**RECOMMENDATION SYSTEM FOR YOUTUBE CONTENT CREATORS**

**MINOR PROJECT REPORT**

**December 2022**

**SYNOPSIS**

A Recommendation System for YouTube Content Creator is used to give YouTube content creators (YouTubers) suggestions for their content i.e., videos in YouTube according to the current trends going on which are popular in YouTube.

This system works by getting real life data from YouTube which is then preprocessed to make it fit for analysis. Then, many factors like views, post date, etc. are analyzed to get a result for the recommendation on when and what to post a video for the YouTubers. These are visualized in an attractive manner to make the analysis part easy.

This system can be implemented for YouTubers make it easy for them to decide on what type of videos to make depending on the current trends in the platform.

The interface of the project work entitled **"Recommendation System for YouTube Content Creators"** is developed using **STREAMLIT** as Front end, **PANDAS** as data processing and manipulation tool, **NATURAL LANGUAGE TOOLKIT(NLTK)** as language processor and **PYTHON** as Back end.

**TABLE OF CONTENTS**

**CONTENTS**  **PAGE NO.**

1. **INTRODUCTION**
   1. Introduction to the project 1
   2. Overview of the system 2
2. **SYSTEM CONFIGURATION**
   1. Software Specification 3
   2. Hardware Configuration 3
   3. Software Description 4
3. **SYSTEM STUDY**
   1. Existing System 8
   2. Proposed System 9
4. **SYSTEM ANALYSIS AND DESIGN** 
   1. Modular Description 10
   2. Database Design 11
   3. Input Design 13
   4. Output Design 14
5. **SCOPE FOR FUTURE ENHANCEMENTS** 15
6. **CONCLUSION** 16
7. **BIBLIOGRAPHY**  17
8. **APPENDICES**
9. Screen Shots 18
10. Sample Coding 25

**1.INTRODUCTION**

* 1. **INTRODUCTION TO THE PROJECT:**

YouTube is a video sharing service where users can watch, like, share, comment and upload their own videos. The video service can be accessed on PCs, laptops, tablets and via mobile phones. Users can search for and watch videos, create a personal YouTube channel, upload videos to your channel, like/comment/share other YouTube videos, users can subscribe/follow other YouTube channels and users, create playlists to organize videos and group videos together.

Recommendations are a key method for information retrieval and content discovery in today’s information-rich environment. They allow users facing a huge amount of information to navigate that information in an eﬃcient and satisfying way. As the largest and most-popular online video community with vast amounts of user-generated content, YouTube presents some unique opportunities and challenges for content discovery and recommendations.

This recommendation system for YouTube Content Creators can be used to give or get guidance for the content that these content creators (YouTubers) make or release to get maximum coverage and also positive reach based on the data each country’s YouTube data.

YouTube maintains a list of the top trending videos on the platform. To determine the year’s top-trending videos, YouTube uses a combination of factors including measuring user interactions (number of views, shares, comments and likes). This data is used here for each countries where they are analyzed to get proper recommendations

**1.2 OVERVIEW OF THE SYSTEM:**

The project entitled “**Recommendation System for YouTube Content Creators**” is used to get good suggestions for the creators without having to spend go through a lot of other videos or spend a lot of time in deciding on the content to make. Since they are not cost effective and time consuming, this project making use of YouTube’s trending video statistics is proposed.

**“Recommendation System for YouTube Content Creators”** project is a web application is developed using Streamlit along with NLTK for processing of the human languages used in many of the webpages and video titles. As the project title implies, the main aim of this project is to create a platform for the Content Creators to get Recommendations for their videos.

The project helps to automate data preprocessing to clean the datasets, then data analysis to get needed statistics and information. Then visualize these data to make decision making easier and avoid longer time taken in such processes.

**2. SYSTEM CONFIGURATION**

The software is designed to be light-weighted so that it doesn’t be a burden on the machine running it. This system is being build keeping in mind the generally available hardware and software compatibility. Here are the minimum hardware and software requirement:

**2.1 HARDWARE CONFIGURATION:**

**MICROPROCESSOR :** Intel i3 or above

**RAM :** 8 GB

**HARD DISK :** 256 GB SSD

**2.2 SOFTWARE CONFIGURATION:**

**OPERATING SYSTEM :** Windows 7(32 bit) or above

**FRONT END :** Streamlit

**LANGUAGE USED :** Python

**APPLICATION :** Any web browser

**2.3 SOFTWARE DESCRIPTION:**

**PYTHON:**

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

**Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

**Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

**Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

**Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features, such as list comprehensions and a garbage collection system using reference counting. Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible. Python 2 was discontinued with version 2.7.18 in 2020. Python consistently ranks as one of the most popular programming languages

**Python Features:**

* **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read −** Python code is more clearly defined and visible to the eyes.
* **A broad standard library −** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode −** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable −** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable −** You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **GUI Programming −** Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

**STREAMLIT:**

Streamlit is an open-source python based framework for developing and deploying interactive data science dashboards and machine learning models. This means that you do not have to rely on a team of front end developers or spend large amounts of time learning web design languages such as HTML, CSS or JavaScript in order to deploy your dashboard or model.

With Streamlit, no callbacks are needed since widgets are treated as variables. Data caching simplifies and speeds up computation pipelines. Streamlit watches for changes on updates of the linked Git repository and the application will be deployed automatically in the shared link.

It is built on top of Python and supports many of the mainstream Python libraries such as scikit-learn, Keras, PyTorch, SymPy(latex), NumPy, pandas, Matplotlib etc.

**JUPYTER NOTEBOOK:**

Jupyter notebook is an interactive notebook allowing you to write documents with embedded code, and execute this code on the fly. It was originally developed as a part of the Ipython project, and could only be used for Python code at that time. Nowadays, the Jupyter notebook integrates multiple languages, such as R, Julia, Haskell and much more – the notebook supports about 50 languages.

