A D Patel Institute of Technology
(A Constituent College of CVM University)

A.D. PATEL INSTITUTE OF TECHNOLOGY

(A Constituent College of CVM University)

New V. V. Nagar

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & DATA SCIENCE

Mini Project Seminar

Submitted By

Name of Student: Hani Jayeshkumar Patel (**12202120601017**)

Name of Student: Jahnvi Rajesh Mistry (**12202120601026**)

Programming with Java (202044502)

Semester 5 A.Y. 2024-25

1. Introduction

Project Title: Car Rental Management System

Objective:
The Car Rental Management System is a desktop-based application designed to streamline and automate the process of renting vehicles for car rental agencies. The primary objective of this project is to enable efficient and error-free booking, billing, and customer management in car rental operations.
This system provides:

1. **Ease of Booking**: A user-friendly interface allows customers to browse available cars, view rental costs, and make reservations easily.
2. **Improved Fleet Management**: Helps the agency keep track of available and rented vehicles, car types, rental costs, and vehicle locations.
3. **Customer Management**: Maintains customers' records, including contact details and rental histories, to facilitate personalized services and enable VIP perks for loyal customers.
4. **Billing and Revenue Tracking**: This system automates billing based on reservation data, tracks revenue generation, and provides access to real-time financial reporting.
5. **Reservation Management**: Supports upcoming, pending, and canceled reservations to allow customers to reschedule or cancel bookings within the system.

This application aims to enhance customer satisfaction, optimize resource management, and increase operational efficiency for car rental businesses.

Technology Stack:

- **Programming Language**: Java - Chosen for its cross-platform capabilities, object-oriented features, and extensive libraries.
- **Java Swing**: Used for creating the graphical user interface (GUI), providing a native look and feel for a desktop application.
- **Java AWT**: Assists with GUI components that require more specific layout and painting requirements.
- **Java Collections Framework**: Utilized for data management, including arrays and lists, which store cars, customers, and reservations.
- **Exception Handling**: Robust error-handling mechanisms are implemented to manage user input errors and unexpected system states.
- **Multithreading**: (If applicable) Used for handling concurrent operations such as simultaneous bookings.
- **File Handling**: Manages data persistence to store reservation records, customer information, and billing details.

2. Project Overview

System Architecture:
The Car Rental Management System follows a layered architecture, consisting of the following components:

- **Presentation Layer (GUI):** Java Swing components provide the user interface, enabling interactions like booking, viewing records, and generating reports.
- **Service Layer:** The core business logic, which includes operations for managing customers, reservations, billing, and reporting. This layer coordinates between the presentation and data layers.
- **Data Layer:** Manages data storage for cars, customers, and reservations. The data layer utilizes Java Collections for storing information, as well as file handling to persist data across sessions.

Features:

- **User Management:** Ability to add, edit, and manage customer information, including VIP status.
- **Fleet Management:** Management of car information, including model, brand, type, location, and availability status.
- **Reservation Management:** Creating, editing, and tracking reservations; showing available and rented cars.
- **Billing and Payments:** Automated billing calculation, payment management, and tracking of total revenue.
- **Reporting:** Summary reports for total revenue, pending reservations, and availability status.
- **Support Module:** Placeholder for future customer support features.

Modules:

- **Car Module:** Manages car data, including rental cost, type, and availability.
- **Customer Module:** Manages customer details, with VIP status management**.**
- **Reservation Module:** Handles reservation details like dates, status, and associated customer and car.
- **Billing Module:** Calculates and tracks payments and overall revenue.
- **Rental Service:** A main controller that connects the different modules and oversees the business logic.

3. Implementation Details

Inheritance and Polymorphism:

The project employs Java's object-oriented features, especially inheritance and polymorphism:

- **Inheritance:** Classes like Reservation and Customer extend general functionality through specific attributes and methods.
- **Polymorphism:** Achieved through method overriding, allowing methods like calculateTotalAmount in the Reservation class to be customized as needed.

