A pool Car rental Management system

A person who has a plan to start a car rental business system in Goa, since Goa is tourist place, where most of peoples are comes here to enjoy their holiday from various regions of India and across the globe. For that the businessperson is approaching you to develop a Car Rental Management System in Python using Object Oriented Programming (OOP). This project should be seen easy design for the customer to access the features as hassle free. Also, administrator access is to be simple and dynamic in nature to update the key attributes of the functionality in this project.

```
import datetime

class Car:
    def __init__(self, car_id, manufacturer, model, year, mileage, last_service_date, tari
        self.car_id = car_id
        self.manufacturer = manufacturer
        self.model = model
        self.year = year
        self.mileage = mileage
        self.last_service_date = last_service_date
        self.tariff = tariff
        self.segment = segment
        self.available = True

    def is_service_due(self):
        today = datetime.date.today()
        last_service_age = today.year - self.last_service_date.year
        return last_service_age >= 1

class CarRentalSystem:
    def __init__(self):
        self.cars = []

    def add_car(self, car):
        self.cars.append(car)

    def rent_car(self, car_id, duration):
        for car in self.cars:
            if car.car_id == car_id and car.available:
                car.available = False
                rental_cost = car.tariff * duration
                return rental_cost
        return None

    def display_available_cars(self, segment=None):
        available_cars = [car for car in self.cars if car.available]
        if segment:
            available_cars = [car for car in available_cars if car.segment == segment]
        return available_cars

if __name__ == "__main__":
    rental_system = CarRentalSystem()

    car1 = Car(1, "Toyota", "Corolla", 2020, 10000, datetime.date(2023, 1, 1), 100, "Basic'
    car2 = Car(2, "BMW", "X5", 2021, 5000, datetime.date(2023, 2, 1), 200, "Luxury")

    rental_system.add_car(car1)
    rental_system.add_car(car2)

    car_id_to_rent = 1
    rental_duration = 5
    rental_cost = rental_system.rent_car(car_id_to_rent, rental_duration)

    if rental_cost is not None:
        print(f"Rental cost: {rental_cost} INR")
    else:
        print("Car not available or not found.")

    available_cars = rental_system.display_available_cars("Basic")
    print("Available Basic Cars:")
    for car in available_cars:
        print(f"Car ID: {car.car_id}, Manufacturer: {car.manufacturer}, Model: {car.model}'
```

```
Rental cost: 500 INR
Available Basic Cars:
```

Railway Management System

• Problem Statement:

The problem statement is to create the Railway Management System that develops a user-friendly software application that facilitates various functionalities related to railway ticketing and management. The system should allow users to book tickets, cancel booking, check fares, view their bookings, and display available trains.

```
class Train:
    def __init__(self, train_id, name, source, destination, departure_time, arrival_time, s
        self.train_id = train_id
        self.name = name
        self.source = source
        self.destination = destination
        self.departure_time = departure_time
        self.arrival_time = arrival_time
        self.seats_available = seats_available
        self.fare = fare

class Booking:
    def __init__(self, booking_id, train, class_type, travel_date):
        self.booking_id = booking_id
        self.train = train
        self.class_type = class_type
        self.travel_date = travel_date

class RailwaySystem:
    def __init__(self):
        self.trains = []
        self.bookings = []
        self.booking_counter = 1

    def add_train(self, train):
        self.trains.append(train)

    def book_ticket(self, train_id, class_type, travel_date):
        for train in self.trains:
            if train.train_id == train_id:
                if train.seats_available > 0:
                    booking = Booking(self.booking_counter, train, class_type, travel_date
                    self.bookings.append(booking)
                    train.seats_available -= 1
                    self.booking_counter += 1
                    return f"Booking successful. Booking ID: {booking.booking_id}"
                else:
                    return "Sorry, no seats available for this train."
        return "Train not found."

    def cancel_booking(self, booking_id):
        for booking in self.bookings:
            if booking.booking_id == booking_id:
                booking.train.seats_available += 1
                self.bookings.remove(booking)
                return "Booking canceled successfully."
        return "Booking not found."

    def check_fare(self, train_id, class_type):
        for train in self.trains:
            if train.train_id == train_id:
                if class_type.lower() == "first class":
                    return f"Fare for {train.name} ({class_type}): {train.fare * 2}"
                elif class_type.lower() == "second class":
                    return f"Fare for {train.name} ({class_type}): {train.fare}"
        return "Train not found."
    def view_bookings(self):
        if not self.bookings:
            return "No bookings found."
        else:
            booking_details = []
            for booking in self.bookings:
                booking_details.append(f"Booking ID: {booking.booking_id}, Train: {booking.
                                       f"Class: {booking.class_type}, Travel Date: {booking
            return "\n".join(booking_details)

    def check_train_availability(self, source, destination):
        available_trains = []
        for train in self.trains:
            if train.source.lower() == source.lower() and train.destination.lower() == dest
                available_trains.append(train)
        if not available_trains:
            return "No trains available for this route."
        else:
            train_details = []
            for train in available_trains:
                train_details.append(f"Train: {train.name}, Departure Time: {train.departur
                                     f"Arrival Time: {train.arrival_time}, Seats Available
            return "\n".join(train_details)

if __name__ == "__main__":
    railway_system = RailwaySystem()

    train1 = Train(1, "Express", "A", "B", "09:00 AM", "12:00 PM", 50, 500)
    train2 = Train(2, "Local", "B", "C", "02:00 PM", "05:00 PM", 100, 200)
    railway_system.add_train(train1)
    railway_system.add_train(train2)

    while True:
        print("\nRailway Management System Menu:")
        print("1. Book a Ticket")
        print("2. Cancel Booking")
        print("3. Check Fare")
        print("4. View Bookings")
        print("5. Check Train Availability")
        print("6. Exit")

        choice = input("Enter your choice: ")

        if choice == "1":
            train_id = int(input("Enter Train ID: "))
            class_type = input("Enter Class (First Class/Second Class): ")
            travel_date = input("Enter Travel Date (YYYY-MM-DD): ")
            result = railway_system.book_ticket(train_id, class_type, travel_date)
            print(result)
        elif choice == "2":
            booking_id = int(input("Enter Booking ID: "))
            result = railway_system.cancel_booking(booking_id)
            print(result)

        elif choice == "3":
            train_id = int(input("Enter Train ID: "))
            class_type = input("Enter Class (First Class/Second Class): ")
            result = railway_system.check_fare(train_id, class_type)
            print(result)

        elif choice == "4":
            bookings = railway_system.view_bookings()
            print(bookings)

        elif choice == "5":
            source = input("Enter Source Station: ")
            destination = input("Enter Destination Station: ")
            trains = railway_system.check_train_availability(source, destination)
            print(trains)

        elif choice == "6":
            print("Thank you for using the Railway Management System!")
            break

        else:
            print("Invalid choice. Please try again.")
```

```
Railway Management System Menu:
1. Book a Ticket
2. Cancel Booking
3. Check Fare
4. View Bookings
5. Check Train Availability
6. Exit
Enter your choice: 4
No bookings found.

Railway Management System Menu:
1. Book a Ticket
2. Cancel Booking
3. Check Fare
4. View Bookings
5. Check Train Availability
6. Exit
Enter your choice: 6
Thank you for using the Railway Management System!
```