These are the following advantages of Jupyter Notebook –

* **All in one place:** As you know, Jupyter Notebook is an open-source web-based interactive environment that combines code, text, images, videos, mathematical equations, plots, maps, graphical user interface and widgets to a single document.
* **Easy to convert:** Jupyter Notebook allows users to convert the notebooks into other formats such as HTML and PDF. It also uses online tools and nbviewer which allows you to render a publicly available notebook in the browser directly.
* **Easy to share:** Jupyter Notebooks are saved in the structured text files (JSON format), which makes them easily shareable.
* **Language independent:** Jupyter Notebook is platform-independent because it is represented as JSON (JavaScript Object Notation) format, which is a language-independent, text-based file format. Another reason is that the notebook can be processed by any programing language, and can be converted to any file formats such as Markdown, HTML, PDF, and others.
* **Interactive code:** Jupyter notebook uses ipywidgets packages, which provide many common user interfaces for exploring code and data interactivity.

**NATURAL LANGUAGE TOOLKIT(NLTK):**

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Thanks to a hands-on guide introducing programming fundamentals alongside topics in computational linguistics, plus comprehensive API documentation, NLTK is suitable for linguists, engineers, students, educators, researchers, and industry users alike. NLTK is available for Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open source, community-driven project.

NLTK has been called “a wonderful tool for teaching, and working in, computational linguistics using Python,” and “an amazing library to play with natural language.”

Natural Language Processing with Python provides a practical introduction to programming for language processing. Written by the creators of NLTK, it guides the reader through the fundamentals of writing Python programs, working with corpora, categorizing text, analyzing linguistic structure, and more.

**WORDCLOUD:**

A word cloud (also known as a tag cloud or text cloud) is a visual representation of a text, in which the words appear bigger the more often they are mentioned.

Word clouds are great for visualizing unstructured text data and getting insights on trends and patterns. For example, take a look at this word cloud we created using a free online word cloud generator to analyze hotel reviews.

**3. SYSTEM STUDY**

Systems analysis the process of observing systems for troubleshooting or development purposes. It is applied to information technology, where computer-based systems require defined analysis according to their makeup and design. Systems analysis can include looking at end-user implementation of a software package or product; looking in-depth at source code to define the methodologies used in building software; or taking feasibility studies and other types of research to support the use and production of a software product, among other things. Systems analysis professionals are often called upon to look critically at systems, and redesign or recommend changes as necessary.

Inside and outside of the business world, systems analysts help to evaluate whether a system is viable or efficient within the context of its overall architecture, and help to uncover the options available to the employing business or other party. Systems analysts are different than systems administrators, who maintain systems day to day, and their roles generally involve a top-level view of a system to determine its overall effectiveness according to its design**.**

**3.1 EXISTING SYSTEM:**

The process of analyzing the existing system is used to find the drawbacks of the existing system. Presently, there are no existing systems for such recommendations as it is mainly done by the users manually or by delegating the process to the concerned people.

**Disadvantages:**

* Requires a lot of time
* Complicated
* Manual labor needed
* Could cause losses

**3.2 PROPOSED SYSTEM:**

The proposed system is used to improve the application as a good and smart product. The following details explains the proposed system:

* This system saves a lot of time and effort of users.
* More data is provided in order to get more accurate results
* Simple User Interface for better understanding
* Profitable

**4. SYSTEM ANALYSIS AND DESIGN**

**4.1 MODULAR DESCRIPTION:**

**Data pre-processing:**

The dataset to be worked on is given as an input and converted to a pandas dataframe to work on. The dataset is cleaned to make it fit for any analysis.

**Approach:**

The data pre-processing is done by doing the following processes,

* Removing null values
* Dropping unwanted columns
* Updating to the latest data
* Getting rid of repetitive values
* Convert to proper datatypes

**Data Visualization:**

After being cleaned and preprocessed, the dataset is fit for analysis. Then, the data is analyzed and presented in visual forms like bar chart and scatter plot for making it easier to understand the statistics of the data.

**Language Processing:**

NLTK is used to work with the language data where it checks for all the common words used in a certain language that are repetitive in a lot of cases, and they are removed. Such words are called stopwords. In the English language, some of the common stopwords are [“a”, “the”, “is”, “are” ]. The stopword list of English actually consists of 733 elements.

**WordCloud Generation:**

We can create a wordcloud for each column of the dataset by creating a function for it. It will do the following process- Create an object of class WordCloud and call the generate() method. WordCloud() takes several arguments as per the need. The .generate() method takes one argument of the text we create. Use the .imshow() method of matplotlib.pyplot to display the Word Cloud as an image.