Inner Classes:

Several inner classes are used within the main CarRentalSystem class:

- **Car:** Represents individual car objects, with properties for the model, type, brand, etc.
- **Customer:** Manages customer-specific details, such as name, contact information, and VIP status.
- **Reservation:** Manages each booking, including date range, customer, and car details.
- **Billing:** Keeps track of all completed transactions and calculates the total revenue.

Exception Handling

Error-handling mechanisms are implemented across the system to manage:

- **User Input Validation**: Ensures fields are completed and contain valid information before proceeding.
- **Operational Errors**: Handles exceptions that may arise during database access or data processing.
- **File Handling Errors**: Manages read/write exceptions for file-based persistence.

Threading

The system may use threading in cases like simultaneous bookings to handle concurrent operations and prevent data conflicts.

Collection API

Java Collections, including ArrayList and HashMap, are used extensively:

- **ArrayList**: For managing lists of cars, customers, and reservations.
- **HashMap**: Could be used for more complex data management (e.g., storing cars by location).

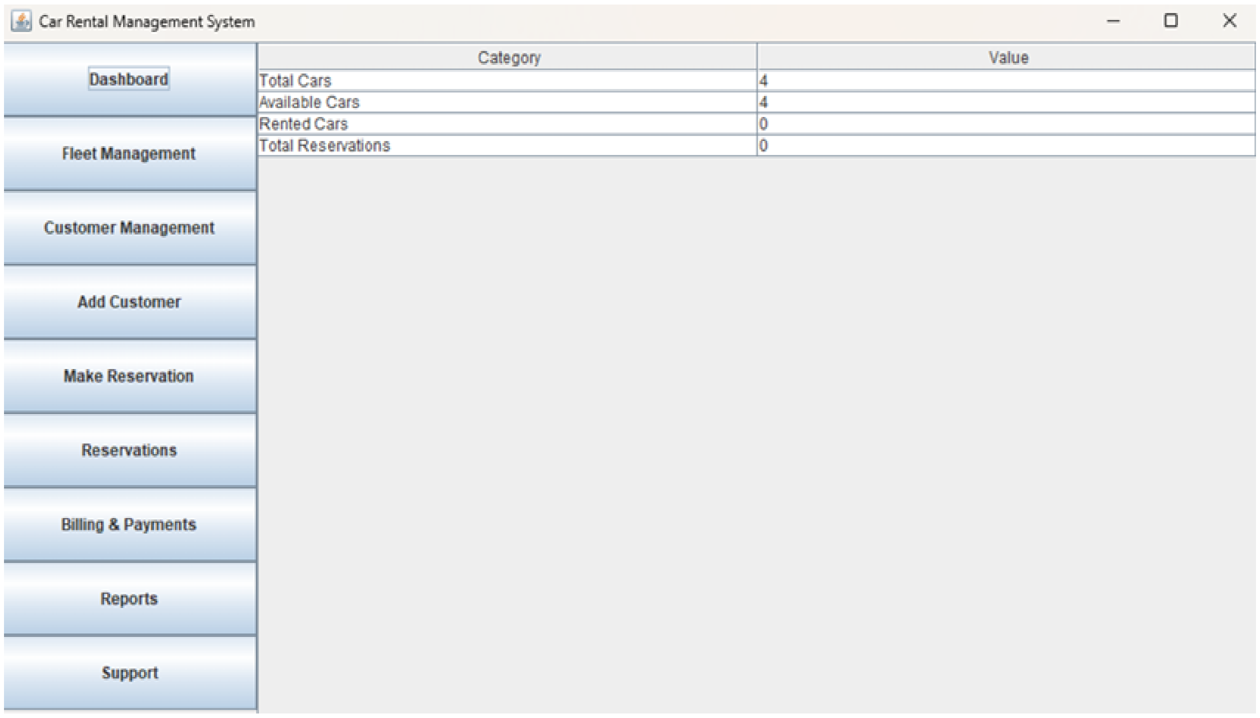
File Handling

- The system uses Java's file I/O to persist reservation and customer data. This is critical for ensuring data is retained across sessions.
- Overall, these implementation details highlight using Java's object-oriented principles and various features to create a functional and efficient car rental system.