YouTube Video Statistics

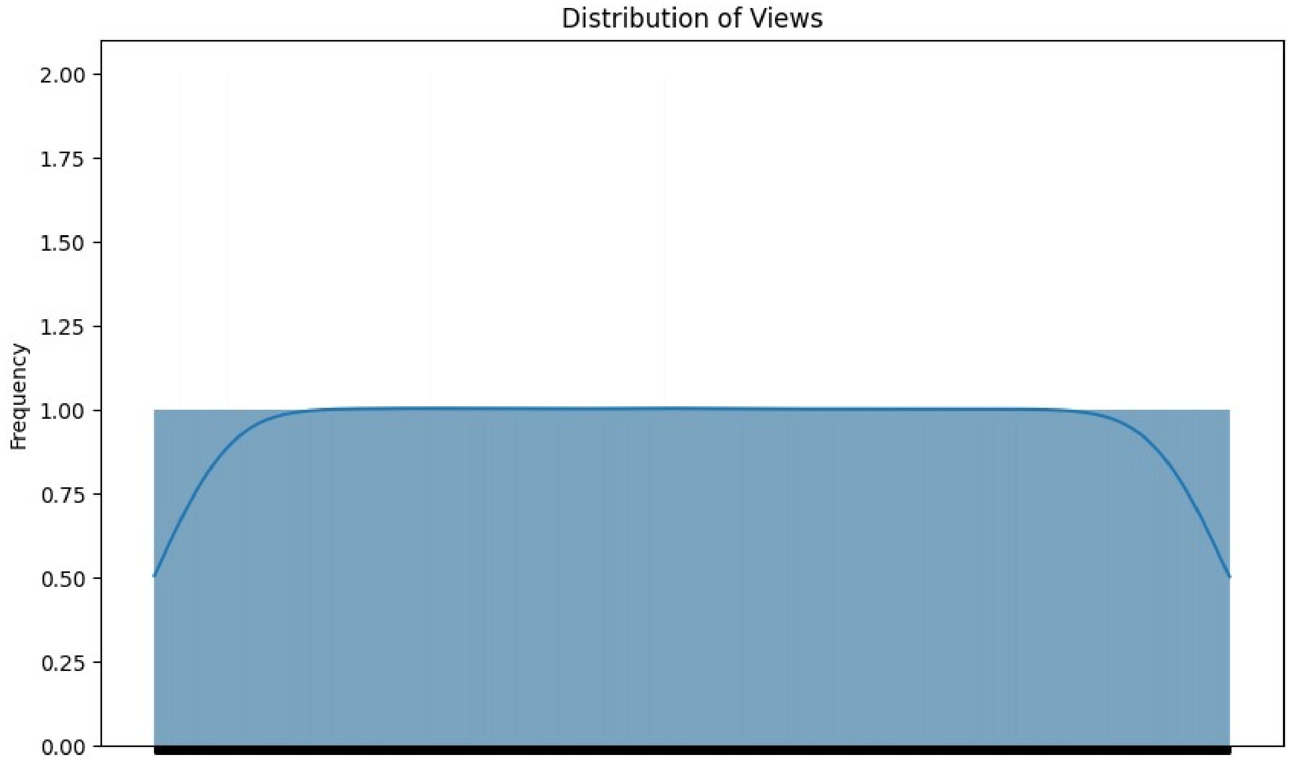
Abstract:

YouTube (the world-famous video sharing website) maintains a list of the top trending videos on the platform. According to Variety magazine, to determine the year's top- trending videos, YouTube uses a combination of factors including measuring user's interactions (number of views, shares, comments, and likes). Note: that they're not the most-viewed videos overall for the calendar year.

