PROJECT REPORT

PROJECT TITLE: Car Rental System

SUBMITTED BY: Priyanka Kashyap

SAP ID: 590025638

Course: Btech. Computer Science & Engineering

Course Code: CSEG10321

INDEX

ABSTRACT

The Car Rental System is a software-based project designed to manage the process of renting cars in an efficient, secure, and user-friendly manner. This system allows the admin to add cars, display available cars, rent a car to a customer, and return rented vehicles seamlessly. The automation of manual work reduces errors, saves time, and increases productivity for rental companies. Through this system, users can quickly check car availability, calculate rental charges, and generate bills

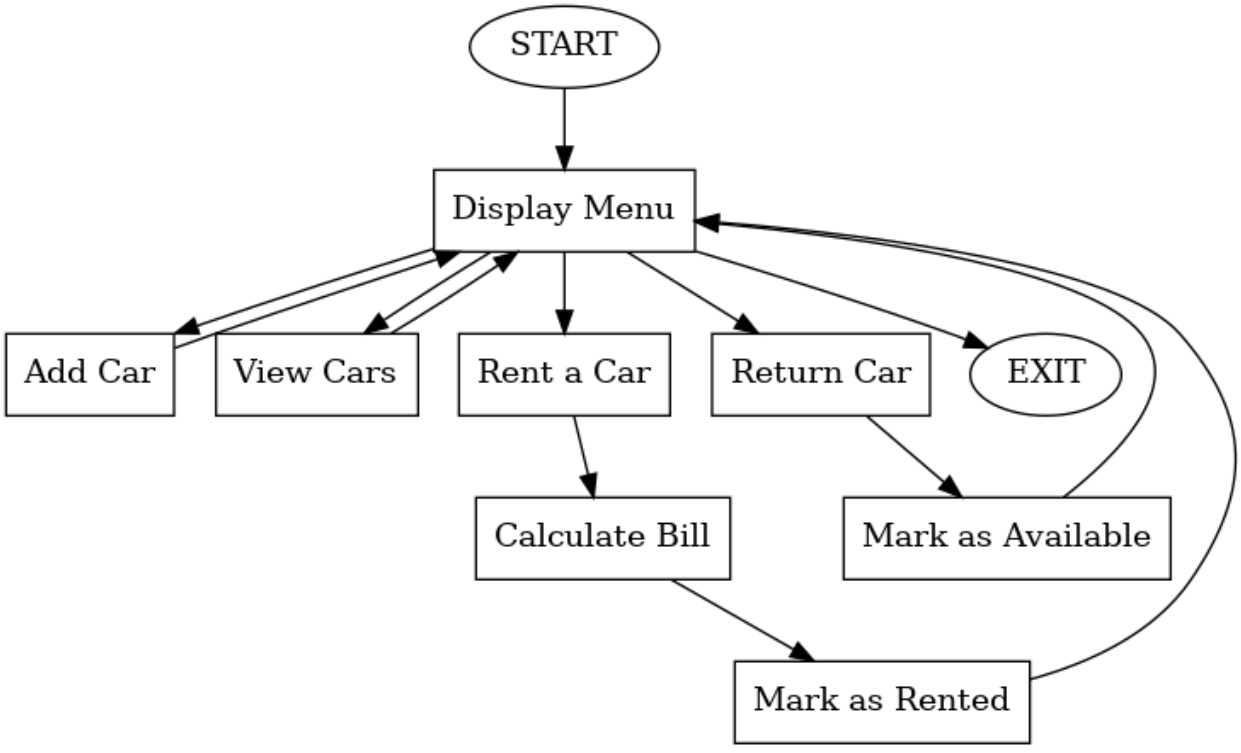
PROBLEM DEFINATION

Problem Definition – Car Rental System

Traditional car rental businesses often rely on manual paperwork, which leads to several issues such as inefficient booking, delays in retrieving information, lack of accurate records, chances of data loss, and calculation mistakes in billing. Customers face difficulty in checking the availability of cars, while administrators struggle with maintaining updated rental and return details.

The main problem is the absence of a simple, computerized system that can handle the complete rental process in an organized manner. Therefore, a Car Rental System is required to automate tasks like adding new cars, viewing available cars, renting cars, returning vehicles, and calculating rental charges accurately. The proposed system aims to reduce human error, save time, and make the entire process faster, reliable, and more user-friendly.