4. Project Demonstration

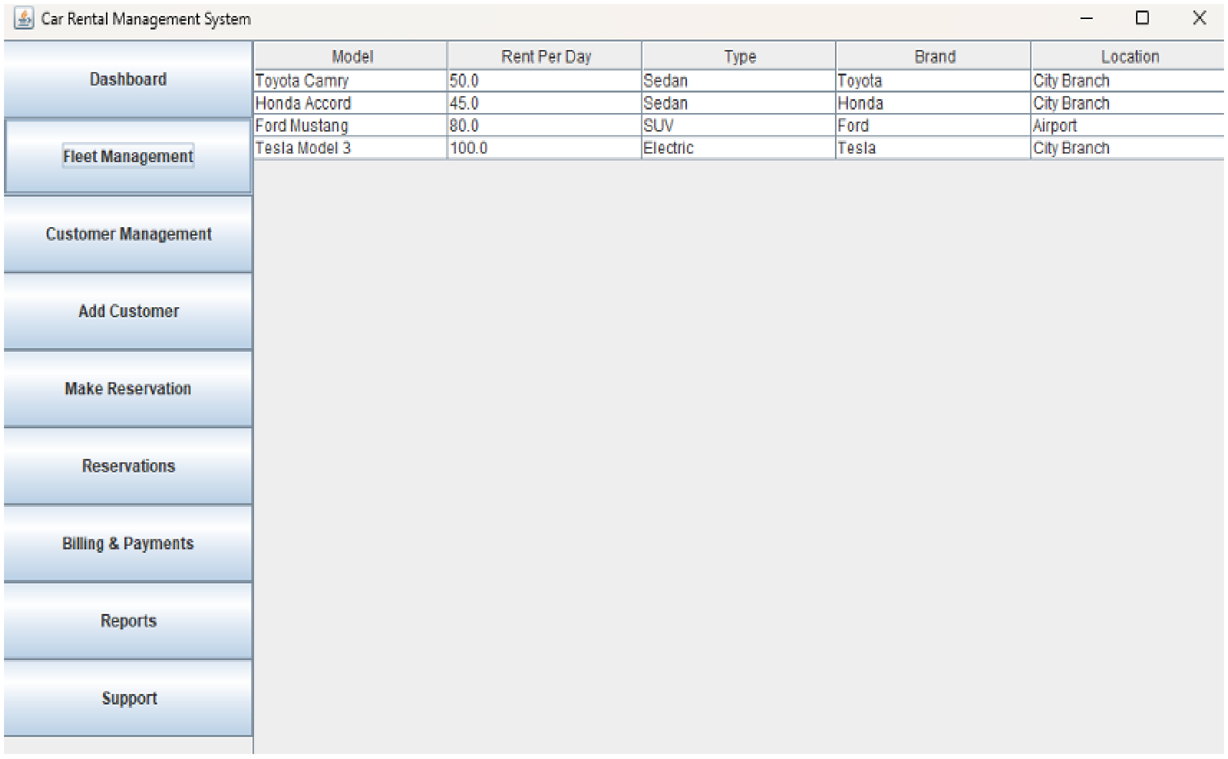
Demo:

Dashboard:



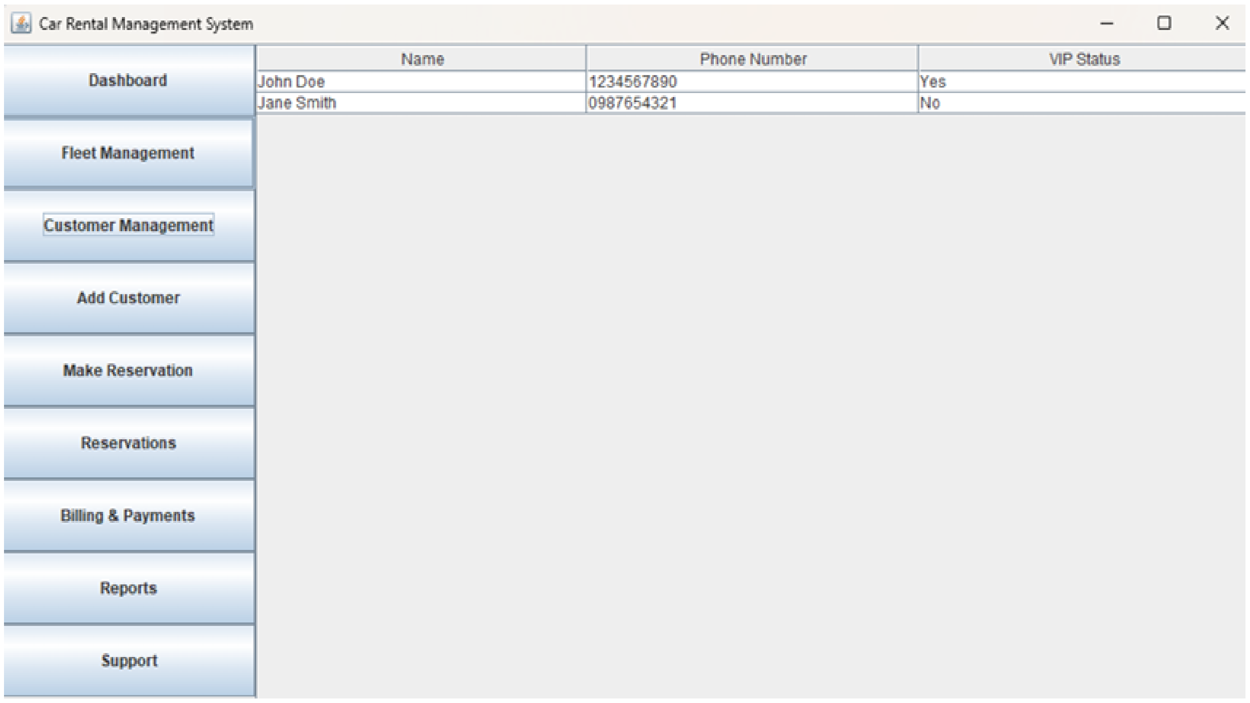
Category	Value
Total Cars	4
Available Cars	4
Rented Cars	0
Total Reservations	0

Fleet management:



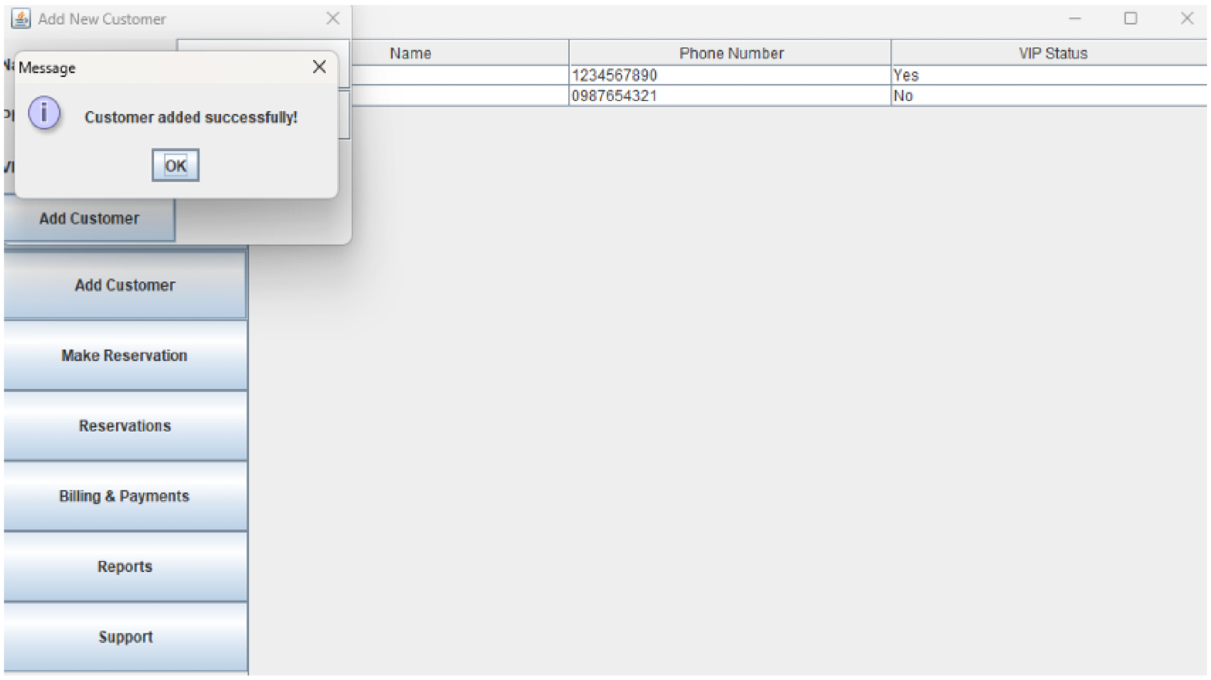
Model	Rent Per Day	Type	Brand	Location
Toyota Camry	50.0	Sedan	Toyota	City Branch
Honda Accord	45.0	Sedan	Honda	City Branch
Ford Mustang	80.0	SUV	Ford	Airport
Tesla Model 3	100.0	Electric	Tesla	City Branch