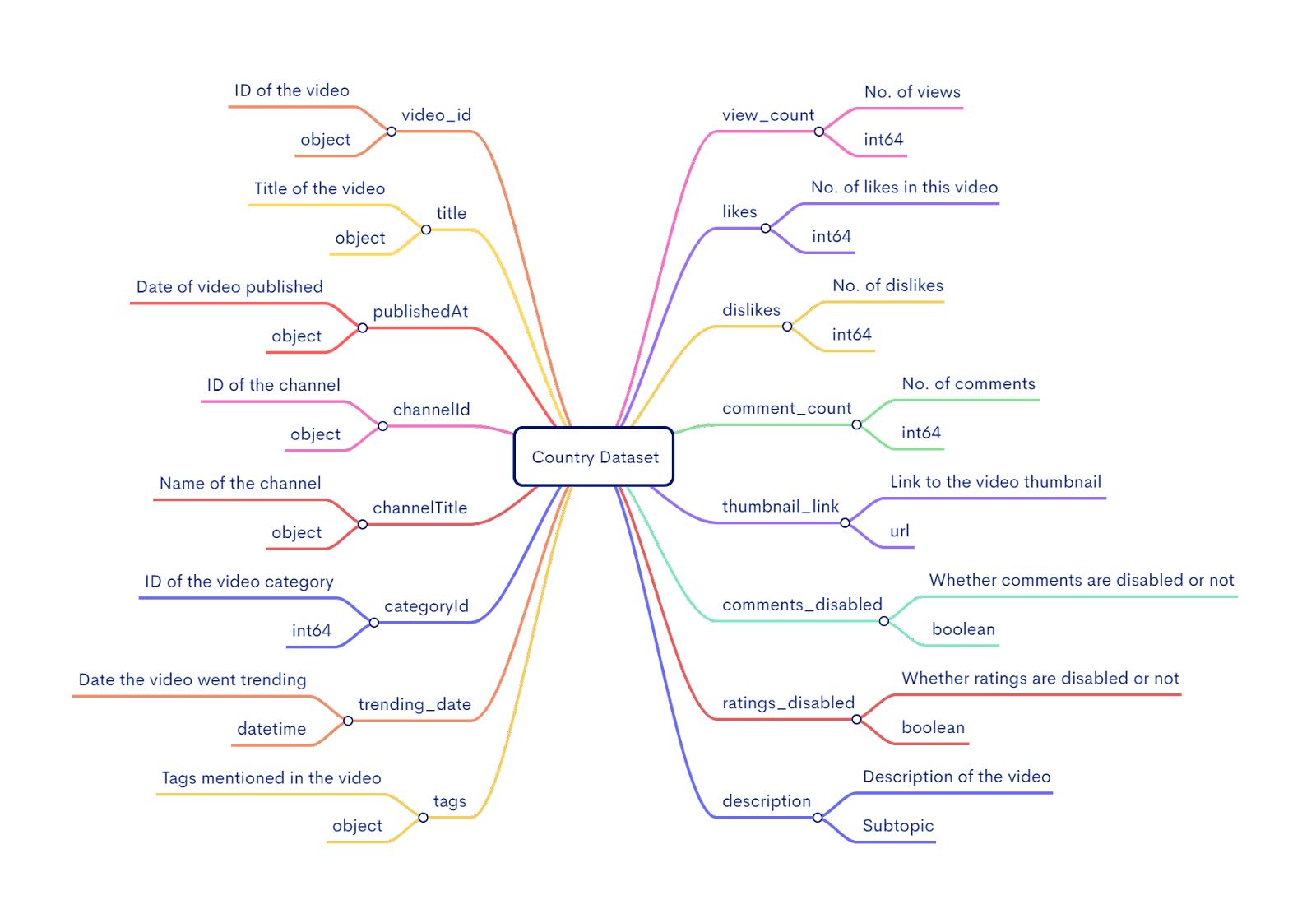
We can pass the generator for each column by calling them with their respective variables and display them. It will show us the most used words from each of the column in an visualized manner that is easily understood by everyone.

**4.2 DATABASE DESIGN:**

Database design can be generally defined as a collection of tasks or processes that enhance the designing, development, implementation, and maintenance of enterprise data management system.

The important consideration that can be taken into account while emphasizing the importance of database design can be explained in terms of the following points given below.

* Database designs provide the blueprints of how the data is going to be stored in a system. A proper design of a database highly affects the overall performance of any application.
* The designing principles defined for a database give a clear idea of the behavior of any application and how the requests are processed.
* Another instance to emphasize the database design is that a proper database design meets all the requirements of users.
* Lastly, the processing time of an application is greatly reduced if the constraints of designing a highly efficient database are properly implemented.

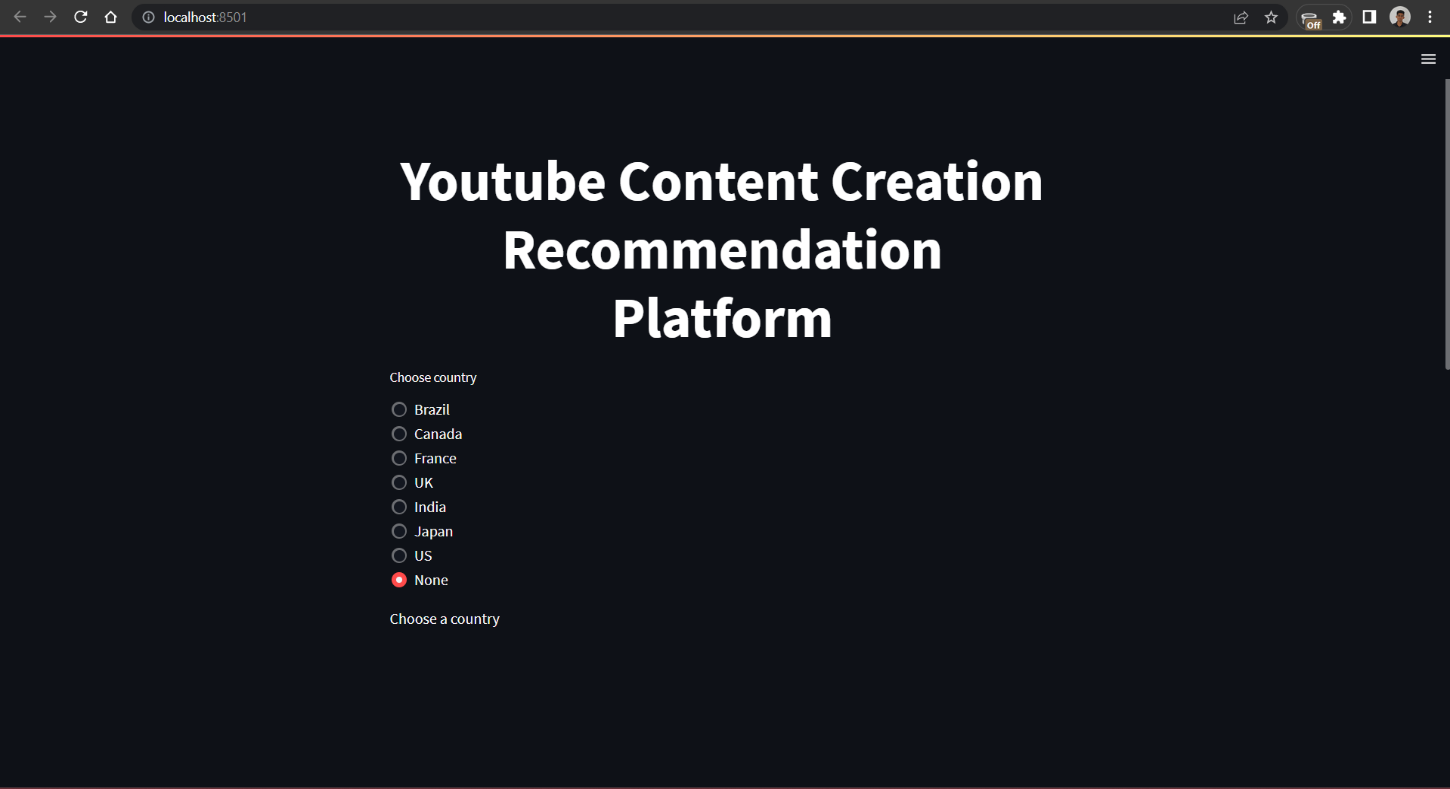


**4.3 INPUT DESIGN:**

Input design is one of the most expensive phases of the operation of computerized system and is often the major problem of a system. The decisions made during the input design are:

* To provide cost effective method of input.
* To achieve the highest possible level of accuracy.
* To ensure that input is understood by the user.

System analysts decide the following input design details like, what data item to input, what medium to use, how the data should be arranged or coded data items and transaction needing validations to detect errors and at last the dialogue to guide users in providing input. Input data of a system may not be necessarily a raw data captured in the system from scratch. These can also be the output of another system or sub-system. The design of input covers all phases of input from the certain of initial data to actual entering the data to the system for processing.

****

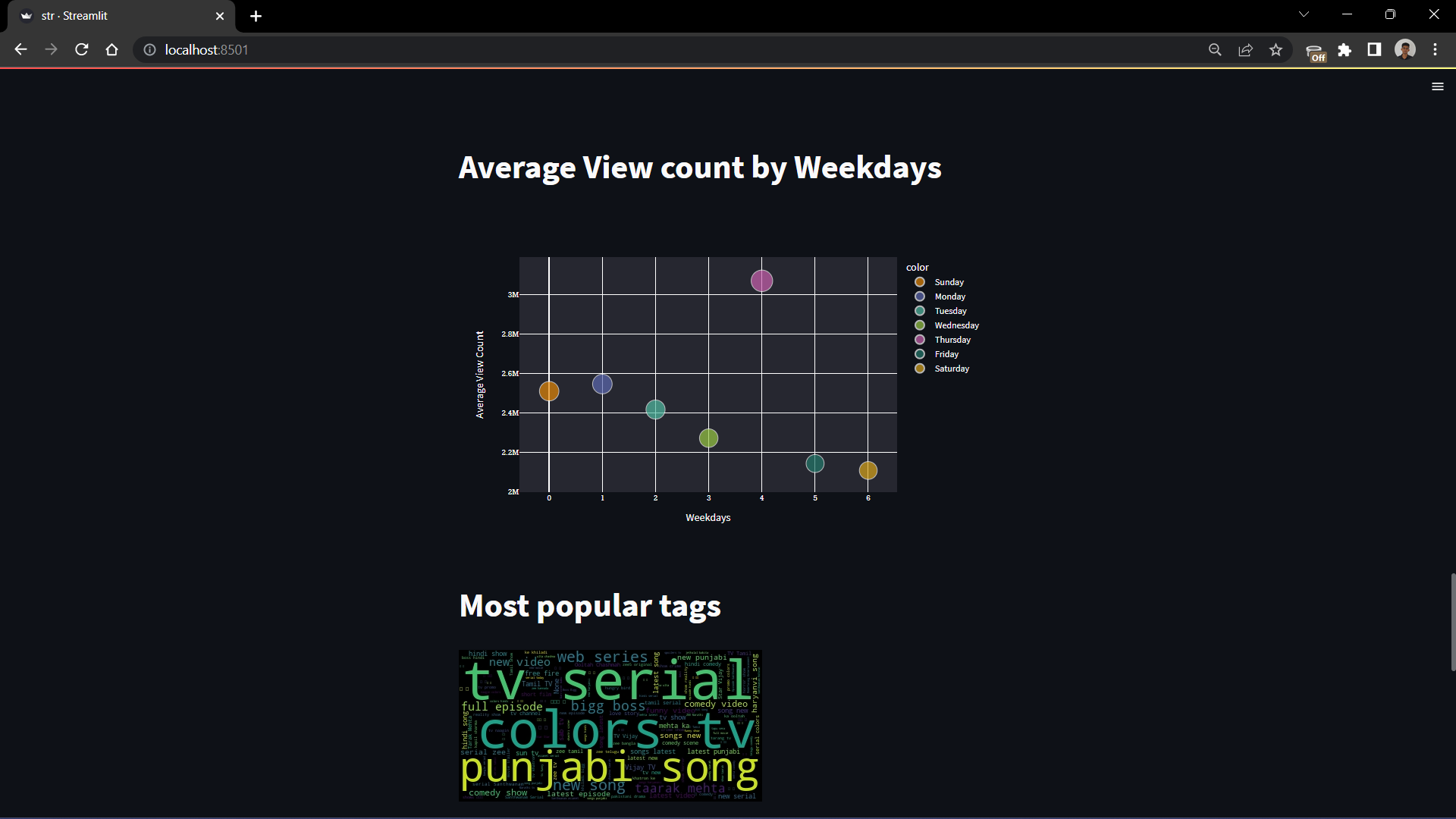
**DESCRIPTION:**

Here this is the input design of the page where the user can choose which country he/she wants the content creation recommendation for.

**4.4 OUTPUT DESIGN**

Output design generally refers to the results and information that are generated by the system. For many end-users, output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application. The objective of a system finds its shape in terms of output. The analysis of the objective of a system leads to determination of outputs. Outputs of a system can take various forms.

The outputs vary in terms of their contents, frequency, timing and format. The users of the output, its purpose and sequence of details to be printed are all considered. When designing output, the system analyst must accomplish things like, to determine what information to be present, to decide whether to display or print the information and select the output medium to distribute the output to intended recipients.

****

**DESCRIPTION:**

Here this is the output of the page where the user could see the visualized data that can be used to interpret a lot of information.

