

**PREMIUM DELUXE MOTORSPORT**

**Car Rental System - Report**

## I Introduction :

The car rental system is designed to automate the processes of renting and returning cars. The developed program is intended to create a car rental system using a concept in python called object oriented programming. The overview of the instructions, the rental shop contains several cars available for rent. All of the cars are classified into three types, Hatchback, Sedan, and SUVs. The rental prices depend upon the type of the car, the duration and if the customer is a VIP. The rents of cars less than a week : Hatchbacks - 30\$, Sedans - 50\$, SUVs - 100\$; The rents of cars over a week : Hatchbacks - 25\$, Sedans : 40\$, SUVs - 90\$; Rents for VIPs are : Hatchbacks - 20\$, Sedans - 35\$, SUVs - 80\$. To rent a car the user should give the name of the car and its type and at the end there should be a prompt stating the user's name and the car he rented and its overall price. When returning the car, the car must be returned back to the rental pool. The program should have at least two classes and the program has to perform transactions like rent and return and produce a bill.

Systems like these are used everywhere in today's day and age, everything that uses a database to every system that needs to conduct a transaction. It starts from small businesses to all the way up to companies that sell digital products themselves. We can make python to automate anything that mimics real life and programs like these make quality of life easier and quicker to arrive at targets and keep better track of our clients and where the money is moving and that can be inside a business or for personal use itself. The program that we are building focuses largely on making it interactive with the user and making it so that transactions are flawless for renting a car and returning a car. There are 6 processes in this car rental system. They are to Rent a car, Return a car, Display all the available cars, Display all the rented cars, Inquire about the prices and finally Exit.

## II Program Structure :

The structure of this program is pretty straight-forward, I've decided to go with only one class called the Car Rental System and define methods based on the requirements of the coursework's instructions. I have made a flow chart (Fig.1.) illustrating the overall idea in which the code was developed. The idea was to make it as interactive as possible hence we will mostly be taking inputs from the user and give him what he needs. There are 4 functions that are really necessary for a car rental system - Renting a car, Returning the rented car, displaying all the available cars, displaying the rented cars and lastly inquiring about prices. To do that first, we create the class and initialize it. Then we are creating the pool of cars available into a dictionary with keys and values where keys are the car's type and the values are the models available. Now, we are creating a dictionary for the prices of each type of car. If a car is rented for at least 6 days the prices are normal, if it is rented for more than 7 days the prices are

discounted and lastly if the customer is a VIP then the prices are discounted furthermore. Later that we are making a dictionary for rented cars and a list to store customer's names.

The first method is to display the cars available, to do that we are making a for loop which iterates through the car types first and we are making another for loop inside it to iterate through the models.

