialization and location.- Book appointments.
  + View appointment history.

---

7. Implementation

Key Code Snippets

1. \*\*Admin Login Example:\*\*

java public boolean adminLogin(String email, String password) {

String query = "SELECT \* FROM admin WHERE email=? AND password=?"; try (PreparedStatement stmt = connection.prepareStatement(query)) { stmt.setString(1, email); stmt.setString(2, password); ResultSet rs = stmt.executeQuery(); return rs.next(); } catch (SQLException e) {

e.printStackTrace();

}

return false;

}

```

2. Appointment Booking Example:\*\*

java public boolean bookAppointment(int userId, int doctorId, String date, String time) {

String query = "INSERT INTO appointments (user\_id, doctor\_id, date, time) VALUES (?, ?, ?, ?)"; try (PreparedStatement stmt = connection.prepareStatement(query)) { stmt.setInt(1, userId); stmt.setInt(2, doctorId); stmt.setString(3, date); stmt.setString(4, time); return stmt.executeUpdate() > 0; } catch (SQLException e) {

e.printStackTrace();

}

return false;

}

```

---

8. Testing

Test Cases

| Test Case ID | Module | Description | Expected Outcome | Status |

| ------------ | ------------ | --------------------------- | ----------------- | ------ |

| TC01 | Admin Login | Admin enters valid details | Successful Login | Pass |

| TC02 | Doctor Login | Doctor enters invalid email | Login Fails | Pass |

| TC03 | Appointment | User books appointment | Appointment Saved | Pass |

---

9. Future Enhancements

1. Mobile app for better accessibility.
2. AI-based doctor recommendations.
3. Payment gateway integration for online payments.
4. Telemedicine feature for virtual consultations.

---

10. Conclusion

The Doctor Appointment Booking System is designed to simplify appointment scheduling, enhance patient care, and provide seamless management of healthcare services. Future enhancements aim to expand functionality and accessibility.

---

11. References

1. Java Documentation: https://docs.oracle.com/javase/8/docs/
2. MySQL Documentation: [https://dev.mysql.com/doc/](https://dev.mysql.com/doc/)
3. Eclipse IDE: [https://www.eclipse.org/](https://www.eclipse.org/)