PITSTOP MANAGER

A MINI-PROJECT REPORT

Submitted by

JAYASRI B 241801101

JAYASREE D 241801100

in partial fulfillment of the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

NOVEMBER 2025

BONAFIDE CERTIFICATE

Certified that this project **"PIT STOP MANAGER"** is the bonafide work of **"JAYASRI B,JAYASREE D"** who carried out the project work under my supervision.

SIGNATURE

Dr. E.K.SUBRAMANIAN

ASSOCIATE PROFESSOR

Dept. of Artificial Intelligence and Data Science,
Rajalakshmi Engineering College
Chennai

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

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

I would like to express my sincere thanks to our college, Rajalakshmi Engineering College, for giving me the opportunity to do this project titled “Pitstop Manager”.

I am thankful to all the faculty members of the Artificial intelligence and data science Department for their guidance and support throughout the completion of this project.

I also thank my friends and family for encouraging me and helping me during the project work.

Finally, I would like to thank everyone who supported me directly or indirectly in completing this project successfully.

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 **Dr. J.M. GNANASEKAR** for being ever supporting force during our project work

We also extend our sincere and hearty thanks to our internal guide **Dr. E.K. SUBRAMANIAN,** 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.

JAYASRI B

JAYASREE D

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

A Pitstop Manager is responsible for overseeing and coordinating all operations within a vehicle service pitstop or automotive service outlet. The role involves managing service workflows, ensuring quick turnaround time for vehicles, supervising technicians, maintaining customer satisfaction, and ensuring safety and quality standards. A Pitstop Manager acts as the link between customers, technicians, and management to deliver efficient, reliable, and high-quality service.

1.2 SCOPE OF THE WORK

Scope of Work

The Pitstop Manager project focuses on creating a database-driven system that simplifies and automates the daily operations of an automobile service center or pitstop. The system is designed to store, manage, and process data related to customers, vehicles, mechanics, service bookings, spare parts, and billing in an organized way.

The main scope of this project includes:

- **Developing a centralized database to maintain all service-related records securely and efficiently.**
- **Allowing users to add, update, delete, and view details of customers, vehicles, and mechanics.**

- **Managing service appointments to avoid scheduling conflicts and ensure proper resource allocation.**
- **Tracking vehicle service history for future reference and maintenance reminders.**
- **Generating bills and service reports automatically to minimize manual errors.**
- **Providing search and filter options for easy retrieval of data.**
- **Ensuring data integrity and consistency through relational database concepts like primary keys, foreign keys, and normalization.**

This project helps in improving the accuracy of records, speeding up service processes, and reducing paperwork. It benefits both management and customers by providing transparency and efficiency in operations.

In the future, the system can be extended to include online booking, spare part inventory tracking, customer notifications, and report analytics.

1.3 PROBLEM STATEMENT

Managing a vehicle service center manually is time-consuming, error-prone, and inefficient. Service centers often face difficulties in maintaining customer details, tracking vehicle service history, scheduling appointments, managing mechanic availability, and generating accurate bills.

Paper-based or spreadsheet-based systems lead to data duplication, misplaced records, and delays in service.

There is a need for a database management system that can automate these operations, store all information in an organized manner, and allow quick access and updates. The proposed Pitstop Manager aims to solve these problems by developing a centralized database that efficiently handles all aspects of a service center — from customer registration to billing — ensuring accuracy, reliability, and smooth workflow.

1.4 AIM AND OBJECTIVES OF THE PROJECT

The aim of the Pitstop Manager project is to design and develop a system that efficiently manages pitstop/vehicle service operations by streamlining service workflow, employee task allocation, customer handling, inventory tracking, and service quality monitoring to ensure fast, reliable, and customer-friendly vehicle servicing.

OBJECTIVES

1. To automate pitstop operations such as service scheduling, job assignment, service tracking, and delivery updates.

2. To reduce turnaround time (TAT) for vehicle servicing by optimizing workflow and resource allocation.

3. To enhance customer experience through transparent service updates, timely communication, and quality feedback systems.

4. To maintain accurate records of customer details, service history, technician performance, and billing information.

5. To monitor and manage inventory of tools, spare parts, and consumables to avoid service delays.

6. To ensure high service quality & safety standards by applying performance checks and standard operating procedures (SOPs).

7. To support decision-making through reports and analytics on pitstop performance, revenue, customer satisfaction, and employee productivity.