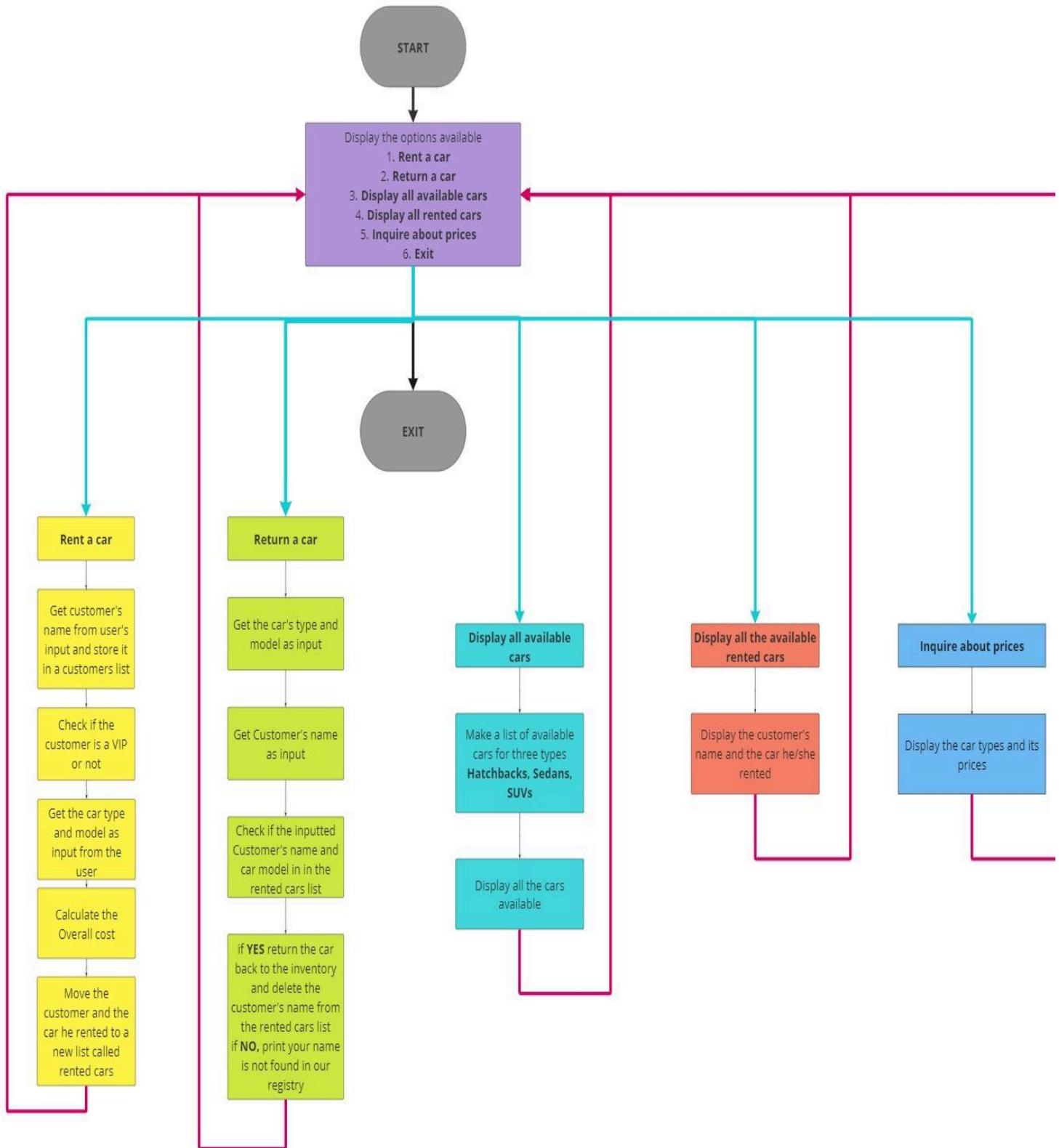
The second method is to rent a car, for that we are making five parameters. Here the main goal is to look out for these three things, the rental period, if the customer is a VIP or not and lastly, create a receipt after calculating the rent. Since we have assigned prices to a dictionary it is easier to use an if statement to assign the prices to the selected car. The calculation is done by multiplying the number of days to the pricing assigned.

The third method is to return a rented car. Here, we take in the customer's name, the car's model and type he rented as parameters and iterate through the rented cars dictionary to check if we could find a record of the customer renting that car. If there is one, we return the car back to the rental pool and if there is no such record, an invalid statement is printed.

And lastly, a method to display the rented cars, for that we are just going to open a for loop and iterate through the things available in the rented cars dictionary.

Now, we are creating a new class called customer's inquiry to inquire about the prices for each car type. Then we are defining a method to display all the prices of the hatchbacks, sedans, and SUVs. For that we use a for loop to iterate through the self.prices dictionary to give us the duration and its prices.

Now to the main program, we are assigning a variable called rental system to act as an instance to the class car rental system. Then we are making a while loop and asking the user to choose what to do with the rental system. For each request, their respective methods will be called and at the end there will be an exit option to break the while loop.



**Fig.1. The Flowchart for the car rental system**

### **III Application Development :**

To develop this program there were various python concepts used, some of them are: object oriented programming - which includes classes, objects, attributes, methods; lists, Dictionaries, nested dictionaries, for loops, while loops, fstrings, if-statements.

**Functions** allow you to reuse a specific block of code again and again with the use of arguments. Arguments are basically the values that get sent into the function.