Customer management:



Name	Phone Number	VIP Status
John Doe	1234567890	Yes
Jane Smith	0987654321	No

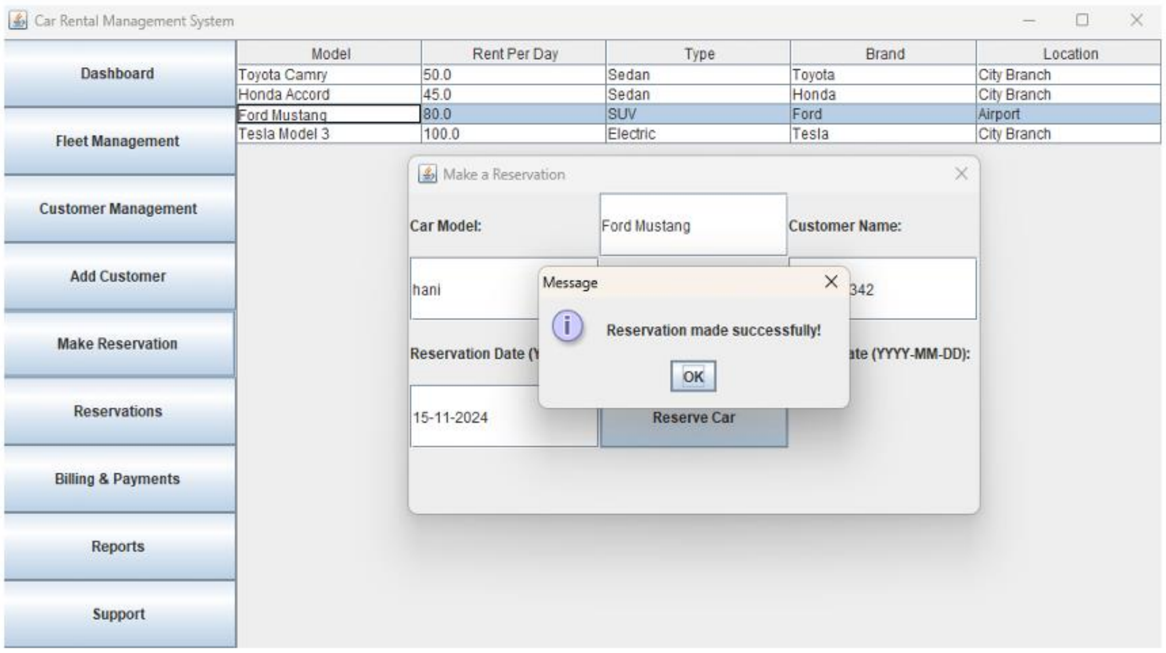
Add customer:



Customer added successfully!

