APPOINTMENT MANAGEMENT AND BOOKING SYSTEM

A MINI-PROJECT REPORT

Submitted by

TUFAIL AHAMED B 241901119

UBENDRA L 241901120

in partial fulfillment of the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

An Autonomous Institute

CHENNAI

NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project **"APPOINTMENT MANAGEMENT AND BOOKING SYSTEM"** is the Bonafide work of **"TUFAIL AHAMED B, UBENDRA L"** who carried out the project work under my supervision.

SIGNATURE

Ms. R. RUPMALA

ASSISTANT PROFESSOR

Dept. of Computer Science and Engineering (Cyber Security),

Rajalakshmi Engineering College

Chennai

This mini project report is submitted for the viva voce examination to be held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We hereby declare that the mini project report "**APPOINTMENT MANAGEMENT AND BOOKING SYSTEM**", submitted as part of the curriculum requirements for the Bachelor of Engineering (B.E) degree affiliated to Anna University, is a bonafide work carried out by us under the supervision of Ms. R. Rupmala, Assistant Professor, Department of Computer Science Engineering and Cyber Security, Rajalakshmi Engineering College, Chennai.

This submission represents our ideas in our own words, and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources.

We also declare that we have adhered to the ethics of academic honesty and integrity and have not misrepresented or fabricated any data, idea, fact, or source in our submission. We understand that any violation of the above will be grounds for disciplinary action by the institute and/or the University and may also evoke penal action from the sources which have not been properly cited or from whom proper permission has not been obtained. This report has not previously formed the basis for the award of any degree, diploma, or similar title of any other University.

Rajalakshmi Engineering College, **TUFAIL AHMED B**

Chennai **UBENDRA L**

November 2025

ABSTRACT

In our state, maintaining law and order plays a major role. Even though there are various national-level crime tracking and reporting systems, the local departments often lack an efficient and hassle-free application to manage crime-related data. To address this drawback in the local law enforcement system, our team developed a database system to help police departments and related agencies maintain and organize crime records efficiently.

The main objective of this project is to record, monitor, and analyze crime incidents to assist in investigation and prevention. This system helps in maintaining details about reported crimes, suspects, victims, and ongoing investigations within a local jurisdiction. By enabling quick access to reliable information and better coordination among officers, this system allows the department to enhance its efficiency and improve public safety.

Keywords: JavaFX, MySQL, JDBC, Vehicle Management,
Data Security, Authentication, CRUD Operations, Maintenance
Tracking, User Interface Design

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved and honorable chairman **MR. S. MEGANATHAN** and the chairperson DR. **M. THANGAM MEGANATHAN** for their timely support and encouragement.

We are greatly indebted to our respected and honorable principal **Dr. S.N. MURUGESAN** for his able support and guidance.

No words of gratitude will suffice for the unquestioning support extended to us by our Head of The Department Mr. **Benedict Jayaprakash Nicholas** for being ever supporting force during our project work

We also extend our sincere and hearty thanks to our internal guide **Ms. R. RUPMALA,** for her valuable guidance and motivation during the completion of this project.

Our sincere thanks to our family members, friends and other staff members of computer science engineering.

TUFAIL AHMED B

UBENDRA L

TABLE OF CONTENTS

LIST OF FIGURES

Abbreviation	Full Term
ABS	**Appointment Booking System**
ABC	**Appointment Booking Calendar**
CS	**Customer Service**
CRM	**Customer Relationship Management**
ETA	**Estimated Time of Arrival**
TAT	**Turnaround Time**
SOP	**Standard Operating Procedure**
SLA	**Service Level Agreement**
UI	**User Interface**
UX	**User Experience**
DBMS	**Database Management System**
SQL	**Structured Query Language**
API	**Application Programming Interface**
OTP	**One-Time Password**

Abbreviation	Full Term
SMS	**Short Message Service**
EMS	**Email Management System**
ADM	**Admin Dashboard Module**
RMS	**Resource Management System**
HMS	**Human Management System (for staff scheduling)**
NTF	**Notification Framework**

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

An Appointment Management System (AMS) is a digital solution designed to streamline the scheduling, tracking, and management of appointments between service providers (such as doctors, salons, consultants, or businesses) and clients. It replaces traditional manual methods like phone calls, paper logs, or spreadsheets with an automated, user-friendly platform that enhances efficiency, reduces errors, and improves customer satisfaction.

1.2 SCOPE OF THE WORK

The Appointment Management System (AMS) will help service providers, staff, and clients efficiently schedule, track, and manage appointments in real time. It aims to provide quick and seamless accessibility to booking slots, staff availability, and service updates. This system enhances coordination between departments, reduces no-shows through automated reminders, and ensures appointment data is securely managed and readily available for analysis. By enabling 24/7 self-service booking and delivering actionable insights, AMS improves operational efficiency, client satisfaction, and decision-making for businesses across healthcare, wellness, and professional services in India.

