NED UNIVERSITY OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF COMPUTER & INFORMATION SYSTEMS ENGINEERING

COMPLEX ENGINEERING PROBLEM

COURSE CODE: CS-116 **COURSE TITLE**: OBJECT OREINTED PROGRAMMING

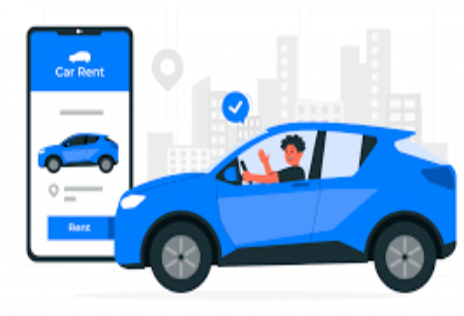
SPRING SEMESTER 2025

TEAM PROJECT TITLE: ONLINE CAR RENTAL SYSTEM

GROUP MEMBERS:

ZAINAB (CS-24051)

BATOOL ZEHRA (CS-24056)



ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our course instructor Miss Ramish Fatima & Miss Fauzia Yasir along with our lab instructor Miss Tehreem Khan for their guidance, support and cooperation. Their guidance helped us to create this successful project of the car rental management system. This project has been an enriching experience, allowing us to apply theoretical knowledge in a practical and meaningful way.

ABSTRACT

This project presents a python-based car rental management system developed with Tkinter for the GUI and CSV files for data storage. It enables users to register, log in, rent and return cars, and submit feedback while providing administrators tools to manage vehicles, view rentals and monitor customer activity. The system applies core object-oriented principles and simulates real-world rental operations, including payment validation and late return handling. It offers an efficient alternative to traditional rental systems.

PROBLEM DESCRIPTION:

Car rental services are increasingly important in urban areas where people need temporary transportation without the burden of ownership. However, many car rental processes are still handled manually or via basic web platforms that lack personalization and administrative control. This project addresses the gap by creating a Graphical User Interface – based car rental system in python, which provide complete digital solution for both customers and administrators.

Distinguishing features and OOP Concepts Implemented:

Object-Oriented Features:

- **Inheritance**:
  CSVstoreable is an abstract base class inherited by user, car, rental and feedback.
- **Association**:
  User aggregates rental objects and car is associated with rental.
- **Method overriding:**
  Subclasses like user and car override the save_to_csv method from CSVstorable.
- **Abstract classes:**
  CSVstoreable defines an abstract save save_to_csv method.
- **Exception handling:**
  Input validations for balance, rental days, and payment details.

Additional Features:

- GUI interface using Tkinter.
- Payment system with credit/debit card validation and cash options.
- Feedback system for users to review rented cars.
- Late return fee calculation.

Flow of the Program:

1. Program start:
   Main GUI window is created
   CarRentalApp class is initialized
   User sees:
   Welcome message, login & register buttons
2. User Action: register or login
   If register:
   User enters:
   - Username, password, name address & balance

   Register() validates and saves user to user.csv.

If login:

- User enter credentials
- User.authenticate() checks credentials from user.csv
- If valid, redirected to dashboard.

3. Dashboard Options (based on role):

If customer:

- View available cars
- Rent a car
- Return a car
- View rental history
- Give feedback

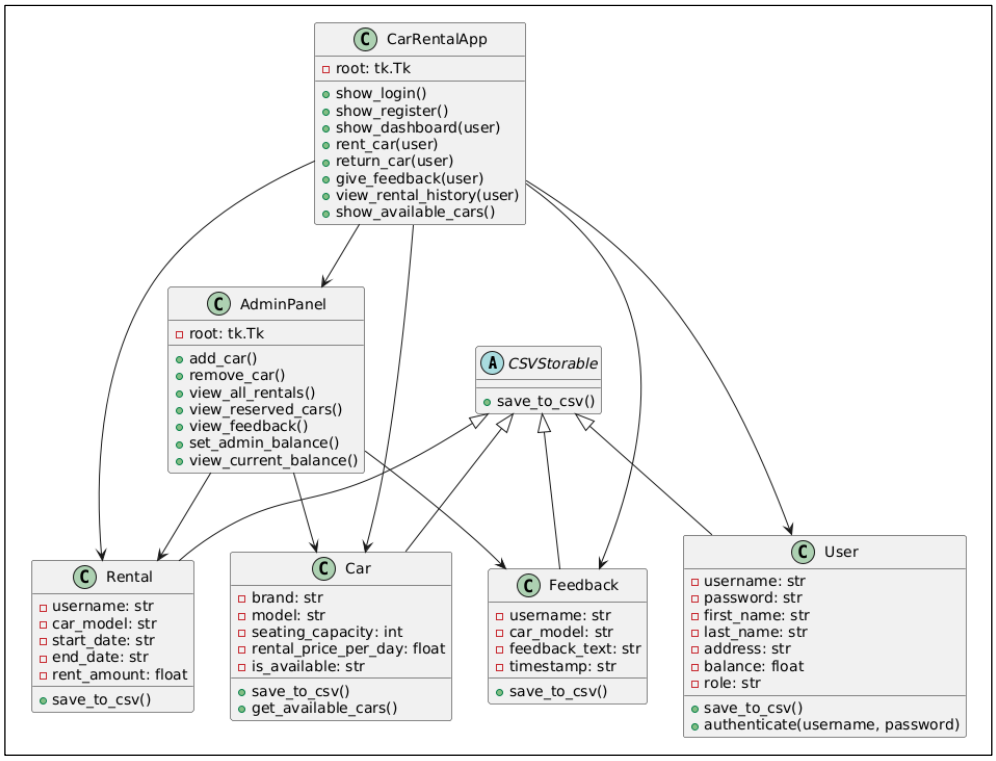
If Admin:

- Add/remove cars
- View all customer rentals
- View reserved cars
- View feedback
- Set rental balance
- View rental history
- Can also perform all customer tasks

4. Renting a car:
   - Customer selects an available car
   - Enters payment method
   - Car marked as unavailable in car.csv
   - Rentals saved in rentals.csv
   - User balance updated in user.csv
5. Returning a car:
   - Customer selects a car to return
   - System checks if its late & deduct extra fees if needed
   - Rental end date updated in rentals.csv
   - Car marked as available again in cars.csv
6. Giving feedback:
   - User selects rented car model
   - Enters feedback
   - Feedback saved in feedback.csv with timestamp
7. Admin-specific functions:
   - Add/remove cars updates cars.csv
   - View feedback reads feedback.csv
   - View rentals read rentals.csv
   - Reserved cars filtered where is_available == false

8. Program Ends:
    - When user closes GUI window or logs out

CLASS DIAGRAM:



Most Challenging Part Of This Project:

1. **GUI EVENT HANDLING:**
   Managing dynamic window transitions (switching between login, dashboard and admin panel).
2. **LATE FEE LOGIC**:
   Calculating extra charged based on datetime differences and updating user balances accordingly.

New Things Learned In Python:

- Tkinter for building interactive GUI applications.
- CSV file operations for persistent data storage.
- Datetime manipulations for rental periods and late fees

Future Expansions:

- **User Interface**: Graphical user interface can be enhanced with animations, themes, or can be switched to web-based interface
- **Real-time Notifications**: Email/SMS alerts for rental reminders
- **Hourly Rentals**: Giving edge to users for renting cars on hourly basis as well.

- **User authentication improvements**: It can be implemented by adding password encryption, email verification and adding forget password functionality

TEST CASE RUNS:

Manual testing was conducted for core features like car registration, user sign-up, return and admin panel. The system includes 6 classes: Car, User, Rentals, AdminPanel, CarRentalApp, and Feedback class.

TEST CASE: 01

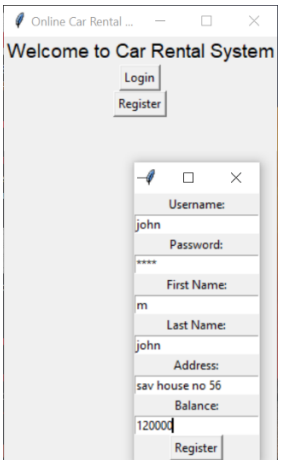
TEST DESCRIPTION:

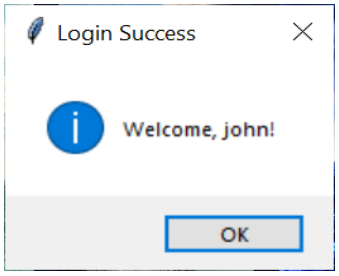
Under the consideration of user class we will check out the user registration in which the user will fill the form, inserting all mentioned credentials carefully and register itself by clicking over the register button.

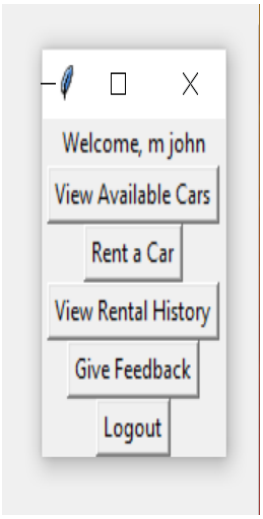
EXPECTED RESULT: user will be added to the system.

ACTUAL RESULT: Displaying a message of "Registration Successful!" ensuring the use has been registered.