8. To improve operational efficiency by minimizing manual errors, duplication of work, and communication gaps.

9. To provide a user-friendly interface for managers, technicians, and customers to access required information easily.

10. To enable scalability, allowing the system to support multiple service bays or branches in future enhancements.

CHAPTER 2

SYSTEM SPECIFICATIONS

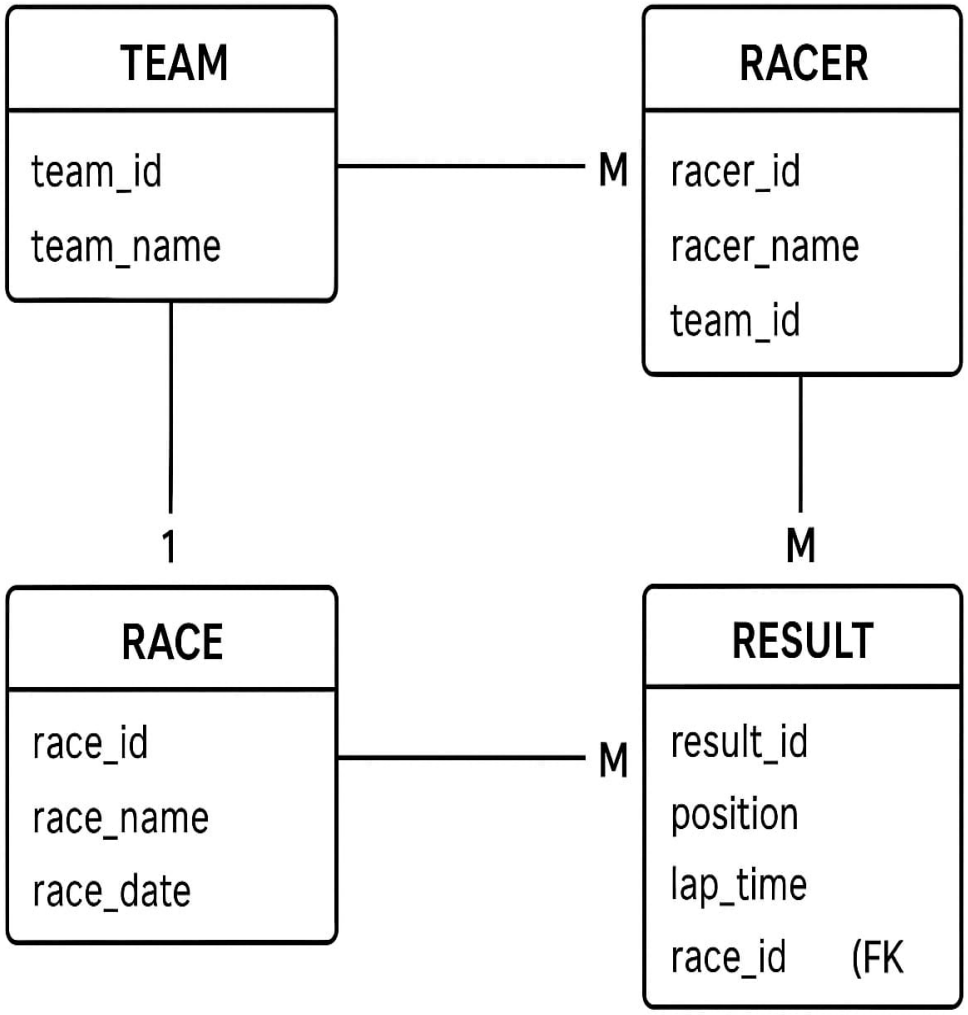
2.1 HARDWARE SPECIFICATIONS

Processor	:	Intel i5
Memory Size	:	8GB (Minimum)
HDD	:	1 TB (Minimum)

2.2 SOFTWARE SPECIFICATIONS

Operating System	:	WINDOWS 11
Front – End	:	Java
Back - End	:	MySql
Language	:	Java,SQL

ER DIAGRAM


TEAM
team_id
team_name
RACER
racer_id
racer_name
team_id
M
1
M
RACE
race_id
race_name
race_date
RESULT
result_id
position
lap_time
race_id (FK
M

CHAPTER 3

MODULE DESCRIPTION

The Pitstop Manager system is divided into several modules to manage different activities of a vehicle service center efficiently.

