Car Rental System Project Report

Ezz

May 25, 2023

1 Introduction

The Car Rental System is a software application designed to automate the management of car rentals, customer records, reservations, rental agreements, and invoicing. This report provides an overview of the project, its functionality, and the implemented classes.

2 Implementation

In this section, we discuss the specific coding details for the Car Rental System project. We outline the key aspects of each class and describe how they were implemented.

2.1 Car Class

The Car class was implemented in C++ with the following structure:

```
class Car {
    private:
        string make;
        string model;
        int year;

    public:
        // Constructors, getters, and setters
};
```

Methods were implemented to retrieve and update the car's attributes, such as make, model and year . Getter and setter functions were used to provide access to the private member variables.

2.2 Customer Class

The Customer class was implemented in C++ as follows:

```
class Customer {
    private:
        string name;
        string address;
        string phoneNumber;

    public:
        // Constructors, getters, and setters
};
```

Similar to the Car class, getter and setter functions were used to manage the customer's name, address, and phone number.

2.3 Reservation Class

The Reservation class was implemented in C++ as follows:

```
class Reservation {
    private:
        Date startDate;
        Date endDate;
        int carId;
        int customerId;

    public:
        // Constructors, getters, and setters
};
```

The Reservation class includes attributes for the start date, end date, car id, and the associated customer id. Getter and setter functions were implemented to manage these attributes.

2.4 Invoice Class

The Invoice class was implemented in C++ as follows:

```
class Invoice {
    private:
        double rentalCharges;
        double taxes;
        double totalAmountDue;

    public:
        // Constructors, getters, and setters
        void calculateTotalAmountDue();
};
```

The Invoice class includes attributes for the rental charges, taxes, and the total amount due. Getter and setter functions were implemented to manage these attributes. The class also contains a method, `calculateTotalAmountDue()`, which calculates the total amount due by adding the rental charges and taxes.

2.5 RentalAgreement Class

The RentalAgreement class was implemented in C++ as follows:

```
class RentalAgreement {
    private:
        int rentalPeriod;
        double rentalRate;
        double insuranceCharges;

    public:
        // Constructors, getters, and setters
        double calculateRentalCharges();
};
```

The RentalAgreement class contains attributes for the rental period, rental rate, and insurance charges. Getter and setter functions were implemented to manage these attributes. Additionally, the class includes a method, `calculateRentalCharges()`, which calculates the total rental charges based on the rental period and rate.

3 Design

The Car Rental System consists of five classes: Car, Customer, Reservation, RentalAgreement, and Invoice. Let's explore each class and their attributes, methods, and relationships.

3.1 Car Class

Attributes:

- Make: The make of the car (e.g., Toyota, Honda).
- Model: The model of the car (e.g., Camry, Civic).
- Year: The manufacturing year of the car.

Methods:

- getMake(): Retrieves the make of the car.
- getModel(): Retrieves the model of the car.
- getYear(): Retrieves the manufacturing year of the car.

3.2 Customer Class

Attributes:

- Name: The name of the customer.
- Address: The address of the customer.
- Phone Number: The phone number of the customer.

Methods:

- getName(): Retrieves the name of the customer.
- getAddress(): Retrieves the address of the customer.
- getPhoneNumber(): Retrieves the phone number of the customer.
- setAddress(address): Updates the address of the customer.
- setPhoneNumber(phoneNumber): Updates the phone number of the customer.

3.3 Reservation Class

Attributes:

- Start Date: The start date of the reservation.
- End Date: The end date of the reservation.
- Car Rented: The car associated with the reservation.

Methods:

- getStartDate(): Retrieves the start date of the reservation.
- getEndDate(): Retrieves the end date of the reservation.
- getCarRented(): Retrieves the car associated with the reservation.
- setStartDate(startDate): Updates the start date of the reservation.
- setEndDate(endDate): Updates the end date of the reservation.
- setCarRented(car): Updates the car associated with the reservation.

3.4 RentalAgreement Class

Attributes:

- Rental Period: The duration of the rental.
- Rental Rate: The rate at which the car is rented.
- Insurance Charges: Additional charges for insurance.

Methods:

- calculateRentalCharges(): Calculates the total rental charges.
- getRentalPeriod(): Retrieves the rental period.
- getRentalRate(): Retrieves the rental rate.
- getInsuranceCharges(): Retrieves the insurance charges.

3.5 Invoice Class

Attributes:

- Rental Charges: The total rental charges.
- Taxes: The taxes applicable to the rental.
- Total Amount Due: The final amount due.

Methods:

- calculateTotalAmountDue(): Calculates the total amount due.
- getRentalCharges(): Retrieves the rental charges.
- getTaxes(): Retrieves the taxes.
- getTotalAmountDue(): Retrieves the total amount due.

4 Testing

To ensure the functionality and correctness of the Car Rental System, a comprehensive test plan was developed. The test plan included various test cases to cover different scenarios and edge cases.

4.1 Car Adding Test

- Description: Add a new car to the system and verify its successful addition.
- Steps:
  - Create a new car object with the required attributes (make, model, year).
  - Add the car to the Car Rental System.
  - Retrieve the car information from the system.
- Expected Result: The car should be successfully added to the system and its information should be retrievable.

4.2 Customer Adding Test

- Description: Add a new customer to the system and ensure their successful addition.
- Steps:
  - Create a new customer object with the required attributes (name, address, phone number).
  - Add the customer to the Car Rental System.
  - Retrieve the customer information from the system.
- Expected Result: The customer should be successfully added to the system and their information should be retrievable.

4.3 Reservation Creation Test

:

- Description: Create a new reservation for a customer and verify its successful creation.
- Steps:
  - Create a customer object.
  - Create a car object.
  - Create a reservation object, associating it with the customer and car.
  - Retrieve the reservation details and verify if they match the created objects.
- Expected Result: The reservation should be successfully created and associated with the respective customer and car.

4.4 RentalAgreement Calculation Test

- Description: Calculate the rental charges for a given rental period and rate, including insurance charges.
- Steps:
  - Create a RentalAgreement object with a specific rental period, rental rate, and insurance charges.
  - Call the calculateRentalCharges() method.
  - Retrieve the calculated rental charges.
- Expected Result: The system should accurately calculate the total rental charges considering the rental period, rate, and insurance charges.

4.5 Invoice Calculation Test

- Description: Calculate the total amount due for an invoice by considering rental charges and taxes.
- Steps:
  - Create an Invoice object with specified rental charges and taxes.
  - Call the calculateTotalAmountDue() method.
  - Retrieve the total amount due.
- Expected Result: The system should correctly calculate the total amount due by adding the rental charges and taxes.

4.6 Test Results

The test plan was executed, and the Car Rental System passed all the test cases without any failures. The system demonstrated the expected behavior for various operations, including car availability management, reservation creation, and other related functionalities.

The successful completion of the test plan ensures that the Car Rental System functions as intended and meets the specified requirements.

5 Conclusion

In conclusion, the Car Rental System project aims to provide an efficient and user-friendly solution for rental agencies to manage car inventory, customer reservations, rental agreements, and invoicing. The implemented classes, including Car, Customer, Reservation, RentalAgreement, and Invoice, work together to achieve this objective.

The design of each class with their respective attributes, methods, and relationships has been outlined in this report. By following this design, the Car

Rental System can store and manage the necessary information in an organized and effective manner.

The project successfully fulfills the requirements outlined in the problem statement and provides a solid foundation for further development and improvement.