Assignment 1

Sairam CHari

October 18, 2024

1 Rental Management system

I have written code to assist in the logistics of rental management that interacts with the user using an Interactive and user-friendly CLI. The code relies on multiple functions and classes working hand in glove. This approach allows for a higher level of modularity allowing for easy modification of the functionality of the code

2 Features included

- Vehicle renting
- Vehicle returning
- Listing available vehicles
- Listing Rented Vehicles
- Adding and removing vehicles
- Adding customers

Functions for the same are defined in the respective object classes, not in the rental manager (There is still some logic in the rental manager). This allows for modularity of the code and for the functions to be called for other logic without the need for a rental manager object, reducing it to just an UI element. (Enforcing the KISS Philosophy)

3 Obstacles faced

The code became long and complex, which caused issues maintaining the same function syntax across the code (passing customer instead of customerid as an argument, etc.). This was the main issue to solve while debugging. Each question in the assignment required me to change the functions already defined by me earlier which broke the flow and again, caused issues maintaining proper syntax.

4 Known issues

The code is satisfactory and fulfills the needs set forward by the assignment. However, a few improvements can be made (Given enough time).

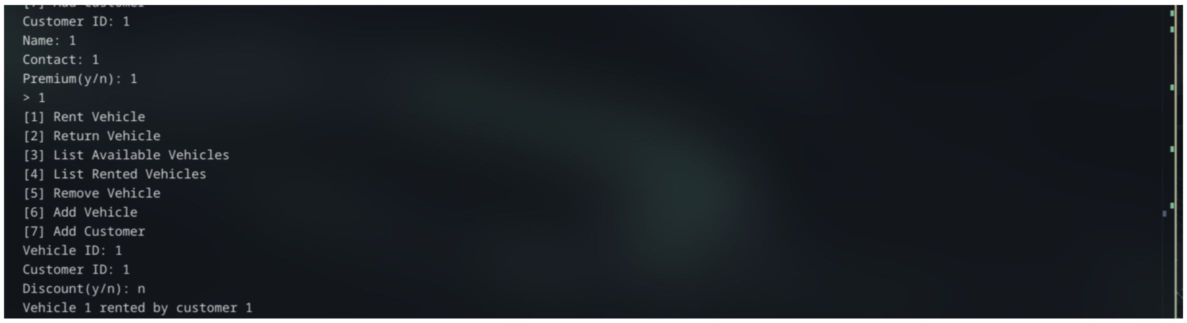
- Vehicle and customerid were user inputted. Ideally, it should autogenerate unless overridden
- Regular vs Premium customers could have been handled better. Instead of making two lists (and checking both in every function that required the same), Regular customers could have been assigned even IDs, and premium customers could have been assigned odd IDs eliminating a whole list and making maintaining the system easier.
- There is no redundancy. The data is not stored on a file and is lost when the program closes.

These "issues" do not handicap the program for the purpose its made for but improve the UX.

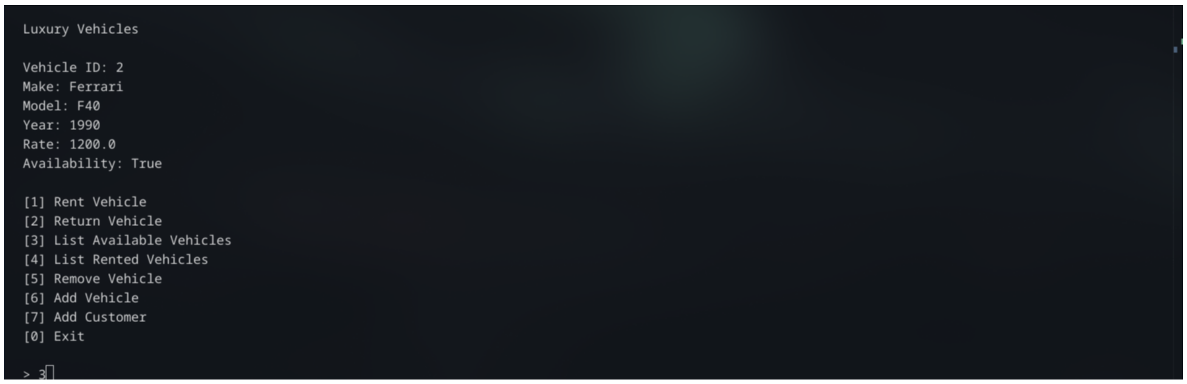
5 Conclusion

The code runs the required logic and gives well-fomatted outputs for all the required functionality asked in the assignment but requires polishing before being deployed in a live environment,

6 Images



```
Customer ID: 1
Name: 1
Contact: 1
Premium(y/n): 1
> 1
[1] Rent Vehicle
[2] Return Vehicle
[3] List Available Vehicles
[4] List Rented Vehicles
[5] Remove Vehicle
[6] Add Vehicle
[7] Add Customer
Vehicle ID: 1
Customer ID: 1
Discount(y/n): n
Vehicle 1 rented by customer 1
```

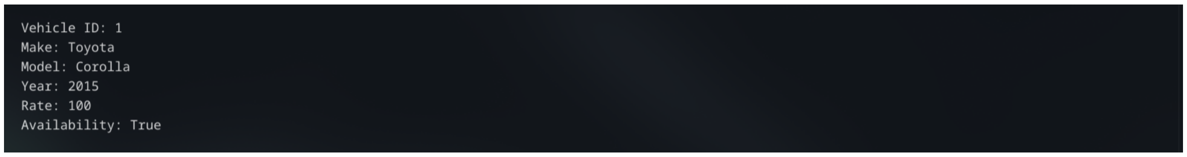


```
Luxury Vehicles

Vehicle ID: 2
Make: Ferrari
Model: F40
Year: 1990
Rate: 1200.0
Availability: True

[1] Rent Vehicle
[2] Return Vehicle
[3] List Available Vehicles
[4] List Rented Vehicles
[5] Remove Vehicle
[6] Add Vehicle
[7] Add Customer
[0] Exit

> 3
```



```
Vehicle ID: 1
Make: Toyota
Model: Corolla
Year: 2015
Rate: 100
Availability: True
```