GLOBAL ACADEMY OF TECHNOLOGY

(Autonomous Institute, Affiliated to VTU, Accredited by NAAC with A Grade)

Rajarajeshwari Nagar, Bengaluru-560098

Department of Artificial Intelligence and Machine Learning

MINI PROJECT REPORT

On

DATABASE MANAGEMENT SYSTEMS

AML23403

By

MANISHANKAR D BHAT
1GA23AI025

G. MANJUNATH
1GA23AI016

Under the Guidance of

Prof. Vasugi I

GLOBAL ACADEMY OF TECHNOLOGY
growing ahead of time
Autonomous Institute, Affiliated to VTU

NEXT 25 YEARS

2024-2025

PROBLEM STATEMENT-35

Title: Vehicle Service Booking System

Objective:

Build a platform where users can book services for their vehicles and service centers can
manage appointments and service records.

System Requirements:

1. **Front-End:** React.js / HTML,CSS, JavaScript
2. **Back-End:** Node.js with Express / Python /Java / PHP
3. **Database:** MySQL / PostgreSQL / MongoDB

Core Functionalities:

1. User registration and login
2. Book service appointments by date, vehicle type, and service type
3. View service history and status updates
4. Admin/service center login to manage bookings, update service records

Deliverables:

1) A fully developed system demonstrating the objectives.
2) A detailed documentation with screenshots of input/output at each development stage.

TABLE OF CONTENTS

Chapter – 1

INTRODUCTION

In the modern digital age, service industries are increasingly adopting online systems to enhance customer convenience and streamline operations. The **Vehicle Service Booking System** is designed to address the inefficiencies of traditional vehicle servicing appointments, which typically involve phone calls, manual scheduling, and physical visits to service centers. These methods are not only time-consuming but also prone to miscommunication, overlapping bookings, and poor service tracking.

This web-based application aims to modernize and digitize the way vehicle services are booked and managed. The system provides a seamless interface for customers to schedule services for their vehicles — such as **Oil change**, **Tire change, General service** and **Specific part check** — from the comfort of their homes. It also enables service providers to maintain organized records of bookings, view service schedules, and optimize resource allocation.

Objectives of the System

- **Automate the service booking process**: Users can schedule appointments by selecting their desired service type, date, and time.
- **Improve user convenience**: No need to visit the service center or call for appointments — everything is available online.
- **Maintain a structured database**: All bookings are stored in an SQLite database, allowing easy retrieval and future referencing.
- **Ensure real-time interactivity**: The use of modern frontend technologies ensures a responsive and user-friendly interface.
- **Support modular development**: Designed in a scalable manner so features like user login, admin dashboard, and payment integration can be added in future.

Chapter – 2

SYSTEM REQUIREMENTS

Software Requirements

Component	Technology Used
Frontend	HTML5, CSS3, JavaScript
Backend	Python 3.x with Flask Framework
Database	SQLite
Server	Flask's built-in development server
Browser	Chrome, Firefox (modern browsers)
IDE	Visual Studio Code

Hardware Requirements

Resource	Minimum Specification
RAM	4 GB
Processor	Intel i3 or higher
Disk Space	200 MB (for source + DB)
Display	13" or larger, 1366x768 minimum

Chapter – 3

IMPLEMENTATION

The system architecture follows a **client-server model**, with the browser acting as the client and Flask as the backend server handling requests and responses.

3.1 Frontend Development

The frontend is created using a combination of HTML, CSS, and JavaScript. It is fully responsive and designed for ease of use.

- **Homepage**: Includes a login and registration page in which we can register and then login. It also has a light about the servicing of vehicles
- **Booking Modal**:
    - A pop-up modal allows users to choose from predefined services:
        - Oil Change
        - Tire change
        - General service
        - Specific parts check
    - Users enter their vehicle type(Car/Bike) and their desired date.
    - After booking a popup message will show up to confirm booking and details will be reset.
- **JavaScript Integration**:
    - Handles opening and closing of the modal.
    - Uses the Fetch API to send booking data asynchronously to the backend.

3.2 Backend Development

The backend is built using the **Flask** framework, which handles routing, API creation, and database interactions.

- **App Initialization**:
    - The app.py script sets up the web server and routes.
    - Includes a database initializer that creates the bookings table if it doesn't exist.

- **Flask Routes**:
  - /: Home route, renders frontend.
  - /api/book: Accepts POST requests with booking data.
  - /api/bookings: Returns a list of all bookings in JSON format.
- **APIs**:
  - **POST /api/book**:
    - Validates JSON payload.
    - Inserts data into the SQLite database.
    - Returns confirmation message.
  - **GET /api/bookings**:
    - Retrieves all booking entries.
    - Returns them as a JSON array.

3.3 Database Integration

- **Database Used**: SQLite
- **File**: database.db
- **Advantages**:
  - Lightweight, file-based database suitable for mini projects.
  - Easy integration with Python using sqlite3 module.