1.3 PROBLEM STATEMENT

Businesses in India rely on manual scheduling via phone calls, WhatsApp, or paper logs, leading to double-bookings, no-shows (20–30% loss), and administrative overload. Clients face long wait times, unclear availability, and missed reminders. Staff struggle with uncoordinated calendars, while managers lack real-time insights into bookings, revenue, or performance. Existing tools are either costly enterprise solutions or fragmented apps lacking local payment (Razorpay/UPI) and communication (SMS/WhatsApp) integration. This results in inefficient operations, poor client experience, and lost revenue. A unified, affordable, India-centric AMS is needed to automate scheduling, ensure accuracy, and drive data-driven decisions.

1.4 AIM AND OBJECTIVES OF THE PROJECT

To develop a secure, cloud-based Appointment Management System (AMS) that automates scheduling, reduces no-shows, and enhances operational efficiency for Indian businesses, enabling 24/7 self-service booking and real-time coordination between clients, staff, and administrators.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 HARDWARE SPECIFICATIONS

Processor	:	Intel Pentium
Memory Size	:	1GB (Minimum)
HDD	:	20MB (Minimum)

2.2 SOFTWARE SPECIFICATIONS

Operating System	:	MacOS, Linux, Windows
Front – End	:	Java FX
Back - End	:	Java, MYSQL
Language	:	SQL, Java (FX for UI)

CHAPTER 3

MODULE DESCRIPTION

This module handles all user authentication and account creation processes for individuals such as customers, service staff, and administrators. It ensures secure access to appointment schedules, booking history, and system features, enabling authorized users to manage and track appointments efficiently.

- **User Management Module** Manages registration, OTP-based authentication (email/phone), role-based access (Client, Staff, Admin, Super Admin), profile updates, password reset, and secure session handling via JWT.
- **Service & Staff Management Module** Enables admins to create/edit services (name, duration, price, max customers, description), onboard staff (first/last name, role), assign services, set weekly availability (Sun–Sat), holidays, and buffer times.
- **Booking Engine Module** Powers real-time calendar, slot selection (e.g., 09:00–11:00, 12:00–2:00), company/service/day filtering, client self-booking with name/email/phone, and waitlist for full slots.
- **Active Services & Bookings View Module** Displays active services (Hairstyle, Massage) with provider details and last booking timestamp; shows bookings table (ID, Customer, Service, Date, Slot) with "No content in table" placeholder when empty.
- **Edit Service Module** Lists services for editing with Delete/Disable/Edit buttons; supports updating provider, max customers, availability days, and description.
- **Notification Module** Triggers automated confirmations, 24h/1h reminders, cancellation/rescheduling alerts via Email, SMS (MSG91/Twilio), and WhatsApp Business API.

- **Payment Module** (Optional) Integrates Razorpay/UPI for deposits/full payments at booking, generates PDF invoices, and handles admin-approved refunds.
- **Admin Dashboard & Analytics Module** Provides real-time insights (total bookings, no-shows, revenue), staff performance, CSV/PDF export, and cancellation policy enforcement.
- **Security & Compliance Module** Ensures HTTPS, data encryption, audit logs, reCAPTCHA, and compliance with IT Act (India) & GDPR.