**5. SCOPE FOR FUTURE DEVELOPMENT**

This system is very flexible so that the maintenance and further amendments based on the changing requirements can be made easily by just adding suitable analysis.

This system can further be developed for real time updation of trends and hits in YouTube by constantly updating the dataset by retrieving and updating the dataset constantly

This can also be later developed into an application for easy execution and availability in other devices making it even more time saving and effortless

**6. CONCLUSION**

The project entitled “Recommendation System for Content Creators” has been successfully finished. The system has been developed with much care that it free from errors. At the same instance it is efficient and less time consuming. The important thing is the system is robust and it is user friendly in it current version which can be improvement much more in the future aspects.

This system will help the content creators to create quality content based on what’s trending and have a positive overall outlook in the platform. It goes through the dataset and compares and analyses every trending video’s statistics to reach into an output which will provide the user with the best possible recommendation on what video to create and post.

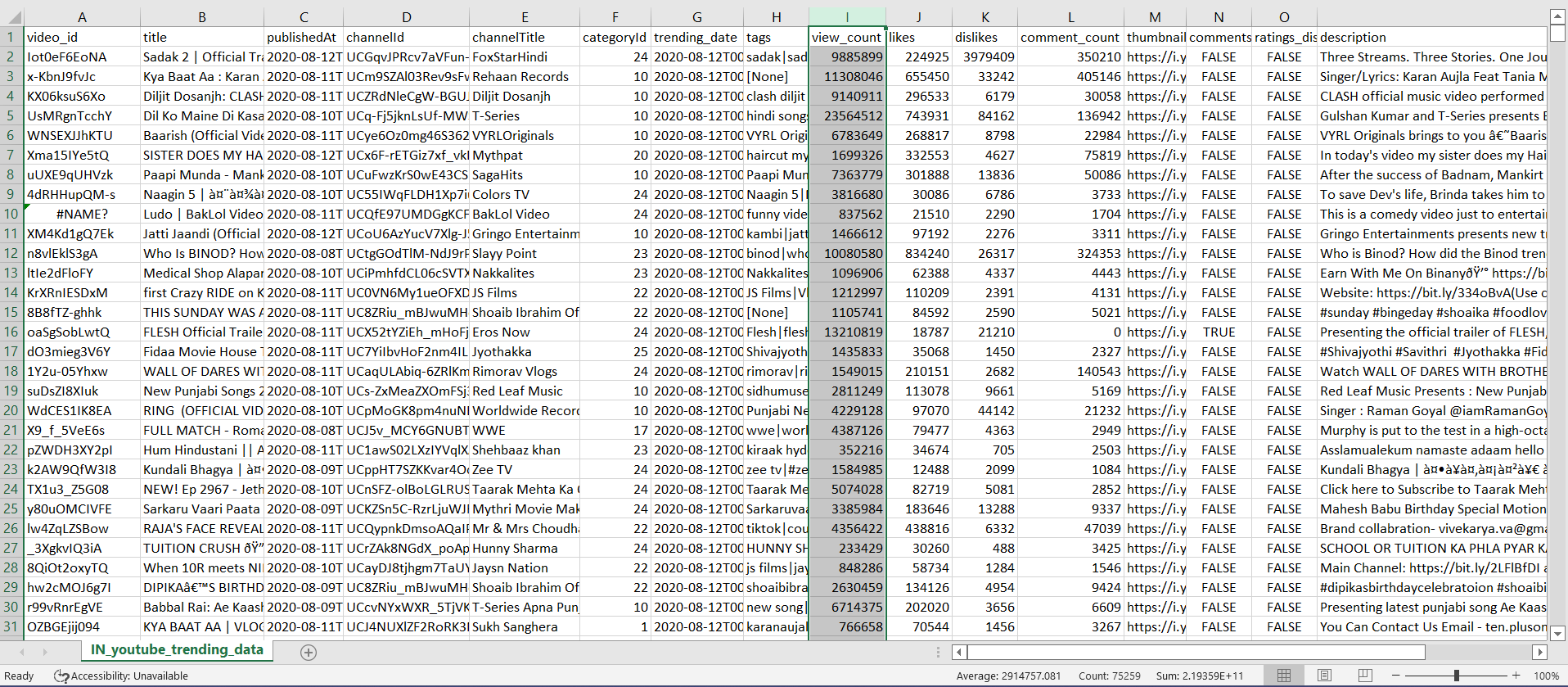
**7.BIBLIOGRAPHY**

1. <https://www.youtube.com>/
2. <https://www.nltk.org/>
3. <https://streamlit.io/>
4. <https://jupyter.org/>
5. <https://anaconda.org/>
6. <https://stackoverflow.com/>
7. <https://www.kaggle.com/>
8. https://scikit-learn.org/