Chapter – 4

SCREENSHOTS

SCREENSHOTS

Fig 1: Homepage

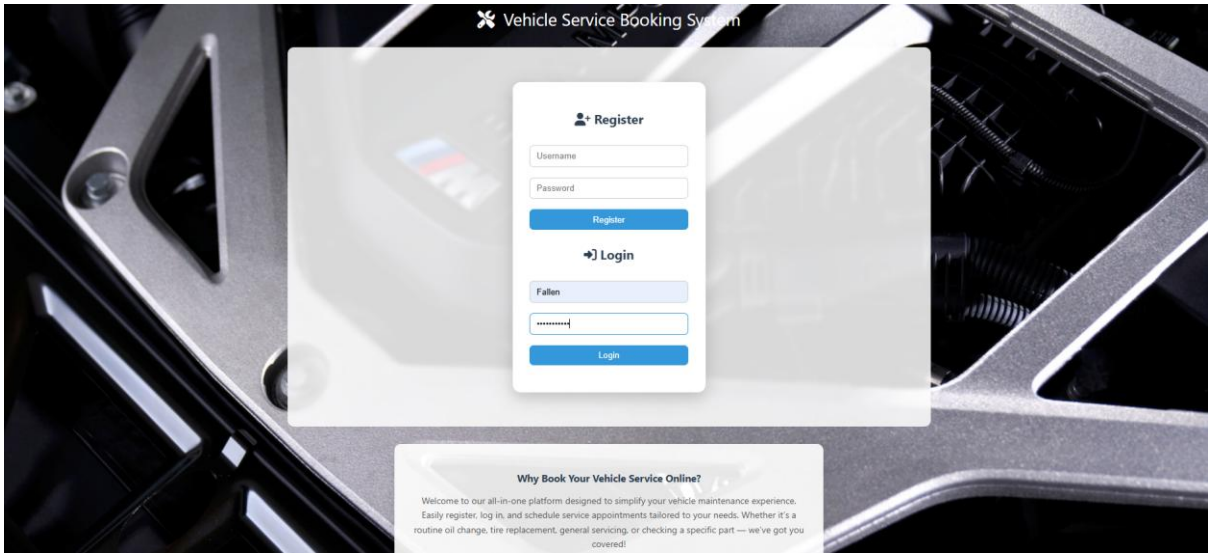

Fig 2:Booking page

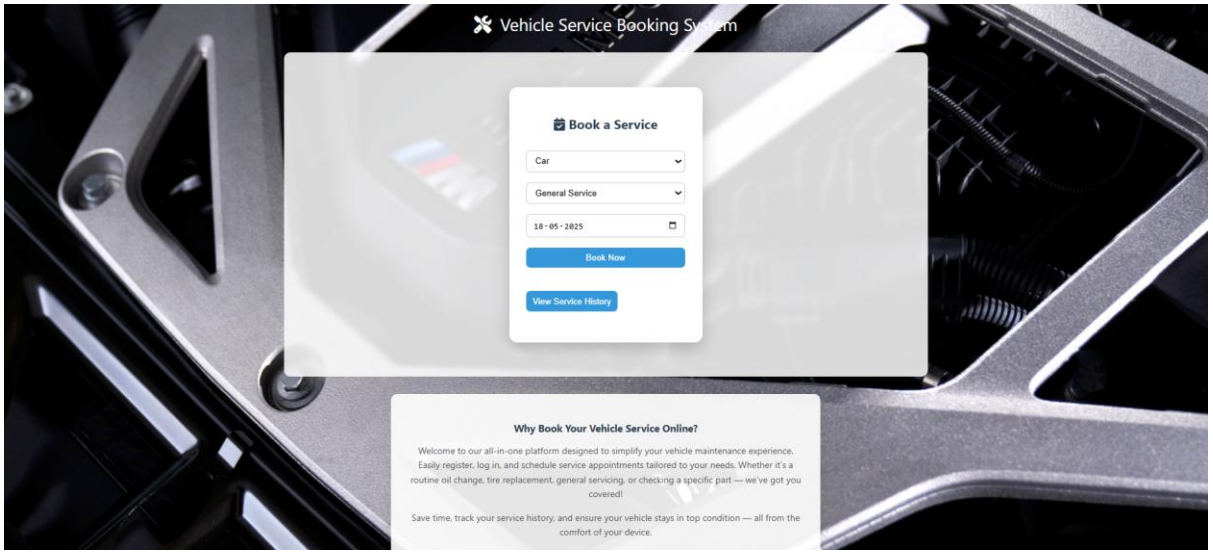

Fig 3:Confirmation

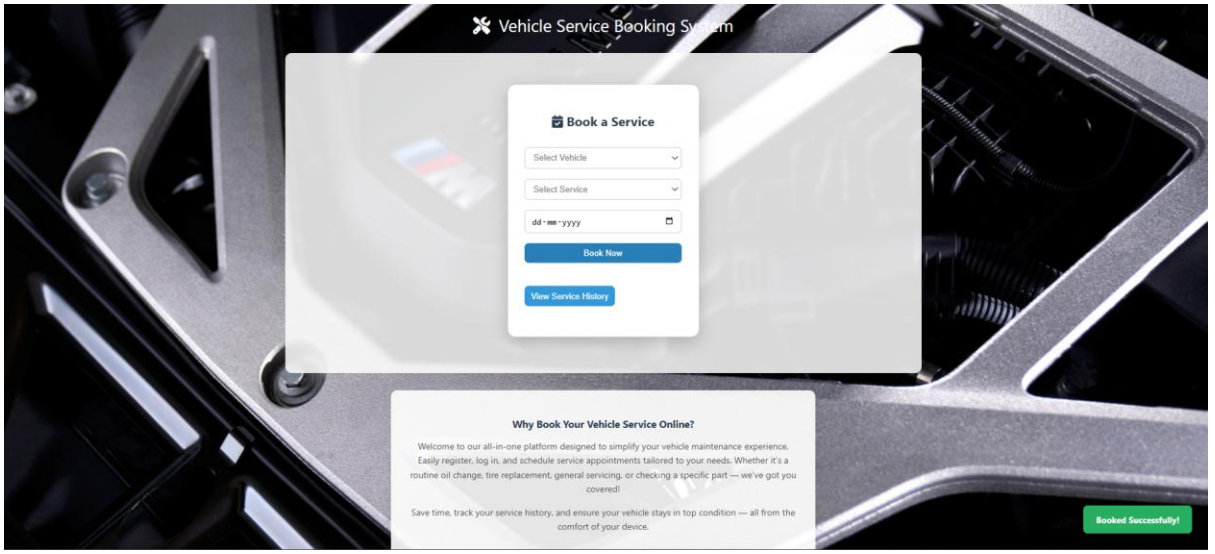

Fig 4:Service History

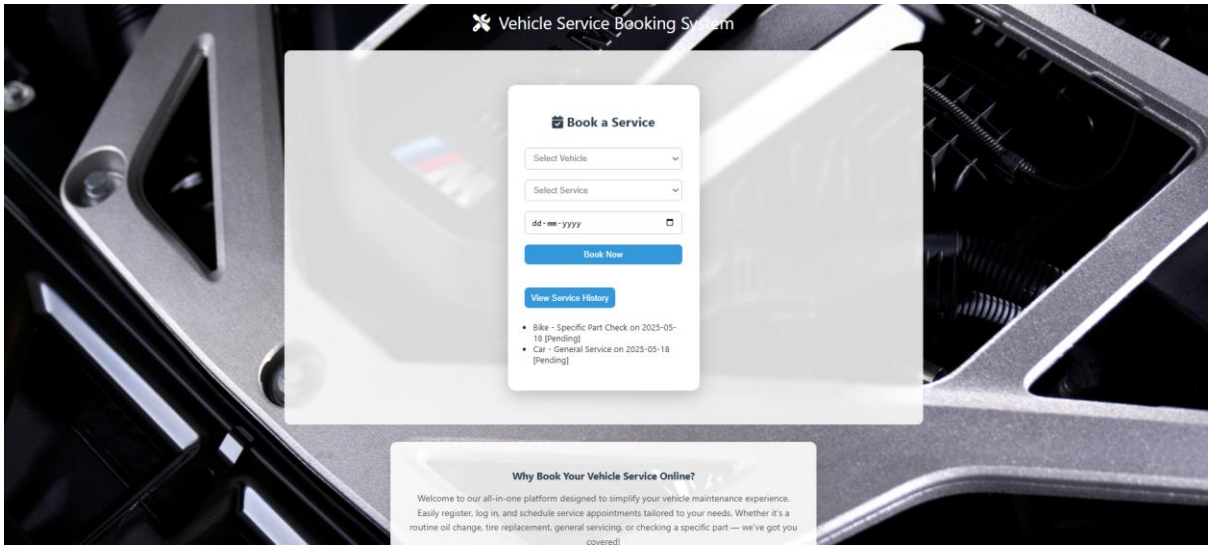

Fig 5:Database

	id	user_id	vehicle_ty...	service_type	service_d...	status
	Filte	Filte	Filte	Filter.	Filte	Filte
1	1	1	Bike	Specific Part Check	2025-05-18	Pending
2	2	1	Car	General Service	2025-05-18	Pending
3						

Chapter – 5

CONCLUSION AND FUTURE SCOPE

Conclusion

This project showcases a complete mini web application implementing end-to-end CRUD functionality. It provides a modern, fast, and intuitive interface for booking vehicle services and storing service data persistently. The use of Flask and SQLite makes it an efficient and lightweight solution suitable for small to mid-scale service centers.

Key Features Implemented:

- Dynamic frontend with JavaScript interaction
- RESTful API with Flask
- SQLite database integration
- Booking confirmation and validation

Future Scope

The current system can be enhanced further with the following features:

Feature	Description
Authentication System	Implement user login/register functionality for security and personalization.
Admin Panel	Allow service center admins to manage bookings, approve/reject requests, and view statistics.
Service History Page	Enable users to view and manage their past and upcoming bookings.
Notification System	Send booking confirmations and reminders via email/SMS.
Payment Gateway	Integrate online payment options such as Razorpay or Stripe.
Mobile App	Extend the system to Android/iOS platforms using React Native or Flutter.
Feedback System	Collect and display user feedback for service centers.

******END******