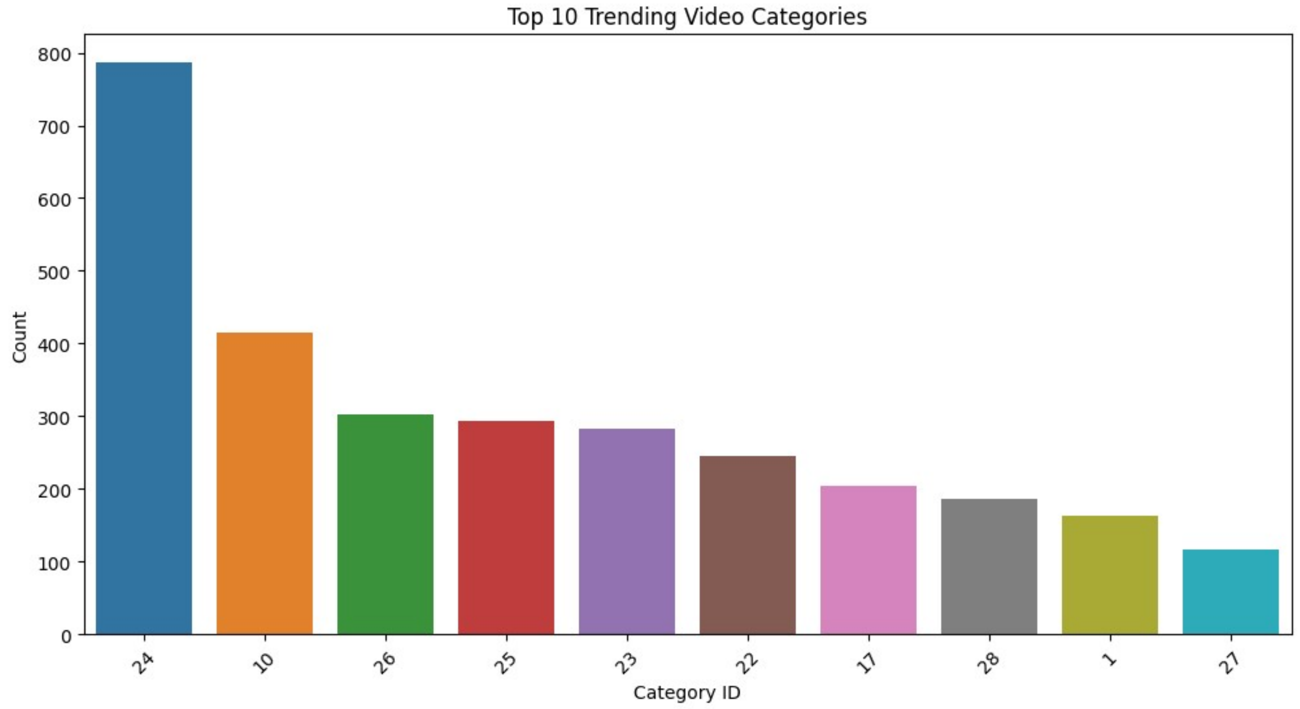
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

youtube_data = pd.read_csv('/content/youtube.csv')

print(youtube_data.head())

print(youtube_data.describe())

print(youtube_data.isnull().sum())

plt.figure(figsize=(10, 6))
sns.histplot(youtube_data['views'], bins=50, kde=True)
plt.title('Distribution of Views')
plt.xlabel('Views')
plt.ylabel('Frequency')
plt.show()

top_categories = youtube_data['category_id'].value_counts().head(10)
plt.figure(figsize=(12, 6))
sns.barplot(x=top_categories.index, y=top_categories.values)
plt.title('Top 10 Trending Video Categories')
plt.xlabel('Category ID')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()

plt.figure(figsize=(10, 6))
sns.scatterplot(data=youtube_data, x='likes', y='dislike', hue='comment_disabled')
plt.title('Likes vs. Dislikes (Color-coded by Comment_disabled)')
plt.xlabel('Likes')
plt.ylabel('Dislikes')
plt.show()