**8.APPENDIX**

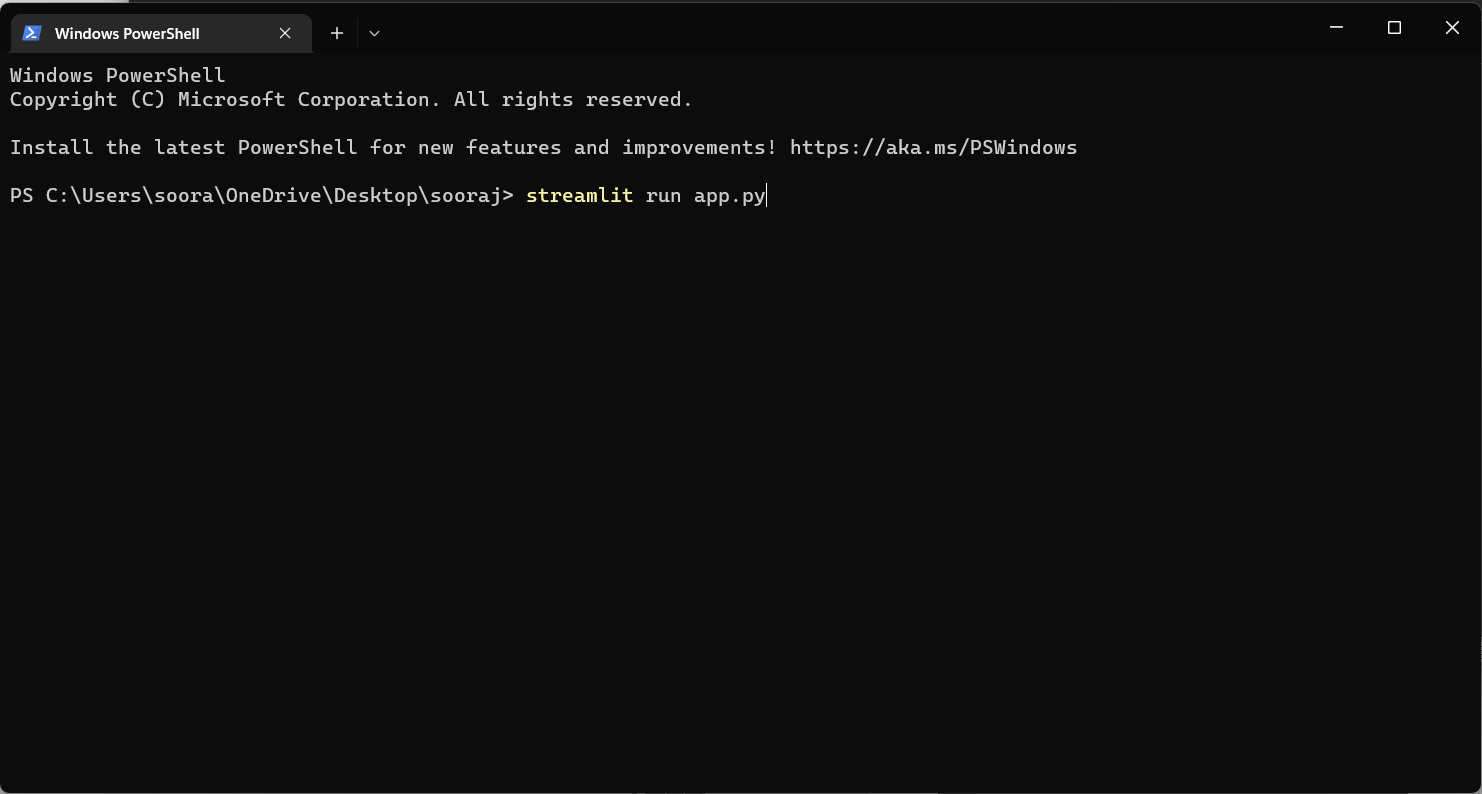
**8.1 SCREEN SHOTS:**

**DATASET:**