FLOWCHART :


START
Display Menu
Add Car
View Cars
Rent a Car
Return Car
EXIT
Calculate Bill
Mark as Available
Mark as Rented

IMPLEMENTATION DETAILS

The Car Rental System project is implemented using the C programming language, following a structured and menu-driven approach. The system uses various programming concepts such as structures, arrays, functions, loops, and conditional statements to provide functionalities like adding cars, viewing cars, renting a vehicle, and returning it.

A structure (struct) is used to store the details of each car, including Car ID, Car Name, Seating Capacity, Rent Per Day, and Availability Status. An array of structures maintains the database of cars within the program. The user interacts with the system through a menu-based interface inside an infinite loop, allowing multiple operations during the same execution.

Separate functions are created for each task:

addCar() – to insert a new car into the system

viewCars() – to display all cars with details

rentCar() – checks availability, enters days, calculates bill, and marks car rented

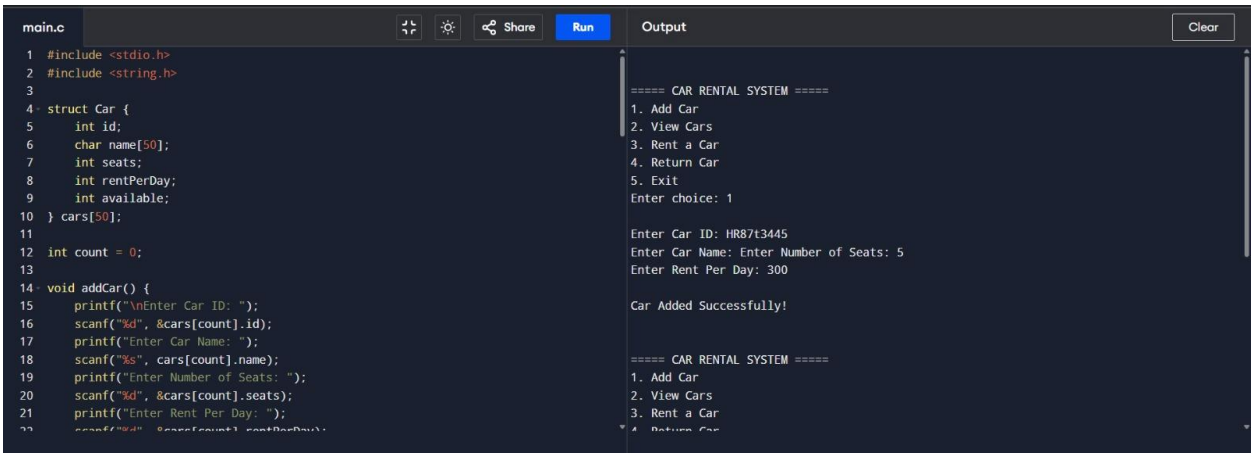
returnCar() – updates the status back to available

Conditional statements ensure that the car cannot be rented if already booked. The system displays the output clearly on the console, making it simple and user-friendly for new learners and rental agencies.

Overall, the implementation demonstrates how basic programming in C can solve real-life management challenges by automating record handling, reducing manual errors, improving accuracy, and increasing efficiency in the rental service operations.

TESTING AND RESULTS

The code was test through various scenarios inputting valid and invalid enteries



main.c

```c
#include <stdio.h>
#include <string.h>

struct Car {
    int id;
    char name[50];
    int seats;
    int rentPerDay;
    int available;
} cars[50];

int count = 0;

void addCar() {
    printf("\nEnter Car ID: ");
    scanf("%d", &cars[count].id);
    printf("Enter Car Name: ");
    scanf("%s", cars[count].name);
    printf("Enter Number of Seats: ");
    scanf("%d", &cars[count].seats);
    printf("Enter Rent Per Day: ");
```

Output

```
1. Add Car
2. View Cars
3. Rent a Car
4. Return Car
5. Exit
Enter choice: 2

--- List of Cars ---
ID: 0 | Name: HR87t3445 | Seats: 5 | Rent/day: 300 | Available


===== CAR RENTAL SYSTEM =====
1. Add Car
2. View Cars
3. Rent a Car
4. Return Car
5. Exit
Enter choice: 5
Thank you!

=== Code Execution Successful ===
```

APPLICATIONS

The Car Rental System has a wide range of real-life applications across various industries and business scales. Some major applications include:

1 Car Rental Companies

Used to manage vehicle bookings, track availability, generate bills, and maintain customer records.