1. **Customer Management: Stores customer details and their service history.**
2. **Vehicle Management: Maintains vehicle information and links it with the customer.**
3. **Service Booking: Handles service scheduling and prevents overlapping of appointments.**
4. **Mechanic Management: Keeps mechanic details, assigns tasks, and tracks availability.**
5. **Billing and Payment: Calculates charges, generates invoices, and records payments.**
6. **Inventory Management (optional): Manages spare parts stock and updates inventory after service.**

CHAPTER 4

SAMPLE CODING

SQL

```
-- Customers
CREATE TABLE Customer (
  customer_id INTEGER PRIMARY KEY AUTOINCREMENT,
  name TEXT NOT NULL,
  phone TEXT,
  email TEXT,
  address TEXT
);
-- Vehicles
CREATE TABLE Vehicle (
  vehicle_id INTEGER PRIMARY KEY AUTOINCREMENT,
  reg_no TEXT UNIQUE NOT NULL,
  model TEXT,
```

```
  manufacturer TEXT,

  customer_id INTEGER NOT NULL,

  mileage INTEGER DEFAULT 0,

  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)

);

-- Mechanics

CREATE TABLE Mechanic (

  mechanic_id INTEGER PRIMARY KEY AUTOINCREMENT,

  name TEXT NOT NULL,

  specialization TEXT,

  phone TEXT

);

-- Services (catalog of service types)

CREATE TABLE ServiceType (

  service_id INTEGER PRIMARY KEY AUTOINCREMENT,

  service_name TEXT NOT NULL,
```

```
  base_cost REAL NOT NULL
);
-- Bookings / Appointments
CREATE TABLE Booking (
  booking_id INTEGER PRIMARY KEY AUTOINCREMENT,
  vehicle_id INTEGER NOT NULL,
  mechanic_id INTEGER,
  service_id INTEGER NOT NULL,
  booking_date TEXT NOT NULL,      -- store as ISO date/time string
  status TEXT DEFAULT 'Scheduled',    -- Scheduled, InProgress, Completed, Cancelled
  remarks TEXT,
  FOREIGN KEY (vehicle_id) REFERENCES Vehicle(vehicle_id),
  FOREIGN KEY (mechanic_id) REFERENCES Mechanic(mechanic_id),
  FOREIGN KEY (service_id) REFERENCES ServiceType(service_id)
);
```

```
-- Parts / Inventory
CREATE TABLE Part (
  part_id INTEGER PRIMARY KEY AUTOINCREMENT,
  part_name TEXT NOT NULL,
  quantity INTEGER DEFAULT 0,
  unit_price REAL DEFAULT 0
);
-- Billing / Invoice
CREATE TABLE Invoice (
  invoice_id INTEGER PRIMARY KEY AUTOINCREMENT,
  booking_id INTEGER NOT NULL,
  total_amount REAL NOT NULL,
  paid_amount REAL DEFAULT 0,
  invoice_date TEXT,
  FOREIGN KEY (booking_id) REFERENCES Booking(booking_id)
);
```