CHAPTER 4

SAMPLE CODING

```
package customer;

import java.security.Provider.Service;

import javafx.collections.FXCollections;
import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import javafx.scene.text.Font;
import javafx.scene.text.FontWeight;

public class AppointmentBooking {

    private ComboBox<String> companyCombo;
    private ComboBox<String> serviceCombo;
    private ComboBox<String> dayCombo;
    private FlowPane slotsPane;
    private VBox slotsSection, formVBox, serviceDetailsBox, companyDetailsBox;
    private Button submitBtn;
    private String selectedSlotTime;
    private AppointmentDAO appointmentDAO = new AppointmentDAO();

    // Service fields
    Label lblServiceName, lblProvider, lblCustomers, lblDescription;

    // Company fields
    Label lblOwner, lblCompany, lblBusinessType, lblPhone,
          lblEmail, lblWebsite, lblWorkTime, lblOffDays, lblAddress;

    public static Scene createScene() {
        AppointmentBooking app = new AppointmentBooking();

        BorderPane root = new BorderPane();
        root.setPadding(new Insets(20));
        root.setStyle("-fx-background-color:#f9fafb;");

        VBox header = new VBox(8);
        header.setAlignment(Pos.CENTER);

        Label title = new Label("Book Your Appointment");
        title.setFont(Font.font("Inter", FontWeight.BOLD, 26));
```

```
Label sub = new Label("Select company → service → day → slot → fill details");

header.getChildren().addAll(title, sub);
root.setTop(header);

VBox mainContent = new VBox(25);
mainContent.setPadding(new Insets(20));

// Company selection
VBox companySelectBox = app.createCompanySelectBox();
mainContent.getChildren().add(companySelectBox);

// Service selection
VBox serviceSelectBox = app.createServiceSelectSection();
serviceSelectBox.setVisible(true);
mainContent.getChildren().add(serviceSelectBox);

app.serviceDetailsBox = app.createServiceDetailsBox();
app.serviceDetailsBox.setVisible(false);
mainContent.getChildren().add(app.serviceDetailsBox);

app.slotsSection = app.createSlotSection();
app.slotsSection.setVisible(false);
mainContent.getChildren().add(app.slotsSection);

app.formVBox = app.createFormSection();
app.formVBox.setVisible(false);
mainContent.getChildren().add(app.formVBox);

app.companyDetailsBox = app.createCompanyDetailsBox();
app.companyDetailsBox.setVisible(false);
mainContent.getChildren().add(app.companyDetailsBox);

ScrollPane scroll = new ScrollPane(mainContent);
scroll.setFitToWidth(true);

root.setCenter(scroll);

Scene scene = new Scene(root, 900, 950);

app.loadDummyData();
```

```
    return scene;
}

// ---------------- Company Section ------------------

private VBox createCompanySelectBox() {
    VBox box = new VBox(10);

    companyCombo = new ComboBox<>();
    companyCombo.setPromptText("Select Company");
    companyCombo.setPrefWidth(Double.MAX_VALUE);
    companyCombo.setOnAction(e -> onCompanySelected());

    box.getChildren().addAll(new Label("Company:"), companyCombo);
    return box;
}

private void onCompanySelected() {
    String company = companyCombo.getValue();

    // Show service selection
    serviceCombo.setDisable(false);
    serviceCombo.setValue(null);

    serviceCombo.getParent().setVisible(true);

    System.out.println("Company Selected: " + company);
    loadServicesForCompany(company);
}

// ---------------- Service Section ------------------

private VBox createServiceSelectSection() {
    VBox box = new VBox(10);

    serviceCombo = new ComboBox<>();
    serviceCombo.setPromptText("Select Service");
    serviceCombo.setPrefWidth(Double.MAX_VALUE);
    serviceCombo.setDisable(true);
    serviceCombo.setOnAction(e -> onServiceSelected());

    dayCombo = new ComboBox<>();
```

```
        dayCombo.setPromptText("Select Day");
        dayCombo.setPrefWidth(Double.MAX_VALUE);
        dayCombo.setOnAction(e -> onDaySelected());

        box.getChildren().addAll(new Label("Service:"), serviceCombo, new Label("Day:"), dayCombo);
        return box;
    }

    private void onServiceSelected() {
    String service = serviceCombo.getValue();

    dayCombo.setValue(null);
    dayCombo.setDisable(false);
    slotsSection.setVisible(false);
    formVBox.setVisible(false);

    Service serviceDetails = appointmentDAO.getServiceDetails(service);
    if (serviceDetails != null) {
        lblServiceName.setText("Service: " + serviceDetails.getServiceName());
        lblProvider.setText("Provider: " + serviceDetails.getProviderName());
        lblCustomers.setText("Max Customers: " + serviceDetails.getMaxCustomers());
        lblDescription.setText("Description: " + serviceDetails.getDescription());
    }

    serviceDetailsBox.setVisible(true);

    ObservableList<String> availableDays = FXCollections.observableArrayList();
    String[] allPossibleDays = {"Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"}; // Use all days f

    for (String day : allPossibleDays) {
        // A day is "available" if the service offers at least one slot
        // AND that slot is not yet fully booked.
        for (String slot : appointmentDAO.getSlots(service, day)) {
            if (appointmentDAO.isSlotAvailable(service, day, slot)) {
                availableDays.add(day);
                break; // Found one available slot, so the day is available, move to the next day
            }
        }
    }

    dayCombo.setItems(availableDays);
```

```
    if (availableDays.isEmpty()) {
        dayCombo.setPromptText("No available days for this service");
        dayCombo.setDisable(true);
    }
}


    // ---------------- Service Details ------------------

    private VBox createServiceDetailsBox() {
        VBox box = new VBox(6);
        box.setPadding(new Insets(12));
        box.setStyle("-fx-border-color:#ccc; -fx-padding:10; -fx-background-color:white;");

        lblServiceName = new Label("Service: ");
        lblProvider = new Label("Provider: ");
        lblCustomers = new Label("Max Customers: ");
        lblDescription = new Label("Description: ");

        box.getChildren().addAll(
            new Label("Service Details:"),
            lblServiceName, lblProvider, lblCustomers, lblDescription
        );
        return box;
    }

    // ---------------- Slots ------------------

    private VBox createSlotSection() {
        VBox box = new VBox(10);
        Label label = new Label("Available Slots:");

        slotsPane = new FlowPane(8,8);
        slotsPane.setPrefWrapLength(400);

        box.getChildren().addAll(label, slotsPane);
        return box;
    }

    private void onDaySelected() {
    slotsPane.getChildren().clear();
    formVBox.setVisible(false);
```