sns.pairplot(data=youtube_data[['views', 'likes', 'dislike', 'comment_count']])
plt.show()
```

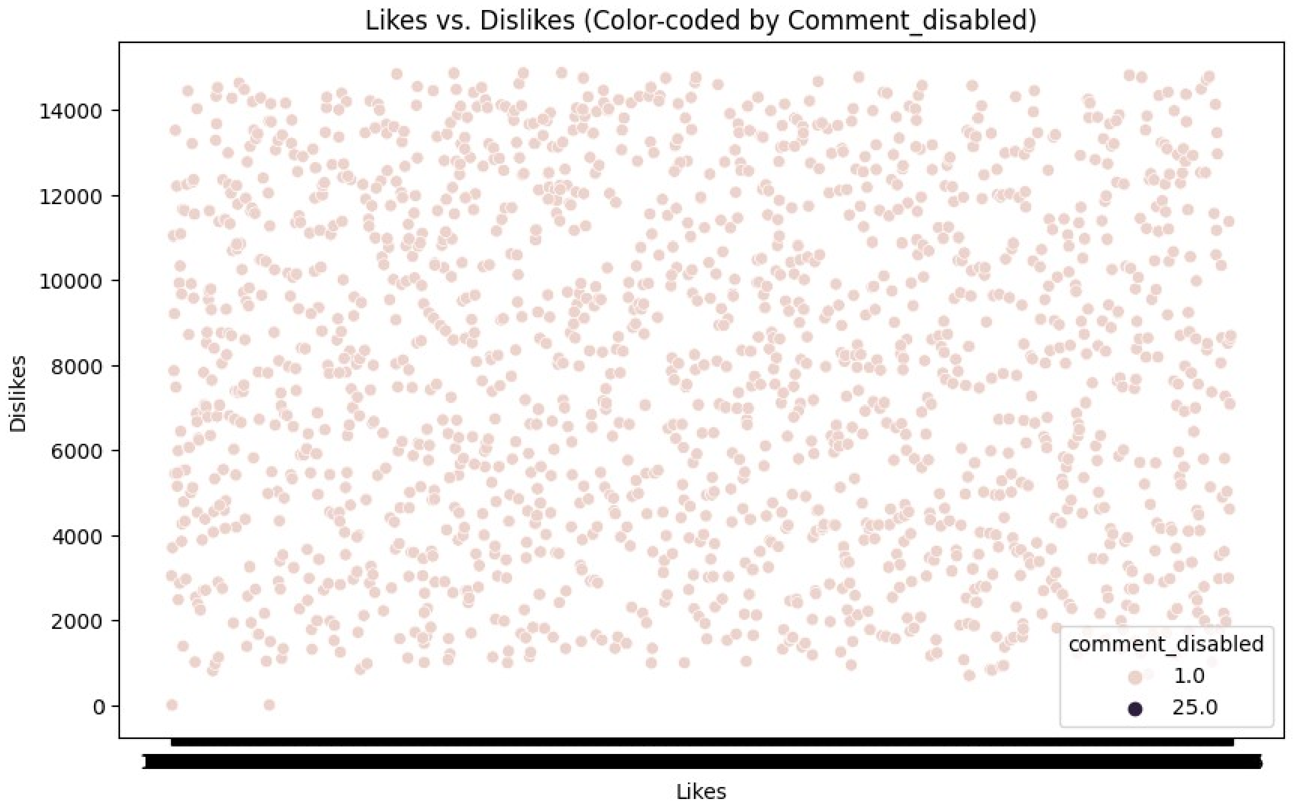
```
    Video_id category_id         channel_title  subscriber  \
0   HDR9SQc79          22          CaseyNeistat   9086142.0
1  KNH52UF?48          24       LastWeekTonight   5937292.0
2  QTW28IRG36          23          Rudy Mancuso   4191209.0
3  MGL76WI]26          24  Good Mythical Morning  13186408.0
4  TWP93KXT70          24              nigahiga  20563106.0

                                               title  \
0                  WE WANT TO TALK ABOUT OUR MARRIAGE
1  The Trump Presidency: Last Week Tonight with J...
2  Racist Superman | Rudy Mancuso, King Bach & Le...
3                    Nickelback Lyrics: Real or Fake?
4                            I Dare You: GOING BALD!?

                                                tags  \
0                                     SHANtell martin
1  last week tonight trump presidency|last week t...
2  racist superman|rudy|mancuso|king|bach|racist|...
3  rhett and link|gmm|good mythical morning|rhett...
4  ryan|higa|higatv|nigahiga|i dare you|idy|rhpc|...