Name	Phone Number	VIP Status
	1234567890	Yes
	0987654321	No

Mark Reservation:



Model	Rent Per Day	Type	Brand	Location
Toyota Camry	50.0	Sedan	Toyota	City Branch
Honda Accord	45.0	Sedan	Honda	City Branch
Ford Mustang	80.0	SUV	Ford	Airport
Tesla Model 3	100.0	Electric	Tesla	City Branch

Make a Reservation

Car Model: Ford Mustang

Customer Name: hani

342

Reservation Date (Y

ate (YYYY-MM-DD):

15-11-2024

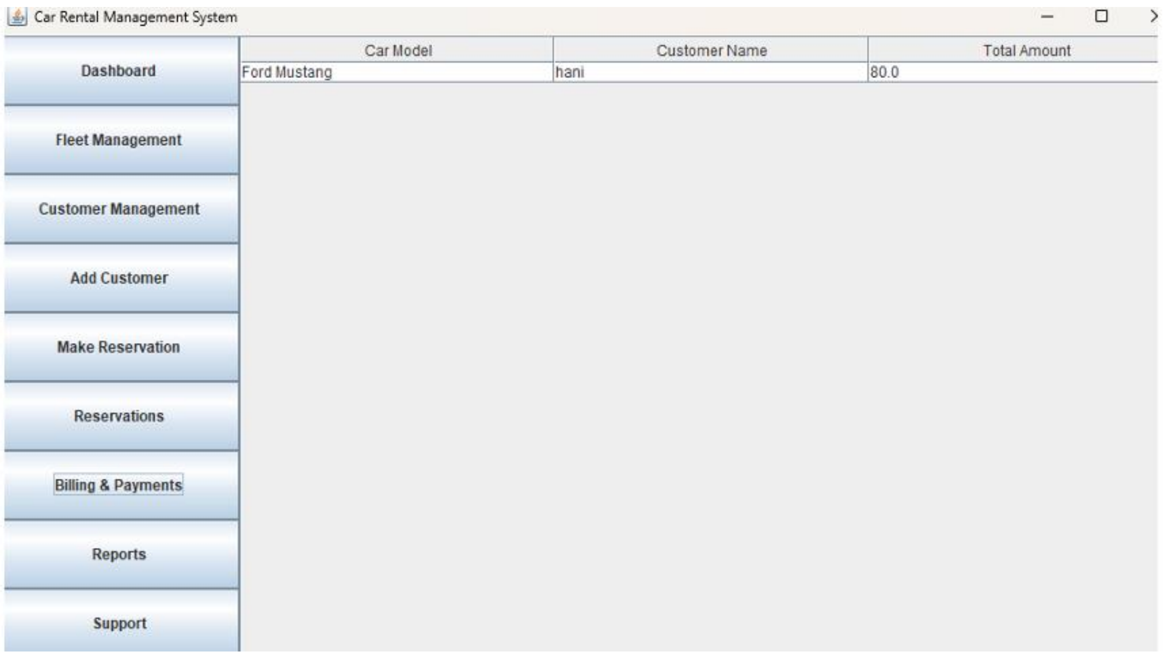
Reserve Car

Message

Reservation made successfully!