CHAPTER 5

SCREEN SHOTS

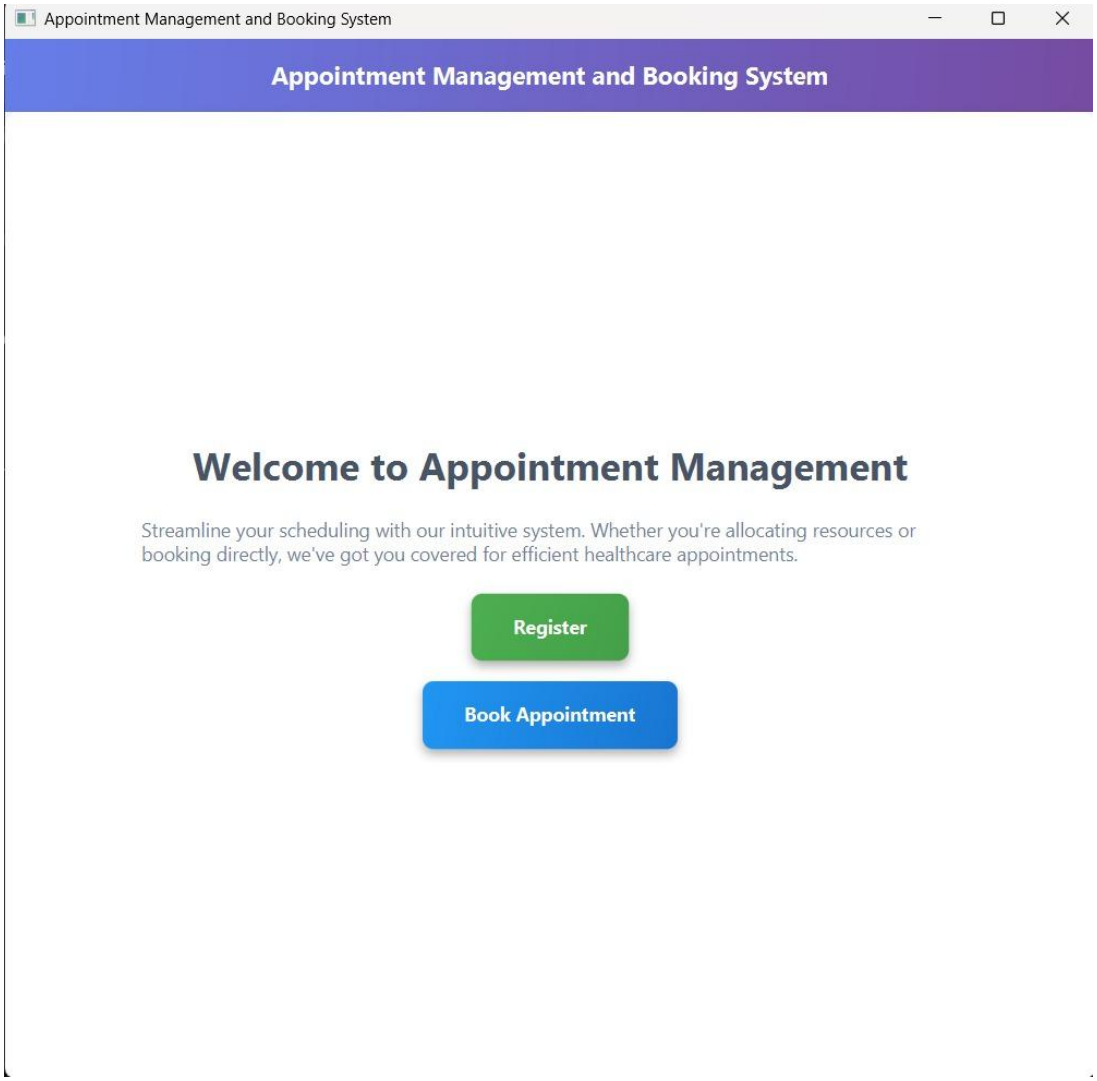

Fig 5.1 Welcome Page

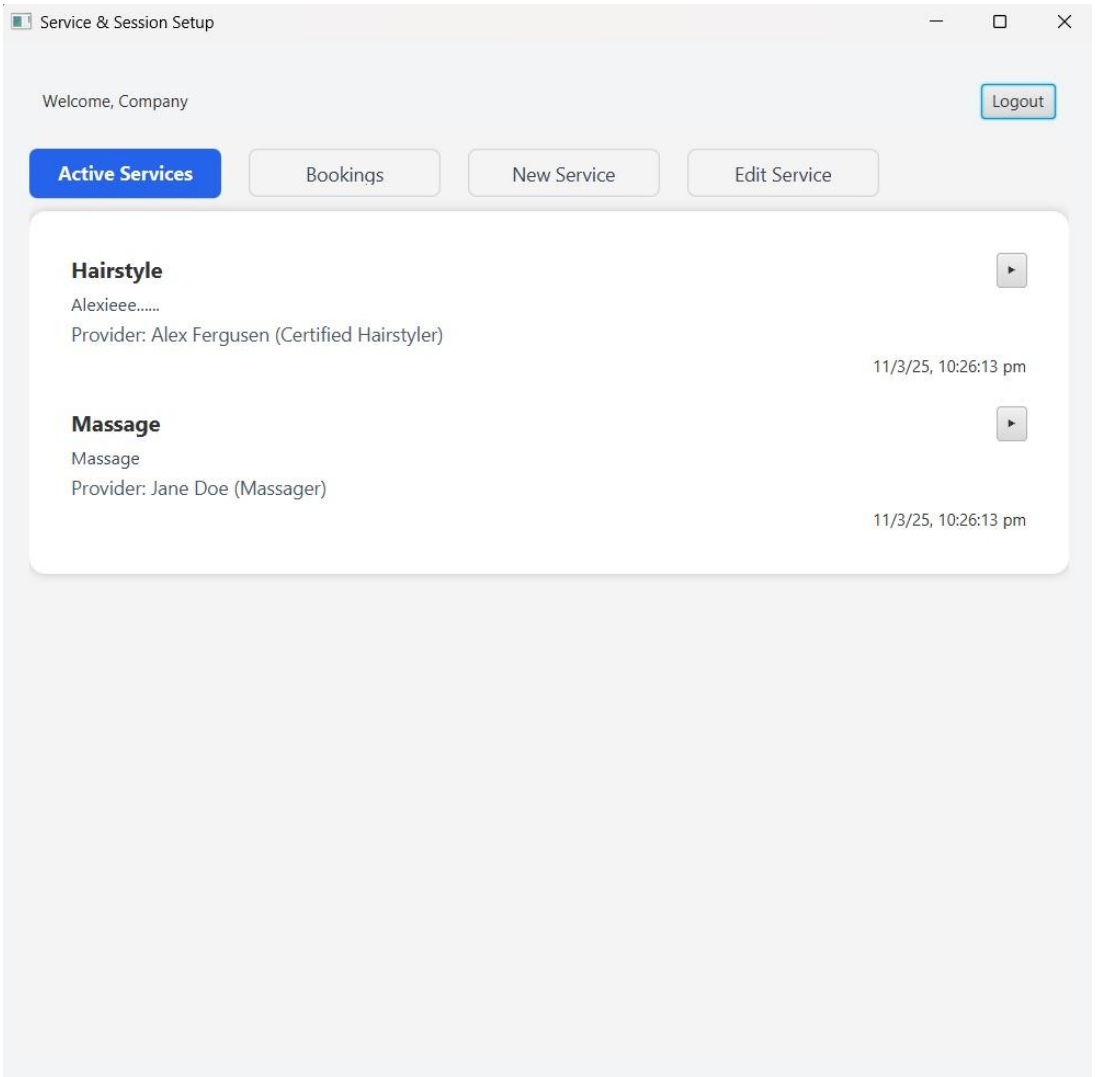

Fig 5.2 Dashboard – Active Services

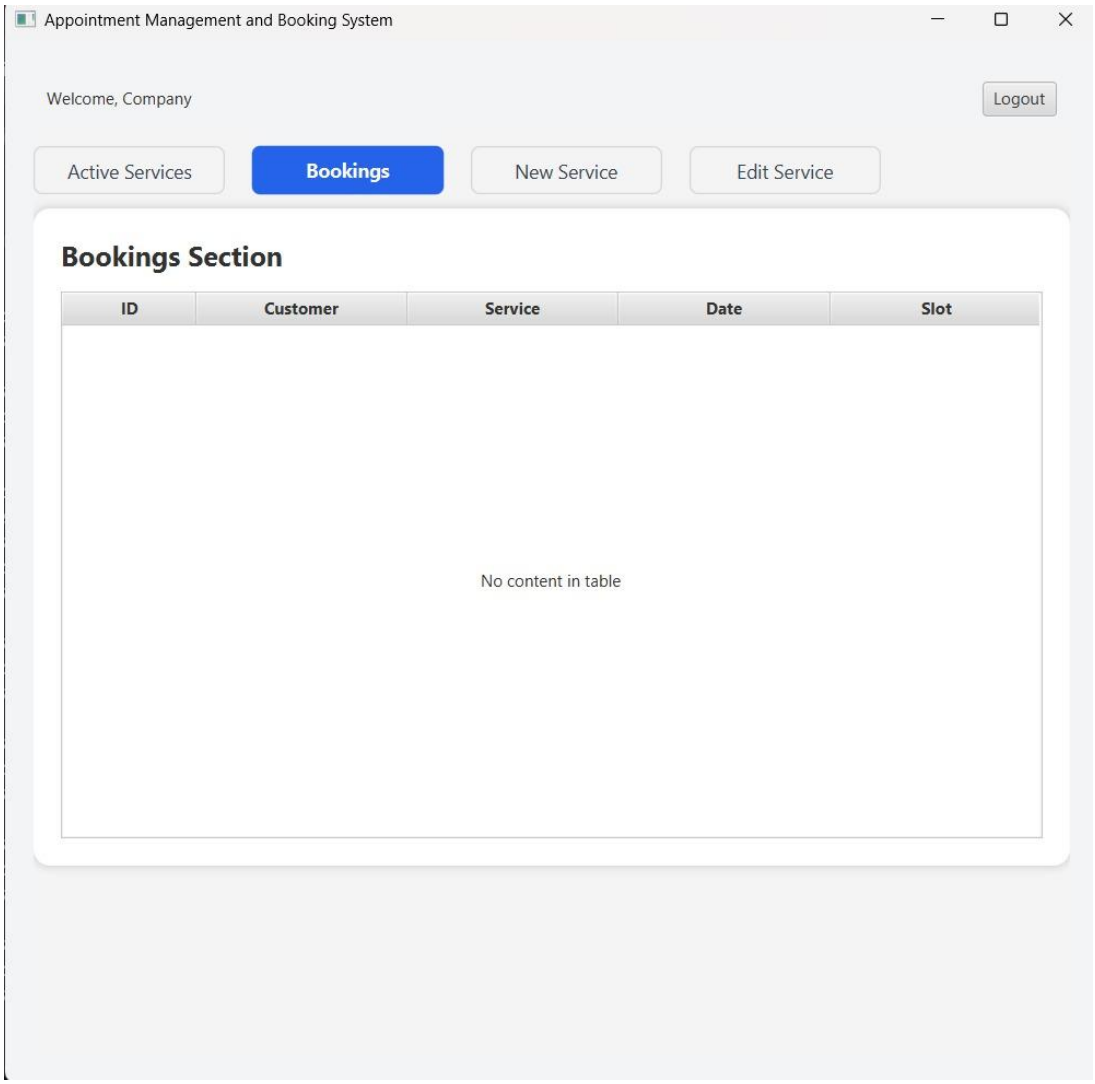

Fig 5.3 Dashboard - Booking

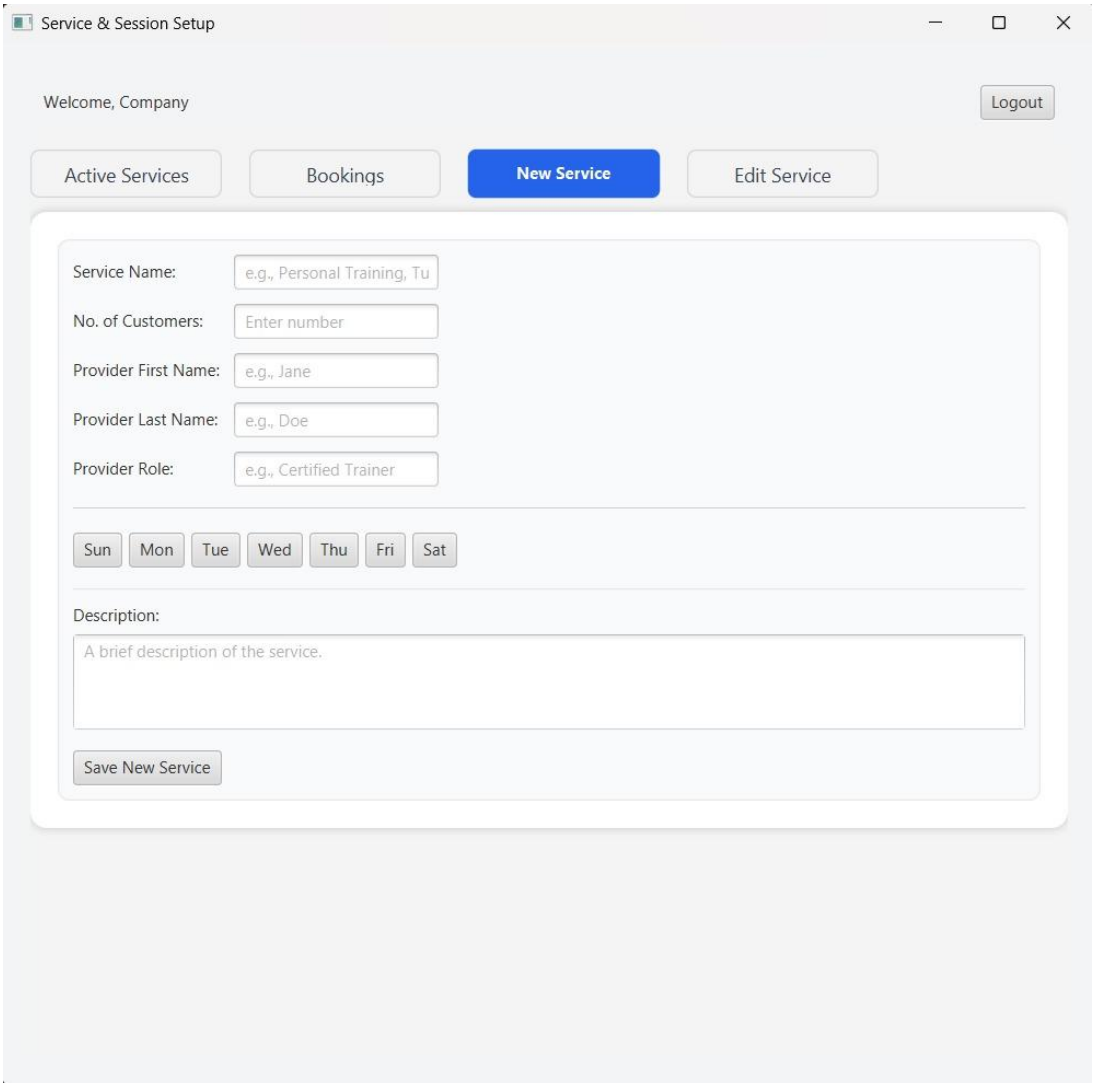

Fig 5.4 New Service

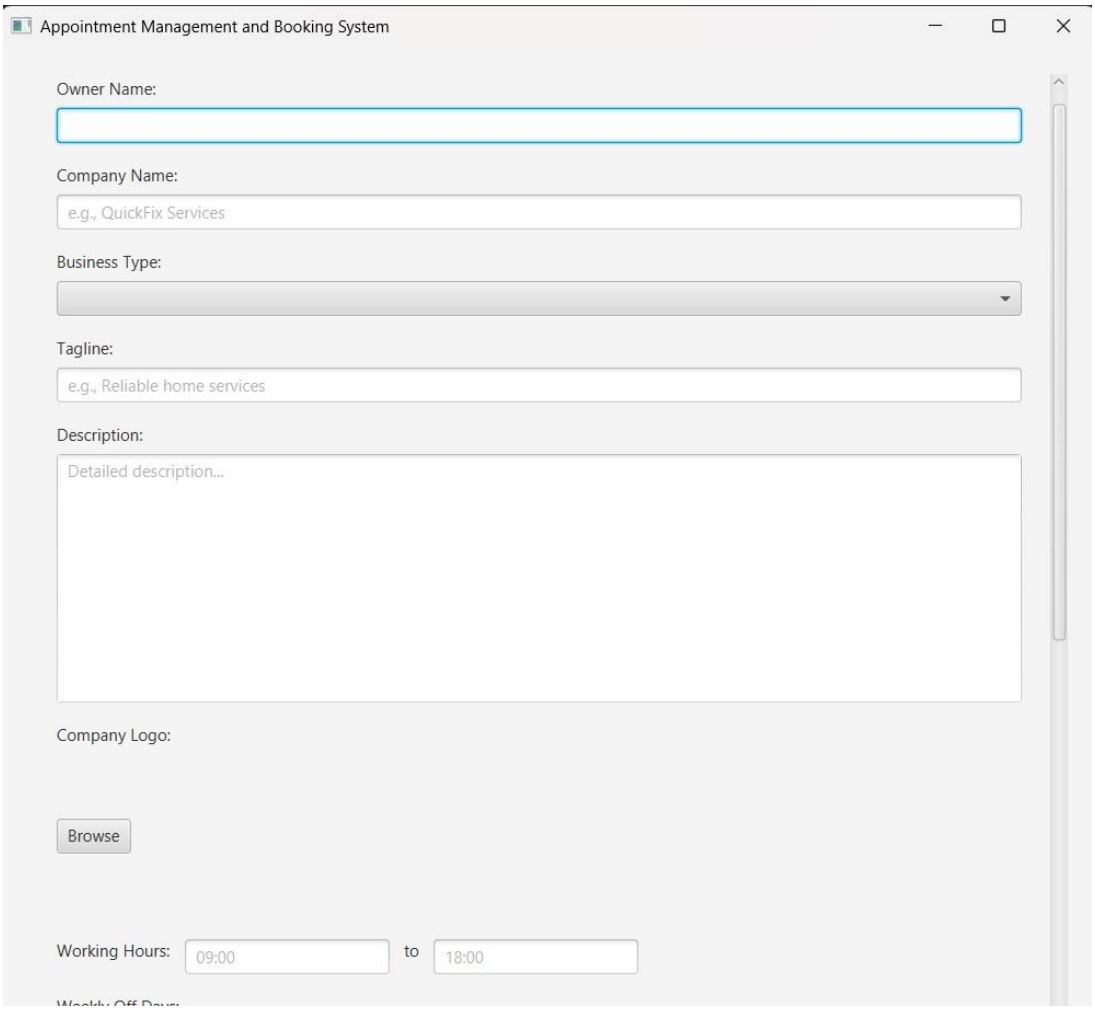

Fig 5.5 Company Bookings

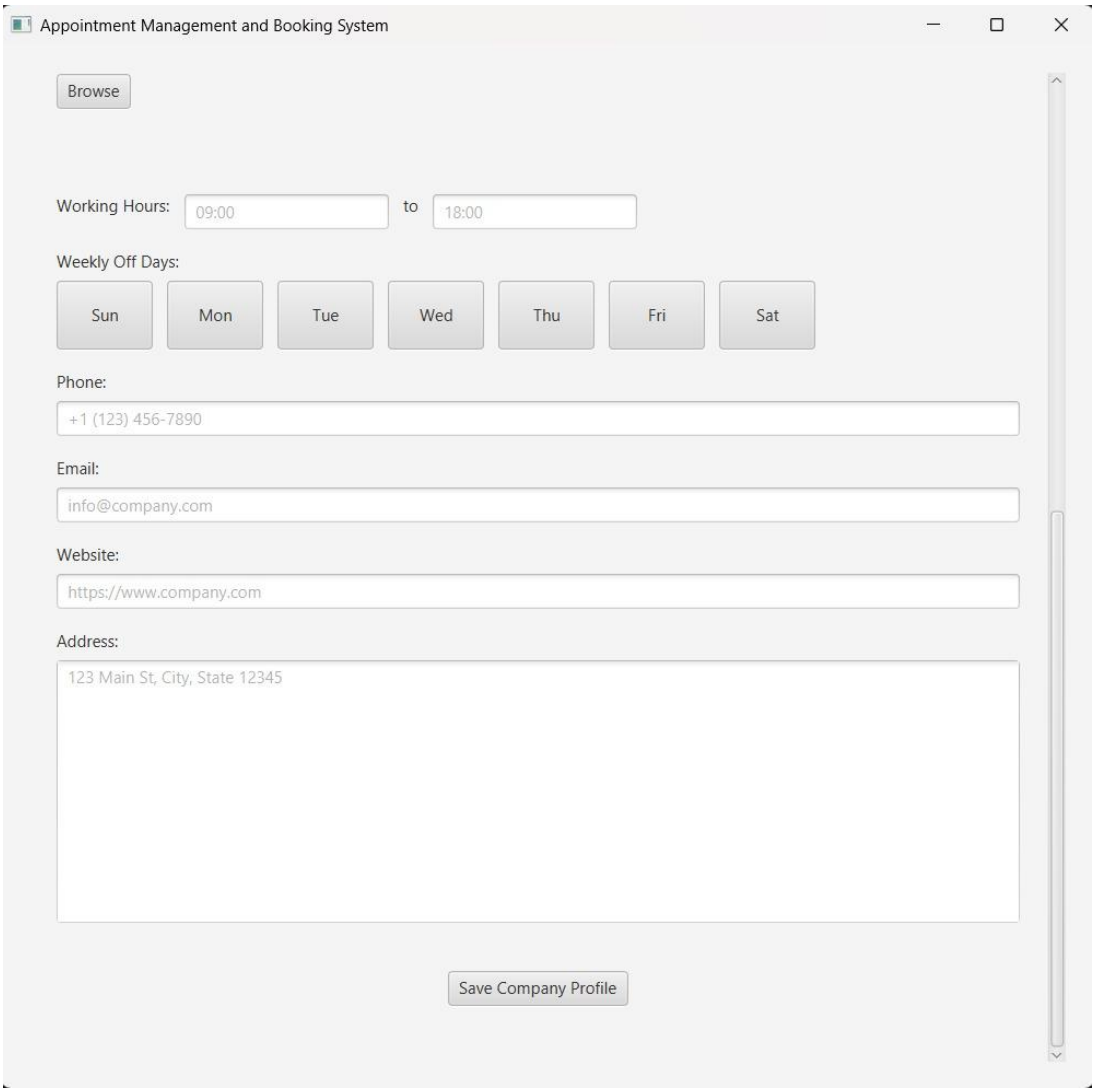

Fig 5.6 User Booking

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