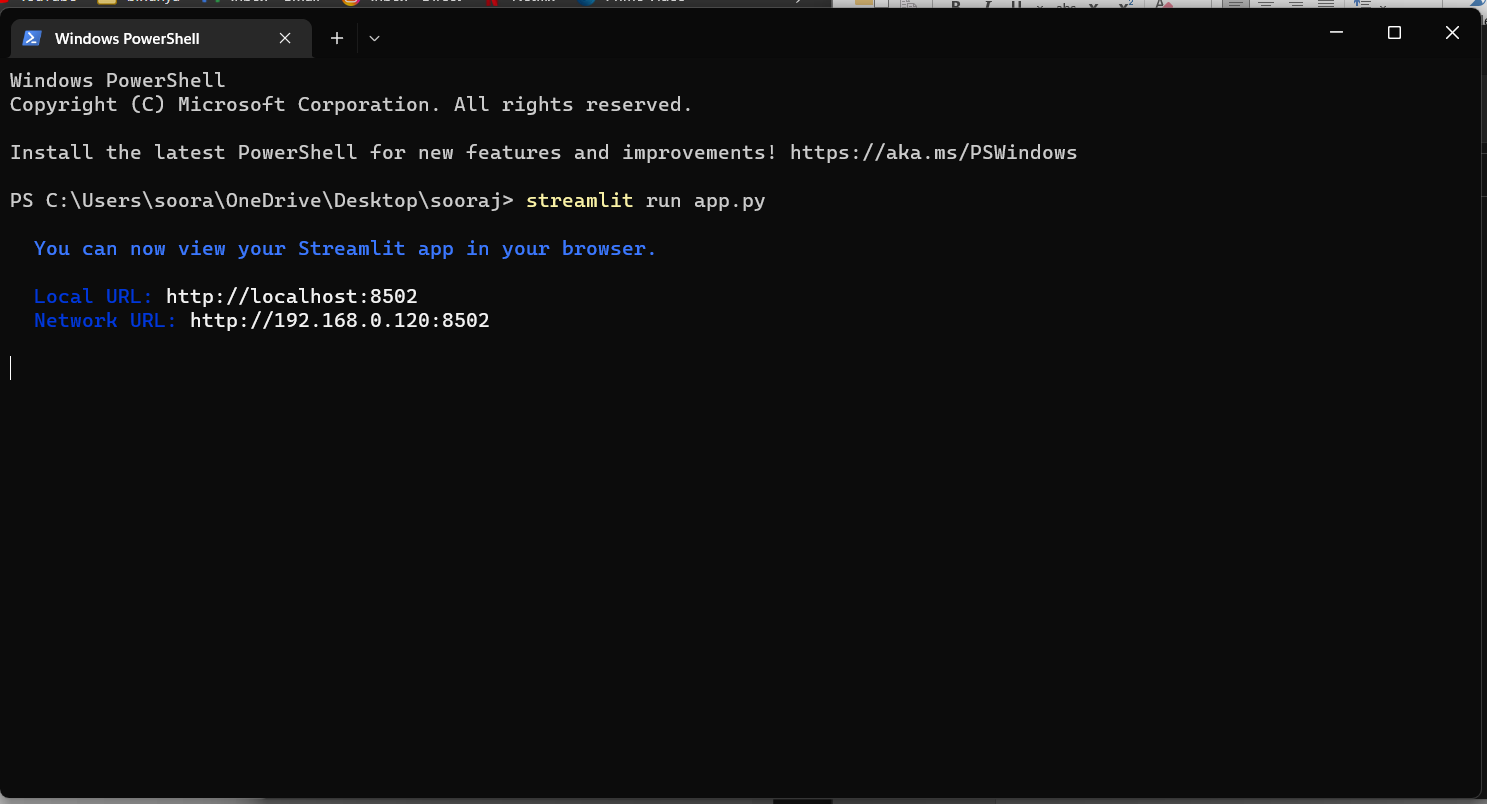
****

**EXECUTION:**

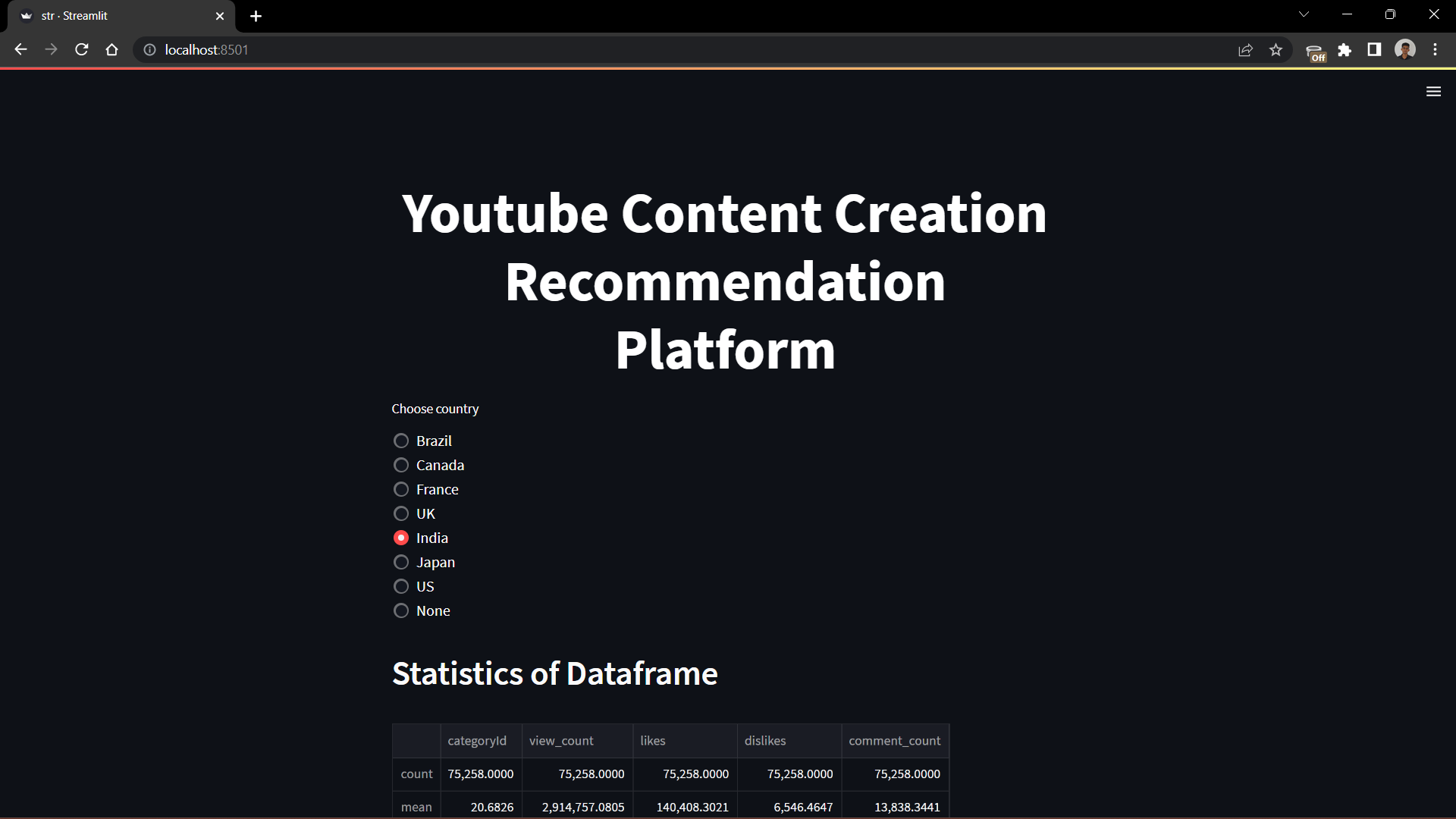