PYTHON

```
import sqlite3

from datetime import datetime

DB = "pitstop.db"

def init_db():

    with sqlite3.connect(DB) as conn:

        cur = conn.cursor()

        # execute the DDL from above (for brevity here we assume you've run the SQL DDL)

        # Example: create a simple Customer table if not exists

        cur.execute('''

        CREATE TABLE IF NOT EXISTS Customer (

         customer_id INTEGER PRIMARY KEY AUTOINCREMENT,

         name TEXT NOT NULL,

         phone TEXT,

         email TEXT,

         address TEXT
```

```
        )''')
        conn.commit()
def add_customer(name, phone=None, email=None, address=None):
    with sqlite3.connect(DB) as conn:
        cur = conn.cursor()
        cur.execute("INSERT INTO Customer (name, phone, email,
address) VALUES (?, ?, ?, ?)",
                    (name, phone, email, address))
        conn.commit()
        return cur.lastrowid
def add_vehicle(reg_no, model, manufacturer, customer_id,
mileage=0):
    with sqlite3.connect(DB) as conn:
        cur = conn.cursor()
        cur.execute('''CREATE TABLE IF NOT EXISTS Vehicle (
            vehicle_id INTEGER PRIMARY KEY AUTOINCREMENT,
            reg_no TEXT UNIQUE NOT NULL,
```

```
        model TEXT, manufacturer TEXT, customer_id INTEGER,
mileage INTEGER DEFAULT 0

    )''')

    cur.execute("INSERT INTO Vehicle (reg_no, model,
manufacturer, customer_id, mileage) VALUES (?, ?, ?, ?, ?)",

              (reg_no, model, manufacturer, customer_id, mileage))

    conn.commit()

    return cur.lastrowid

def book_service(vehicle_id, mechanic_id, service_id,
booking_date=None, remarks=None):

  booking_date = booking_date or datetime.now().strftime("%Y-
%m-%d %H:%M:%S")

  with sqlite3.connect(DB) as conn:

    cur = conn.cursor()

    cur.execute('''CREATE TABLE IF NOT EXISTS Booking (

      booking_id INTEGER PRIMARY KEY
AUTOINCREMENT,

      vehicle_id INTEGER NOT NULL, mechanic_id INTEGER,
service_id INTEGER NOT NULL,
```

```
        booking_date TEXT NOT NULL, status TEXT DEFAULT
'Scheduled', remarks TEXT

    )''')

    cur.execute("INSERT INTO Booking (vehicle_id, mechanic_id,
service_id, booking_date, remarks) VALUES (?, ?, ?, ?, ?)",

              (vehicle_id, mechanic_id, service_id, booking_date,
remarks))

    conn.commit()

    return cur.lastrowid

def generate_invoice(booking_id, total_amount, paid_amount=0):

  with sqlite3.connect(DB) as conn:

    cur = conn.cursor()

    cur.execute('''CREATE TABLE IF NOT EXISTS Invoice (

      invoice_id INTEGER PRIMARY KEY AUTOINCREMENT,

      booking_id INTEGER NOT NULL, total_amount REAL
NOT NULL,

      paid_amount REAL DEFAULT 0, invoice_date TEXT

    )''')
```

```
        cur.execute("INSERT INTO Invoice (booking_id, total_amount, paid_amount, invoice_date) VALUES (?, ?, ?, ?)",
                    (booking_id, total_amount, paid_amount, datetime.now().strftime("%Y-%m-%d")))
        conn.commit()
        return cur.lastrowid

def pending_payments():
    with sqlite3.connect(DB) as conn:
        cur = conn.cursor()
        cur.execute("""
            SELECT i.invoice_id, i.total_amount, i.paid_amount, i.total_amount - i.paid_amount AS due
            FROM Invoice i
            WHERE i.total_amount > i.paid_amount
        """)
        return cur.fetchall()

if __name__ == "__main__":
    init_db()
```

```
cid = add_customer("Asha Rao", "9876543210", "asha@example.com", "Chennai")

vid = add_vehicle("TN01AB1234", "Swift", "Maruti", cid, 42000)

bid = book_service(vid, None, 1, "2025-11-10 10:00:00", "Full service")

inv = generate_invoice(bid, 2800, 0)

print("Customer ID:", cid, "Vehicle ID:", vid, "Booking ID:", bid, "Invoice ID:", inv)

print("Pending:", pending_payments())
```

