# Problem Statement

**Objective:**

Build an online car rental platform using Object-Oriented Programming in Python.

**Problem Statement:**

A car rental company has requested you to build an online car rental platform where customers should be able to view the available cars that can be rented on an hourly, daily, or weekly basis. The company can display the available inventory and confirm requests by checking the available stock. Customers will receive an auto-generated bill when they return the car.

For simplicity, let’s assume that:

1. Customers can rent cars from any one of the following options—hourly, daily, or weekly rental.
2. Customers are free to choose any number of cars they want, provided the number of available cars is more than the number of requested cars.

# Task

1. Create a module (.py file) for car rental and import the built-in module DateTime to handle the rental time and bill.
2. Create a class for renting the cars and define a constructor in it.
3. Define a method for displaying the available cars. Also, define methods for renting cars on an hourly, daily and weekly basis, respectively.
4. Inside these methods, make sure that the number of requested cars is positive and lesser than the total available cars.
5. Store the time of renting a car in a variable, which can later be used in the bill while returning the car.
6. Define a method to return the cars using rental time, rental mode (hourly, daily, or weekly), and the number of cars rented.
7. Inside the return method; update the inventory stock, calculate the rental period, and generate the final bill.
8. Create a class for customers and define a constructor in it.
9. Define methods for requesting the cars and returning them.
10. Next, create the main project (.ipynb) file and import the car rental module.
11. Define the main method and create objects for both car rental and customer classes.
12. Inside the main method, take the customer’s input as a choice for displaying car availability, rental modes, or returning the cars.
13. Use the relevant method for the customer’s input and print relevant messages.
14. Run the main method to start your project.

# Code

***car\_rental.py***

from datetime import datetime, timedelta

class CarRental:

def \_\_init\_\_(self, inventory):

self.inventory = inventory

def display\_available\_cars(self):

print(f"Available cars: {self.inventory}")

def rent\_hourly(self, num\_cars):

return self.rent\_car(num\_cars, "hourly")

def rent\_daily(self, num\_cars):

return self.rent\_car(num\_cars, "daily")

def rent\_weekly(self, num\_cars):

return self.rent\_car(num\_cars, "weekly")

def rent\_car(self, num\_cars, rental\_type):

if num\_cars > self.inventory:

print("Not enough cars available.")

return None

current\_time = datetime.now()

rental\_details = {

'rental\_type': rental\_type,

'rental\_period': num\_cars,

'rental\_start\_time': current\_time

}

self.inventory -= num\_cars

return rental\_details

def return\_car(self, rental\_details):

rental\_end\_time = datetime.now()

rental\_period = rental\_details['rental\_period']

rental\_type = rental\_details['rental\_type']

rental\_start\_time = rental\_details['rental\_start\_time']

# Calculate elapsed time

elapsed\_time = rental\_end\_time - rental\_start\_time

if rental\_type == "hourly":

bill = rental\_period \* 10 # Example hourly rate

elif rental\_type == "daily":

bill = rental\_period \* 50 # Example daily rate

elif rental\_type == "weekly":

bill = rental\_period \* 200 # Example weekly rate

else:

bill = 0

# Generate detailed bill

print(f"Number of cars returned: {rental\_period}")

print(f"Elapsed time: {elapsed\_time}")

print(f"Total bill: ${bill}")

# Update inventory

self.inventory += rental\_period

return bill

class Customer:

def request\_car(self, car\_rental, num\_cars, rental\_type):

if rental\_type == "hourly":

return car\_rental.rent\_hourly(num\_cars)

elif rental\_type == "daily":

return car\_rental.rent\_daily(num\_cars)

elif rental\_type == "weekly":

return car\_rental.rent\_weekly(num\_cars)

else:

print("Invalid rental type.")

return None

def return\_car(self, car\_rental, rental\_details):

bill = car\_rental.return\_car(rental\_details)

print(f"Total bill: ${bill}")

***car\_rental\_platform.ipynb***

from datetime import datetime, timedelta

from car\_rental import CarRental, Customer

def main():

inventory = 10 # Initial inventory of cars

car\_rental = CarRental(inventory)

customer = Customer()

while True:

print("\nOptions:")

print("1. Display available cars")

print("2. Rent a car")

print("3. Return a car")

print("4. Exit")

choice = input("Enter choice (1/2/3/4): ")

if choice == "1":

car\_rental.display\_available\_cars()

elif choice == "2":

num\_cars = int(input("Enter number of cars to rent: "))

rental\_type = input("Enter rental type (hourly/daily/weekly): ")

rental\_details = customer.request\_car(car\_rental, num\_cars, rental\_type)

if rental\_details:

print(f"Successfully rented {num\_cars} cars.")

elif choice == "3":

if car\_rental.inventory == 0:

print("No cars to return.")

continue

num\_cars\_returned = int(input("Enter number of cars to return: "))

rental\_details = {

'rental\_type': 'hourly', # Example previous rental type

'rental\_period': num\_cars\_returned, # Example previous rental period

'rental\_start\_time': datetime.now() - timedelta(hours=3) # Example start time

}

# Return the cars and get the bill

bill = customer.return\_car(car\_rental, rental\_details)

if bill is not None:

print(f"Total bill: ${bill}")

elif choice == "4":

print("Exiting program.")

break

else:

print("Invalid input. Please enter 1, 2, 3, or 4.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

# Output

A white background with black text

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A white background with black text

Description automatically generated