( The syntax for functions are : **def function-name (Arguments):** )

**Object oriented programming** is a way of structuring a program in which we make a class which is a blueprint of something that consists of its attributes and its actions, which we call as methods. With classes we can create unique objects each of which have their own attributes and methods. To put in simpler terms an object is an instance of a class.

**Lists** are sequence of any data types which are separated by commas inside a square bracket. “[ ]”. Lists are mutable which means that we can change the values inside a list unlike its counterpart tuple where it is immutable. The values in a list have indices which start from 0 to n. An index value is the position of something inside a list.

**Dictionary** is a data type where you store values in key, value pairs. Dictionaries are also mutable and use curly braces “{ }”, the keys and values are separated by a colon. Dictionaries are also mutable and can have multiple dictionaries inside them which then makes it a nested dictionary.

**For Loop** is a statement that iterates over a string or lists or tuples or dictionaries. A working for loop must contain a variable and an iterable object like a list.

( The syntax for a for loop is : **for variable in list:** )

**While Loop** is a statement you use if you want to perform a task indefinitely until a condition is met. You have to include a break statement or else your code will run forever for all eternity.

( The syntax for while loop is : **while condition:** )

**Fstrings** are very similar to the print() but you can embed expressions inside your string literals.

**If-statements** only execute a block of code under it if a specific condition is met. You can use elif and else statements to give more conditions.

## IV Program Functionality:

After a lot of tinkering I can say that the program works flawlessly. To make it a bit interactive, the idea was to make the user input the car's type. So here are some of the outputs from the test runs that I did.

```
----- Welcome To Premium Deluxe Motorsport!!!-----
1. Rent a car
2. Return a rented car
3. Display available cars
4. Display rented cars
5. Inquire about car types and prices
6. Exit
Enter your choice (1-6):
```

Fig. 2. Welcome Message

In Fig. 2, is a welcome message and all the available services. We will be going through all of them but we will choose display cars first and then move to rent or return a car.

### 1. Display Available Cars :

```
Enter your choice (1-5): 3

Available Hatchbacks:
- Honda Civic
- Peugeot208
- Dacia Sandero
- Hyundai i10
- Volkswagen Polo
- Volkswagen Golf
- Ford Focus
- Toyota Corolla
- Skoda Octavia
- Kia Picanto
```

Fig. 3.

```
Available Sedans:
- Kia Rio
- Nissan Versa
- Honda Civic Si
- Hyundai Elantra N
- Mazda 3
- Volkswagen Jetta GLI
- Honda Accord
- Hyundai Sonata
- Kia K5
- Audi A3
- Audi S3
- Genesis G70
```

Fig. 4.

#### Available SUVs:

- Skoda Kodiaq
- Volvo XC40
- Audi Q5
- Volkswagen T-cross
- Kia EV6
- Audi Q4 e-tron
- Land Rover Defender
- BMW X3
- Nissan Juke
- Dacia Sandero Stepway

Fig. 5.

All the three screenshots ( i.e Fig. 3,4,5) are the cars available in the rental shop.

#### 2. Rent a car :

```
----- Welcome To Premium Deluxe Motorsport!!!-----
1. Rent a car
2. Return a rented car
3. Display available cars
4. Display rented cars
5. Exit
Enter your choice (1-5): 1
Enter your name: jack ingof
Enter car type (Hatchbacks, Sedans, SUVs): Hatchbacks
Enter car model: Honda Civic
Enter rental duration (in days): 9
Are you a VIP member? (yes/no): no

Renting Honda Civic (Hatchbacks) for 9 days to jack ingof.
Total cost: $225
```

Fig. 6.

In Fig. 6. We are renting a hatchback - Honda Civic for 9 days to a non- VIP. The rent for a hatchback over a week is 25\$ since we are renting it for 9 days,  $25 \times 9 = 225$ \$. So the math checks out. Now let's compare this with the same hatchbacks but under a week.

```
Enter your choice (1-5): 1
Enter your name: ben dover
Enter car type (Hatchbacks, Sedans, SUVs): Hatchbacks
Enter car model: Skoda Octavia
Enter rental duration (in days): 5
Are you a VIP member? (yes/no): no
```

```
Renting Skoda Octavia (Hatchbacks) for 5 days to ben dover.
Total cost: $150
```

Fig. 7.