The Appointment Management System (AMS) successfully transforms manual, error-prone scheduling into a seamless, automated digital platform tailored for Indian businesses. By enabling 24/7 self-service booking, real-time availability, and multi-channel reminders (SMS/WhatsApp/Email), AMS reduces no-show rates by ≥25%, eliminates double-bookings, and cuts administrative workload by up to 70%. With OTP-based authentication, Razorpay/UPI integration, and IT Act & GDPR compliance, the system ensures security and trust. The responsive, mobile-first UI empowers clients and staff alike, while the admin dashboard delivers actionable insights on revenue, performance, and occupancy. Deployed on AWS India (Mumbai), AMS is scalable, reliable, and cost-effective—driving operational efficiency and client satisfaction across healthcare, wellness, and professional services.

1.AI-Powered Slot Prediction:

Use machine learning to suggest optimal booking times based on historical no-show patterns and peak demand.

2. Mobile App (iOS/Android):

Native apps with push notifications, offline booking sync, and biometric login (fingerprint/face ID).

3.Video Consultation Integration:

Embed Zoom/Google Meet links for virtual appointments (telehealth, online coaching).

4.Customer Feedback & Rating System:

Post-appointment surveys with NPS scoring and public reviews to improve service quality.

5.Multi-Location & Franchise Support:

Manage multiple branches with centralized reporting and location-specific calendars.

6.Queue Management & Walk-ins:

Real-time queue display (like salons/hospitals) with token generation and estimated wait time.

7.Chatbot & Voice Booking:

WhatsApp chatbot or voice assistant (Google Assistant/Alexa) for instant bookings.

8.Advanced Analytics Dashboard:

Predictive revenue forecasting, heatmaps of busy hours, and customer lifetime value (CLV) tracking.

9.Loyalty & Subscription Plans:

Reward repeat clients with discounts or membership packages (e.g., 5-session packs).

10.Integration with Google Calendar & Outlook:

Two-way sync for staff personal calendars to avoid conflicts.

REFERENCES

1. https://www.w3schools.com/sql/
2. https://www.wikipedia.org/
3. https://grok.com/
4. https://chatgpt.com/