STATUS: PASS







TEST CASE: 02:

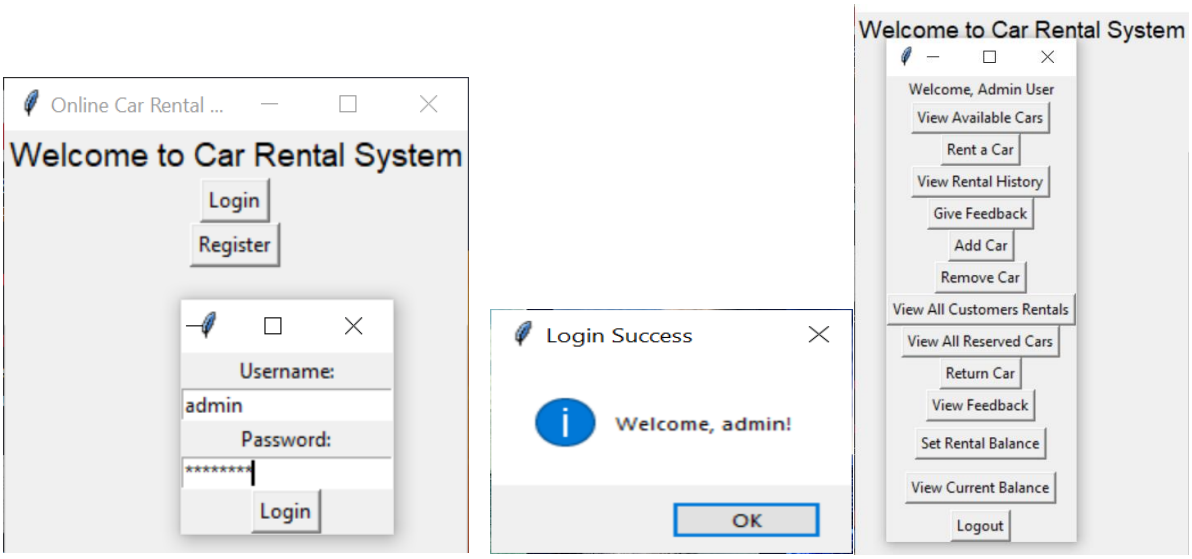
TEST DESCRIPTION:

For admin username and password are fixed and by entering those admin will get access to the admin panel and all its functions

EXPECTED RESULT: Admin will be added to the system

ACTUAL RESULT: Displaying the message of "welcome admin" ensuring that admin is added to the system.

STATUS: PASS



TEST CASE :03:

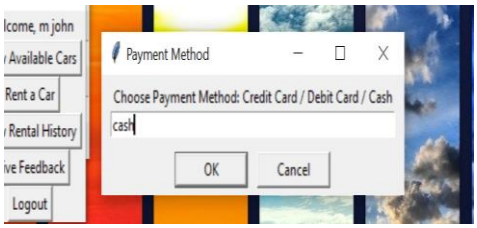
TEST DESCRIPTION:

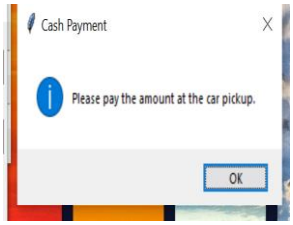
On considering rentals, registering as a user we will rent a car, inserting which model we want to rent and will check out if the car is rented or not from "car rental history"

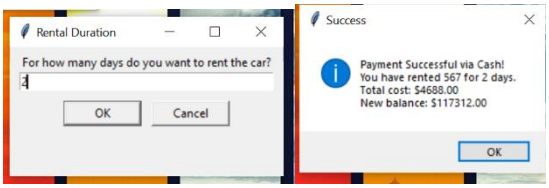
EXPECTED RESULT: Car marked as rented

ACTUAL RESULT: As we have rented the car model to which we want to rent another window opens highlighting which payment method user would like to choose, therefore, after responding to it another panel open asking user for how many days user want to rent the car, confronting all these credentials a message comes insuring that user has rented car for these days.

STATUS: PASS







INDIVIDUAL CONTRIBUTION:

Group members	Contributions
Zainab	○ Backend (maintaining return function including late returns, payment methods, rent function) ○ Documentation (including class diagram) ○ Quality assurance ○ File handling
Batool Zehra	○ Backend (code development) ○ File handling ○ Frontend (GUI interface) ○ Quality assurance

REFERNCES:

Python Official Documentation
https://docs.python.org/3/

Real Python (tutorials on GUI, file handling and more)
https://realpython.com/
Well-regarded tutorials and explanations on Python programming, GUI, file handling, and more.

W3Schools – Python CSV
https://www.w3schools.com/python/python_csv.asp

GeeksforGeeks – Python OOP Concepts
https://www.geeksforgeeks.org/python-oops-concepts/