CSC–102 Programming Fundamentals Semester Project

Car Rental Management System

Section : SE

Submitted by:

Name : Muhammad Abdullah Feroz

Roll Number : 2023-SE-03

Submitted to:

Prof. Dr. Hafiz Muhammad Shehzad

Ms. Mariyam Manzoor

Dated: 19th December 2023

Department of Computer Science

University of Engineering and Technology Lahore, New Campus

Title:

Car Rental Management System Documentation

1. Introduction

- The program is a functional backend for a Car Rental Management System App created as the final project for first semester.
- The CRMS has the ability to distinguish between user as per their roles of admin and customer by logging in using the username and password assigned to them.
- Each role has its own functionalities.

2. System Overview

- The CRMS project covers all the topics studied in the Programming fundamentals course in the first semester.
- Structures are used to store information about users and cars in a group. Dynamic arrays are used for efficient memory allocation and changing the size on runtime.

3. Features

3.1 Admin Features

- **Admin Login**
  - Allows the admin to log in using a username and password.
- **Admin Menu**
  - Provides options for managing car inventory, customers, and viewing reports.

3.2 Customer Features

- **Customer Login**
  - Allows customers to log in using their username and password.

- **Customer Menu**
    - Options for renting a car, returning a car, viewing the car inventory, and generating reports.

3.3 Main Features

- **Car Rental**
    - Functionality for customers to rent a car, considering availability and generating bills.
- **Car Return**
    - Allows customers to return a rented car, updating the inventory and generating reports.
- **Car Inventory Management**
    - Admin can add or remove cars from the inventory.

4. File Handling

- Data is read from files and stored in program memory when the program starts and during runtime all the data is updated in the program memory resulting in increase in efficiency by reducing time to access and read/write data in files.
- Similarly, at the end, before exiting the program all data in the program memory is dumped back to the files in result saving all the activity performed during the execution.

5. User Interface

- The program interacts with the user through a menu-driven interface over a console based system.
- Console-based input offers several advantages, making it a preferable choice in various scenarios. The resource efficiency of console applications,

along with their suitability for scripting and automation tasks, makes them ideal for scenarios where a lightweight, text-based interface is sufficient. Additionally, console-based input is often favored for programming practice, as it provides a straightforward environment for beginners to grasp fundamental concepts without the complexities associated with graphical user interfaces.

6. Functions

6.1 Admin Functions

- **Manage Car Inventory**
    - Allows the admin to add or remove cars from the inventory.
- **Manage Customers**
    - Options to view, add, update or remove customer accounts.
- **View Customer Report**
    - Provides admin with the ability to view reports of customer transactions.
- **View Cars Report**
    - Provide admin with the ability to view report of each car individually showing the stats like times rented and total income generated by the car.

6.2 Customer Functions

- **Rent a Car**
    - Allows customers to rent a car based on availability.
- **Return a Car**

    - Allows customers to return a rented car, generating reports. Upon returning the information of rental service will also be uploaded in the history log, and the car will undergo a maintenance service.

- **Generate Reports**

    - Options for generating rental and transaction history reports.

    - Rental report will give provide customer with the renting bill of car rented currently, if any.

    - Transaction History reports will show customer that how many cars he has rented in the past along with the details of rental.

7. Function Prototypes

<u>General Functions</u>

1. **void refreshDisplay();**
    - ➢ Refreshes the screen clearing the previous output and shows program title.
2. **void adminLogin(user*&, short& size, car*&, short& carSize);**
    - ➢ Validates if the user is admin and then calls the ***adminMenu***
3. **void customerLogin(user*, short size, car*, short carSize);**
    - ➢ Validates whether a user is registered as customer and calls ***customerMenu***
4. **user* loadLoginInfo(short& size);**
    - ➢ Loads data of users from files on program startup and loads them into memory.
5. **void writeLoginInfo(user*, short size);**
    - ➢ Dumps the data of users back into the file when closing the program.
6. **void loadUserLog(user& customer, short& cSize, string filename);**
    - ➢ Loads user log history from files into dynamic array when a user logs in.
7. **void writeUserLog(user& customer, short size, string filename);**
    - ➢ Dumps data from array back to the file when user logs out.
8. **car* loadCarInventory(short& size);**
    - ➢ Reads data of cars from files on program startup and loads them into

memory.

9. **void writeCarInventory(car*, short size);**

- Dumps the data of cars back into the file when closing the program.

10. **void displayCar(car* inventory, short inventorySize);**

- Displays all the cars and their status

Customer Functions

11. **void customerMenu(user& customer, car* carInventory, short invntorySize);**

- Provides a menu with all customer controls and call the required function as per user choice.

12. **void rentCar(user& customer, short& logSize, car* inventory, short inventroySize);**

- Displays cars and allows user to rent one of them and restricts user from renting any other car.

13. **void returnCar(user& customer, short& logSize, car* inventory, short inventorySize);**

- Called when user is returning the rented car and update the user history.

14. **void generateReport(user customer, short logSize);**

- Provides a menu to choose whether user wants to view current bill or past report.

15. **void generatePreviousReport(user customer, short logSize);**

- Generates the history report of the user.

16. **void generateBill(user customer);**

- Displays the bill of currently rented car if any.

Admin Functions

17. **void adminMenu(user*&, short& size, car*&, short& carSize);**

- Provides a menu with all admin controls and call the required function as per user choice.

18. **void manageCustomers(user*& users, short& userSize);**

- Provides admin with the functionalities to view, add, remove, and update customer data.

19. **void viewCustomer(user* users, short userSize);**

- ➢ Displays the registered customers and their data.

20. **void addCustomer(user*& users, short& userSize);**

- ➢ Adds customer by updating the array dynamically.

21. **void removeCustomer(user*& users, short& userSize);**

- ➢ Removes customer by updating the array dynamically.

22. **void updateCustomers(user* users, short userSize);**

- ➢ Allows the admin to choose a customer for updating data.

23. **void updatecustomerData(user& customer);**

- ➢ Updates data of the customer selected by the admin.

24. **void viewCustomerReport(user* users, short userSize);**

- ➢ It shows the report of a selected user.

25. **void manageCar(car*& carInventory, short& inventorySize);**

- ➢ Provides admin with the functionalities to view, add and remove cars.

26. **void addCar(car*& carInventory, short& size);**

- ➢ Adds a new car by updating the car array.

27. **void removeCar(car*& carInventory, short& size);**

- ➢ Removes a car by updating the car array.

28. **void viewCarReport(car* carInventory, short size);**

- ➢ Allows the user to select a car to view its usage report.

29. **void generateCarReport(car vehicle);**

- ➢ Generates the report of a selected car.

8. File Structure

- The files are also stored in a proper manageable structure. All the files are present in a folder named “textFiles” and another sub folder is present to store the inventory status files of each customer.

9. Usage Instructions

- The user is guided along with the help of a user friendly menu-driven interface that will validate the choices entered by the user and will also handle any exceptions thrown by the invalid input from user side.

10. Conclusion

- This project at the end of semester allows the students to revise and practice all the concepts learned throughout the course and apply them to solve a simulation of a real world problem.
- Moreover this project teaches students to grasp extra-curricular techniques of software engineering and self-learning by enabling them to look through the resources available all over the internet in order to solve their problems efficiently.