2 Travel Agencies

Agencies offering tourism packages use such systems to provide cars for airport pickups, sightseeing, and tours.

3 Corporate Transportation Services

Companies rent vehicles for employee travel, client meetings, and logistics support.

4 Self-Drive Car Services

Platforms like ZoomCar, Ola Drive, and Revv use rental software for on-demand bookings.

5 Educational Demonstration / Learning Project

Ideal project for engineering and computer science students to understand real-world software development concepts.

6 Taxi and Fleet Management

Helps track multiple vehicles, drivers, fuel usage, and maintenance schedules.

7️⃣ Government & Public Transport Departments

Used for managing rental buses, jeeps, and official travel services.

.

CONCLUSION

The Car Rental System successfully demonstrates how computer programming can simplify and enhance real-world business operations. By automating the process of adding cars, maintaining records, renting vehicles, calculating bills, and updating their availability, the system eliminates the need for manual paperwork and reduces human errors. It provides a faster, more accurate, and efficient way to manage car rental services.

This project highlights the practical application of the C programming language through the use of structures, loops, and conditional statements in creating a user-friendly management solution. The system not only improves service

quality for customers but also supports rental companies in organizing data, saving time, and improving workflow.

Overall, the project achieves its objective of offering a cost-effective and reliable solution that enhances productivity and demonstrates how basic programming concepts can solve real-life business problems effectively

APPENDIX

```
#include <stdio.h>
#include <string.h>

struct Car {
   int id;
   char name[50];
   int seats;
   int rentPerDay;
```

```
    int available;
} cars[50];

int count = 0;

void addCar() {
    printf("\nEnter Car ID: ");
    scanf("%d", &cars[count].id);
    printf("Enter Car Name: ");
    scanf("%s", cars[count].name);
    printf("Enter Number of Seats: ");
    scanf("%d", &cars[count].seats);
    printf("Enter Rent Per Day: ");
    scanf("%d", &cars[count].rentPerDay);

    cars[count].available = 1;
    count++;

    printf("\nCar Added Successfully!\n");
}

void viewCars() {
    printf("\n--- List of Cars ---\n");
```

```
    for(int i = 0; i < count; i++) {
        printf("ID: %d | Name: %s | Seats: %d | Rent/day: %d | %s\n",
               cars[i].id, cars[i].name, cars[i].seats,
               cars[i].rentPerDay,
               cars[i].available ? "Available" : "Rented");
    }
}

void rentCar() {
    int id, days;
    printf("\nEnter Car ID to Rent: ");
    scanf("%d", &id);

    for(int i = 0; i < count; i++) {
        if(cars[i].id == id && cars[i].available == 1) {
            printf("Enter number of days: ");
            scanf("%d", &days);

            int total = days * cars[i].rentPerDay;
            printf("\nCar Rented Successfully!");
            printf("\nBill Amount: Rs %d\n", total);

            cars[i].available = 0;
```

```
            return;
        }
    }
    printf("\nCar Not Available OR Invalid ID\n");
}

void returnCar() {
    int id;
    printf("\nEnter Car ID to Return: ");
    scanf("%d", &id);

    for(int i = 0; i < count; i++) {
        if(cars[i].id == id && cars[i].available == 0) {
            cars[i].available = 1;
            printf("\nCar Returned Successfully!\n");
            return;
        }
    }
    printf("\nInvalid Car ID!\n");
}

int main() {
    int choice;
```

```
    while(1) {
        printf("\n\n===== CAR RENTAL SYSTEM =====");
        printf("\n1. Add Car");
        printf("\n2. View Cars");
        printf("\n3. Rent a Car");
        printf("\n4. Return Car");
        printf("\n5. Exit");
        printf("\nEnter choice: ");
        scanf("%d", &choice);

        switch(choice) {
            case 1: addCar(); break;
            case 2: viewCars(); break;
            case 3: rentCar(); break;
            case 4: returnCar(); break;
            case 5: printf("Thank you!"); return 0;
            default: printf("Invalid Choice!");
        }
    }
}
```

THANK YOU