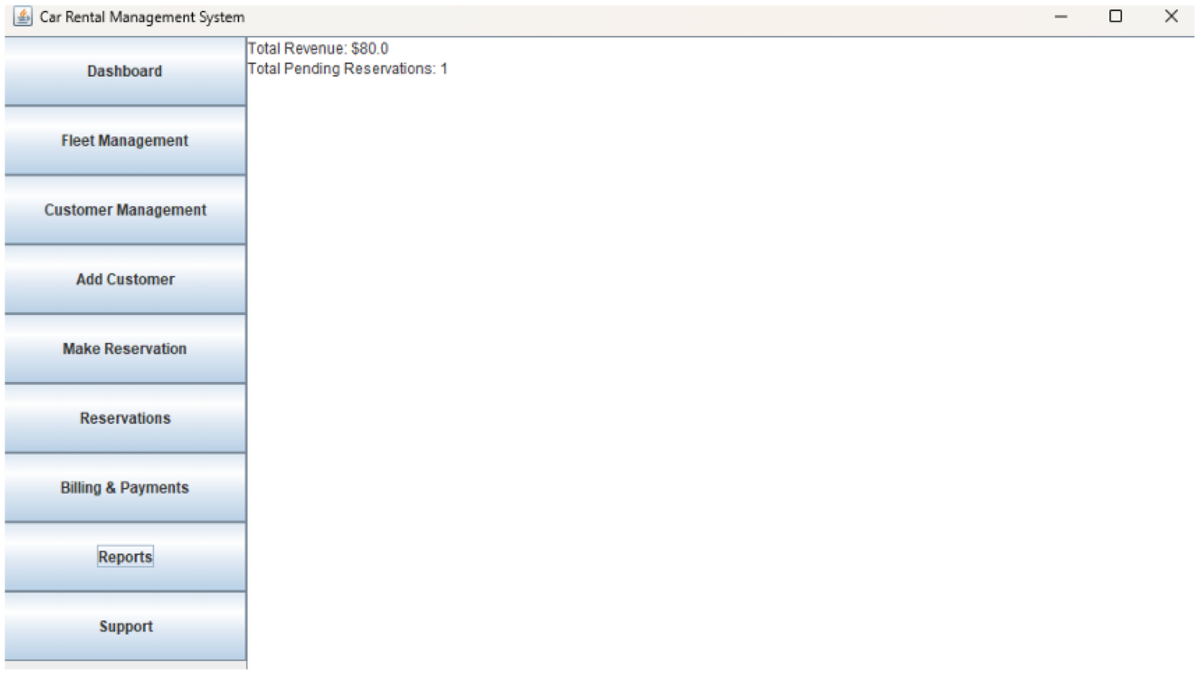
OK

Billing & Payments:



Car Model	Customer Name	Total Amount
Ford Mustang	hani	80.0

Reports:



```
Total Revenue: $80.0
Total Pending Reservations: 1
```

Key Functionalities:

- **Dashboard:** Displays an overview of total cars, available cars, rented cars, and reservations.
- **Fleet Management:** List of cars with availability status and rental information.
- **Customer Management:** List of customers and their details, with options to edit or add new customers.
- **Add customer:** adds new customers.
- **Make Reservation:** Makes a reservation for the rental car.
- **Reservation:** Displays a list of all the reserved cars.
- **Billing & payments:** Shows the billing amount of the car.

Conclusion of Demonstration:

- Reiterate the simplicity and effectiveness of the application in handling the car renting management system.
- Encourage questions from the audience regarding the functionality and technical implementation of the project.

5. Challenges and Solutions

Development Challenges

- <u>Data Persistence:</u> Managing persistent storage for data in the absence of a database.
- <u>Concurrency:</u> Ensuring thread safety when multiple users attempt to book simultaneously.

Solutions Implemented

- <u>File Handling for Persistence:</u> File-based storage is used to save and load data reliably.
- <u>Multithreading (if applicable):</u> Thread-safe reservations handling using Java's synchronized blocks or methods.

6. Conclusion & Future work

The Car Rental Management System successfully implements an end-to-end solution for managing car rentals. It is efficient and improves operational management by automating tasks related to customer booking, fleet management, and billing.

Future Enhancements:

1. <u>Database Integration:</u> Replace file handling with a SQL/NoSQL database for improved scalability.
2. <u>Mobile Application:</u> Develop a mobile version for wider accessibility.
3. <u>Enhanced Reporting:</u> Advanced reports, including customer behavior analytics, seasonal demand, etc.
4. <u>Payment Integration:</u> Integrate third-party payment services for seamless payment processing.