In Fig. 7. The same car type (hatchbacks) is rented by a different customer for 5 days. The price of a hatchback under a week is 30\$ so the overall cost is  $30 \times 5 = 150$ \$, the math checks out here as well. For the next one, let's rent the hatchbacks again but this time we do it as a VIP.

```
Enter your choice (1-5): 1
Enter your name: Naveen
Enter car type (Hatchbacks, Sedans, SUVs): Hatchbacks
Enter car model: Peugeot208
Enter rental duration (in days): 11
Are you a VIP member? (yes/no): yes
```

```
Renting Peugeot208 (Hatchbacks) for 11 days to Naveen.
Total cost: $220
```

Fig. 8.

Now, since the VIP price for hatchbacks is 20\$, The overall cost would be  $20 \times 11 = 220$ . So the math checks out here as well. Now let's move to displaying the rented cars.

### 3. Display the rented cars :

```
4. Display rented cars
5. Exit
Enter your choice (1-5): 4
```

```
Rented Cars:
```

```
The Hatchbacks - Honda Civic
- is rented by jack ingof
```

```
The Hatchbacks - Peugeot208
- is rented by Naveen
```

```
The Hatchbacks - Skoda Octavia
- is rented by ben dover
```

Fig. 9.

From fig. 9. We can see all the cars we rented as of now.

#### 4. Return a rented car :

```
Enter your choice (1-5): 2
Enter your name: jack ingof
Enter car type (Hatchbacks, Sedans, SUVs): Hatchbacks
Enter car model: Honda Civic

Returning Honda Civic (Hatchbacks) by jack ingof.
```

Fig. 10.

From fig. 10. We can say that the car rented by jack ingof has been returned in order to confirm it we should display all the rented cars again.

```
Enter your choice (1-5): 4
Rented Cars:

The Hatchbacks - Peugeot208
- is rented by Naveen

The Hatchbacks - Skoda Octavia
- is rented by ben dover
```

Fig. 11.

From the figure above we can see that the car returned by jack ingof is gone from the rented cars list. With this we can say that the code is working fine.

#### 5. Inquire about car types and prices :

From the image to the right (fig.12) we can confirm that, the prices are displayed

Car Types and Prices:	
Hatchbacks:	
1-6 days:	\$30
7+ days:	\$25
VIP:	\$20
Sedans:	
1-6 days:	\$50
7+ days:	\$40
VIP:	\$35
SUVs:	
1-6 days:	\$100
7+ days:	\$90
VIP:	\$80

Fig. 12.

## 6. Exit :

The exit just exits the loop with a goodbye message, the screenshot for it will be inserted below.

```
----- Welcome To Premium Deluxe Motorsport!!!-----
1. Rent a car
2. Return a rented car
3. Display available cars
4. Display rented cars
5. Inquire about car types and prices
6. Exit
Enter your choice (1-6): 6
-----GOOD BYE!!! SAFE TRAVELS!!!-----
PS C:\Users\navee> |
```

## V Conclusion and Further work :

The code that I've developed is very basic and long, but there are few upgrades that can be made. One idea is, create a subclass called memberships inside the main class (Car rental system) and define two extra memberships like Regular, VIP and Premium where VIP is automatically assigned to regular customers who rent multiple cars and have the ability to buy premium membership for like 400\$, and also introduce new vehicle class called luxury and exotic cars, rental yachts, Planes, and so it can be scaled to a very good business product. I could've also made multiple classes and import them all together, the code will get smaller but it didn't seem necessary. Also for the main program, the user inputs has to be how it's instructed (for example, when you're selecting the type of the car you want, if you type hatchbacks, it wont recognise the car type, it has to be Hatchbacks. We can use the lower() to rectify this but then we will have to change them in their dictionary itself and if we change that they look weird when you are displaying the list of cars available so it better stays as it is.