CHAPTER 5

SCREEN SHOTS

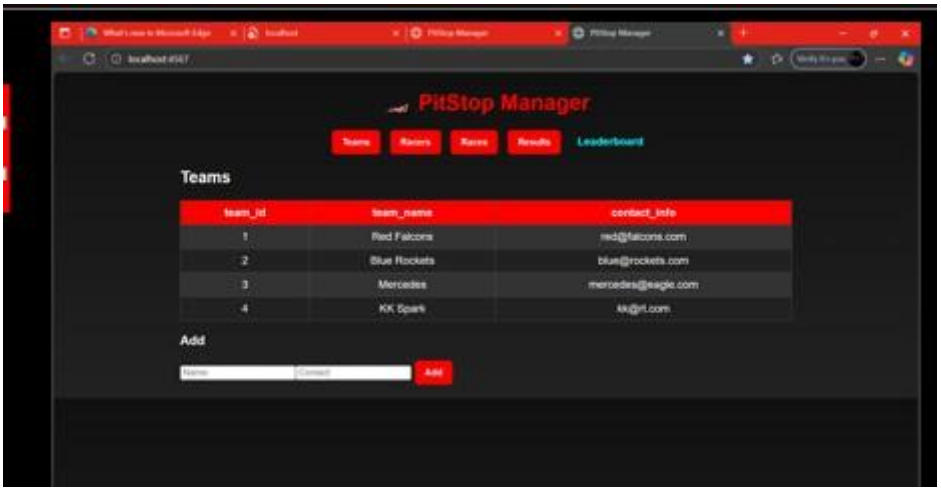


Fig 5.2

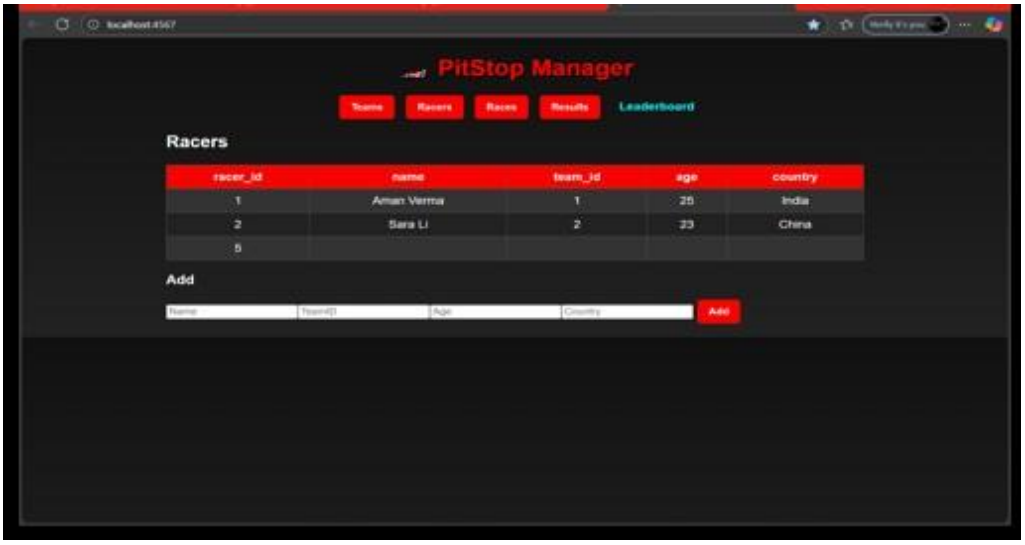


Fig 5.3

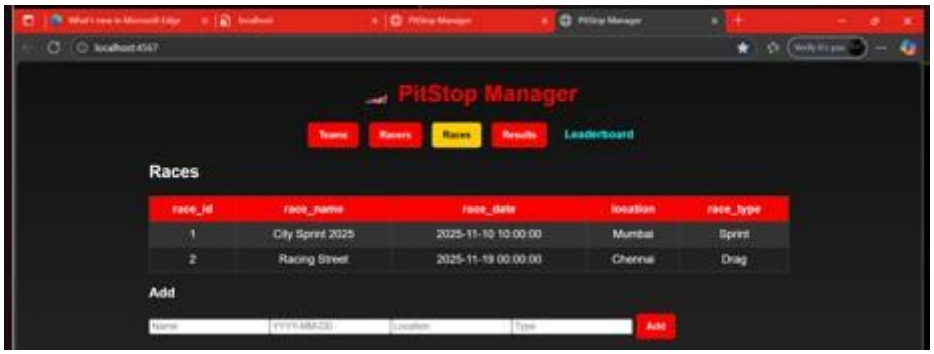


Fig 5.4

Fig 5.5

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

The Pitstop Manager project helps make the work in a vehicle service pitstop easier and more organized. It reduces manual work, saves time, and helps the manager handle customers, workers, and service details smoothly. The system improves the speed and quality of service and keeps proper records of all activities. Overall, it makes pitstop operations more efficient and increases customer satisfaction.

In the future, the Pitstop Manager system can be improved with new features such as:

A mobile app for customers and workers to track service and updates easily.

Online payment and e-bill options for customer convenience.

Automatic stock checking and ordering spare parts when items are low.

Customer feedback system to improve service quality.

Support for multiple branches so one system can manage many pitstops.

Data reports and charts to help the manager make better decisions.

REFERENCES

1. https://www.w3schools.com/sql/
2. https://www.tutorialspoint.com/sqlite/index.htm
3. https://www.wikipedia.org/
4. https://www.codecademy.com/learn/learn-python