                                         description  Trend_day_count  \
0  SHANTELL'S CHANNEL - https://www.youtube.com/s...              6.0
1  One year after the presidential election, John...              1.0
2  WATCH MY PREVIOUS VIDEO â–¶ \n\nSUBSCRIBE â–º ...             10.0
3  Today we find out if Link is a Nickelback amat...             12.0
4  I know it's been a while since we did this sho...             11.0

  Tag_count Trend_tag_count comment_count  comment_disabled  \
0        21               6                            NaN
1        23               1        116266              1.0
2        22               3        257850              1.0
3        17               5        263939              1.0
4        15               7        268085              1.0

   like dislike disabled  likes  dislike  tag appered in title    views  \
0                    NaN  13342   6089.0                   NaN  1978978
1                    NaN   5761   3044.0                   NaN  1487870
2                    1.0      0      0.0                   1.0  1502102
3                    1.0      0      0.0                   1.0  3519302
4                    1.0      0      0.0                   1.0  4835374

   Unnamed: 17  Unnamed: 18
0          NaN          NaN
1          NaN          NaN
2          NaN          NaN
3          NaN          NaN
4          NaN          NaN
         subscriber  Trend_day_count  comment_disabled  like dislike disabled  \
count  3.175000e+03      3197.000000       2183.000000            844.000000
mean   3.823981e+06         7.964342          1.010994              1.003555
std    2.865771e+07        78.556055          0.513670              0.103264
min    0.000000e+00         0.000000          1.000000              1.000000
25%    2.428800e+05         4.000000          1.000000              1.000000
```

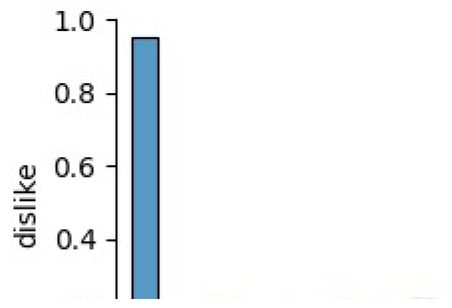
```
50%    1.241220e+06       7.000000        1.000000               1.000000
75%    3.812622e+06      10.000000        1.000000               1.000000
max    1.576229e+09    4444.000000       25.000000               4.000000

            dislike  tag appered in title  Unnamed: 17  Unnamed: 18
count   3197.000000                2108.0          1.0          0.0
mean    5784.686268                   1.0       2544.0          NaN
std     4860.754493                   0.0          NaN          NaN
min        0.000000                   1.0       2544.0          NaN
25%        0.000000                   1.0       2544.0          NaN
50%     5354.000000                   1.0       2544.0          NaN
75%    10042.000000                   1.0       2544.0          NaN
max    14858.000000                   1.0       2544.0          NaN
Video_id                 665
category_id              671
channel_title            668
subscriber               688
title                    668
tags                     817
description               65
Trend_day_count          666
Tag_count                666
Trend_tag_count          666
comment_count            665
comment_disabled        1680
like dislike disabled   3019
likes                    665
dislike                  666
tag appered in title    1755
views                    665
Unnamed: 17             3862
Unnamed: 18             3863
dtype: int64
```

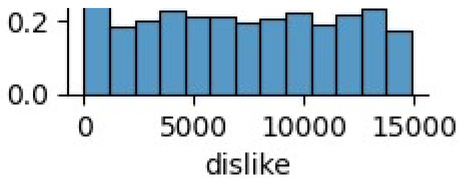


Views


Top 10 Trending Video Categories
Count
800
700
600
500
400
300
200
100
0
24
10
26
25
23
22
17
28
1
27
Category ID


Likes vs. Dislikes (Color-coded by Comment_disabled)
Dislikes
14000
12000
10000
8000
6000
4000
2000
0
comment_disabled
1.0
25.0
Likes


1.0
0.8
0.6
0.4
dislike


0.2
0.0
0
5000
10000
15000
dislike