#running streamlit code

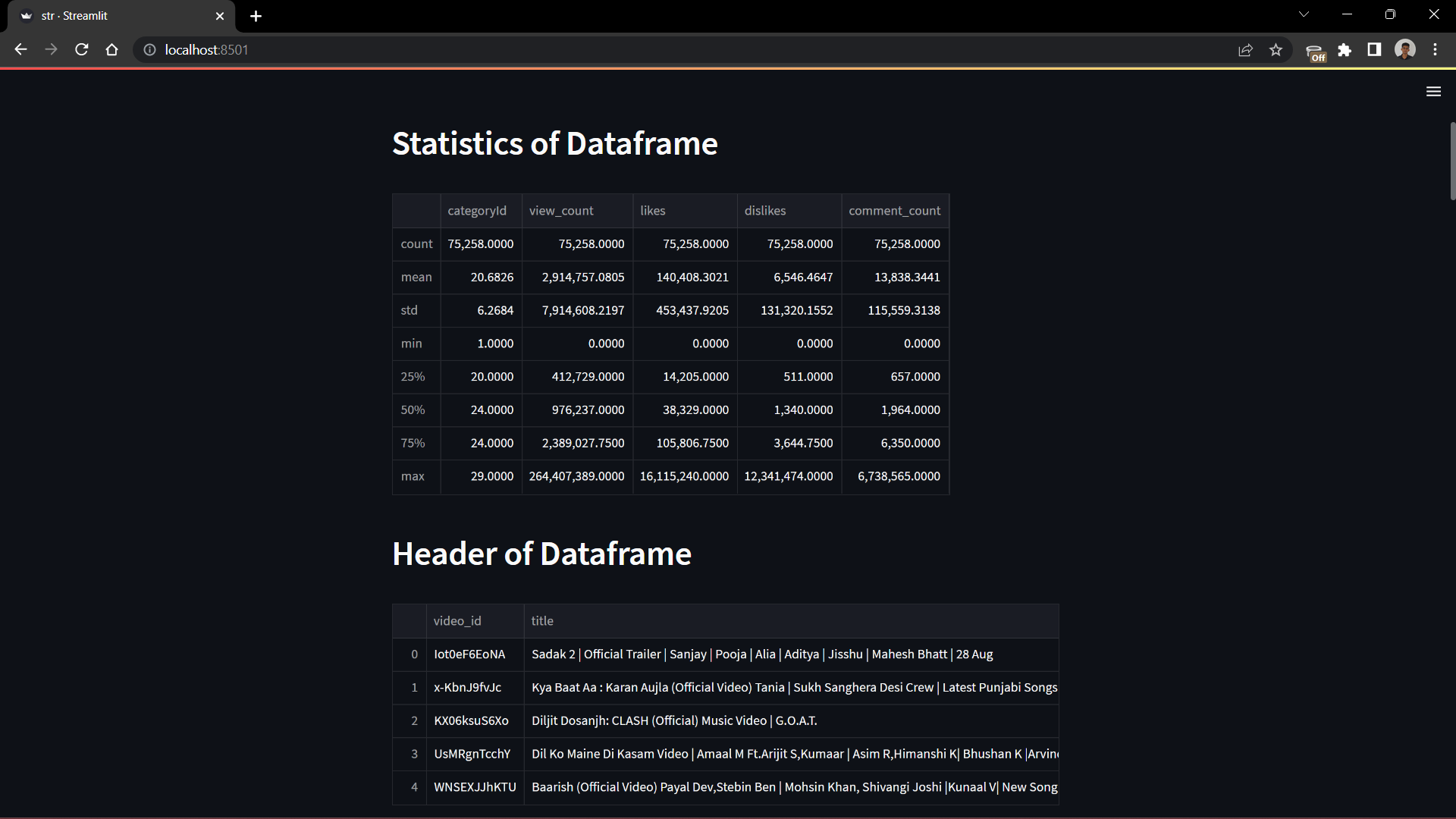
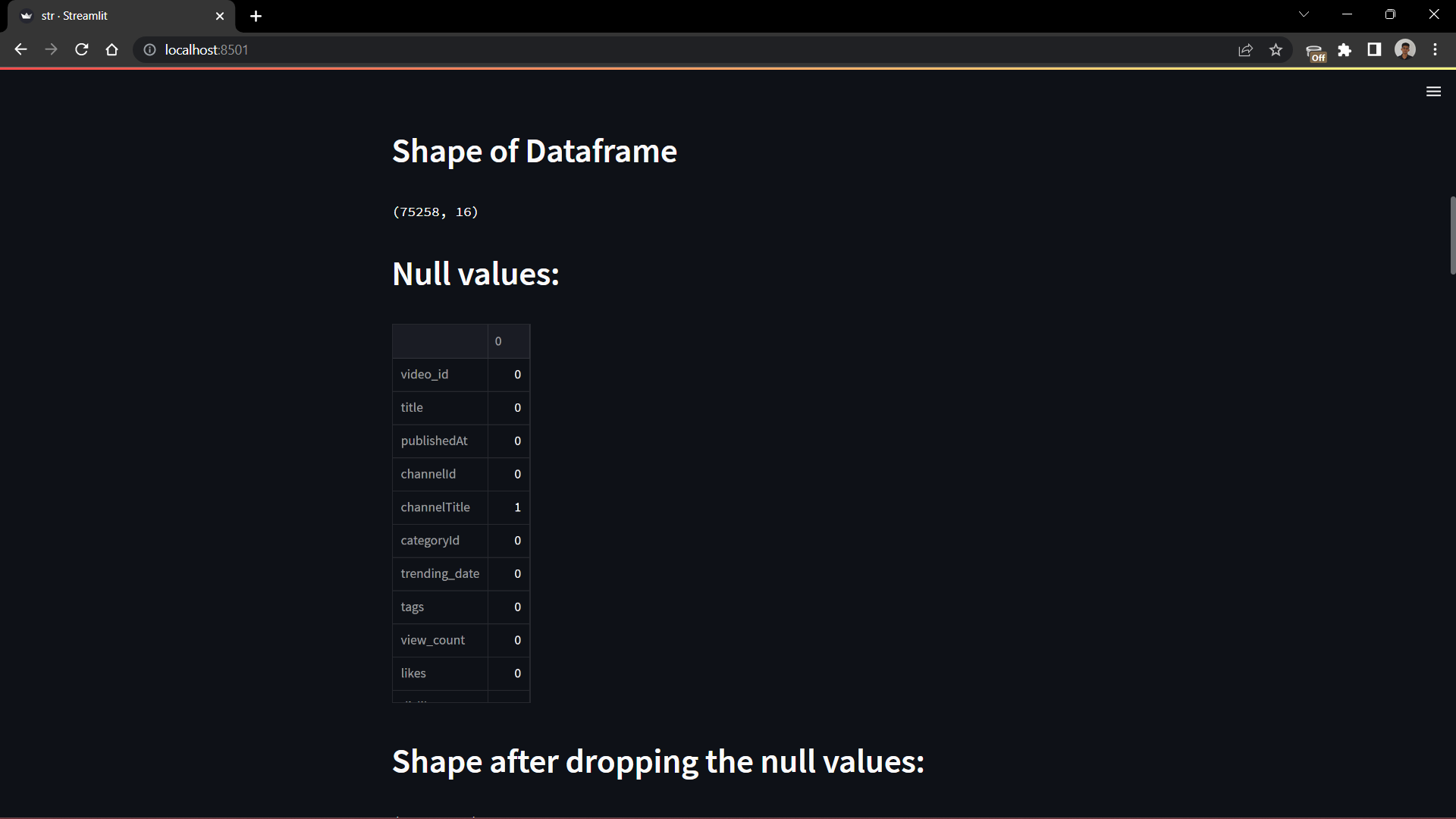
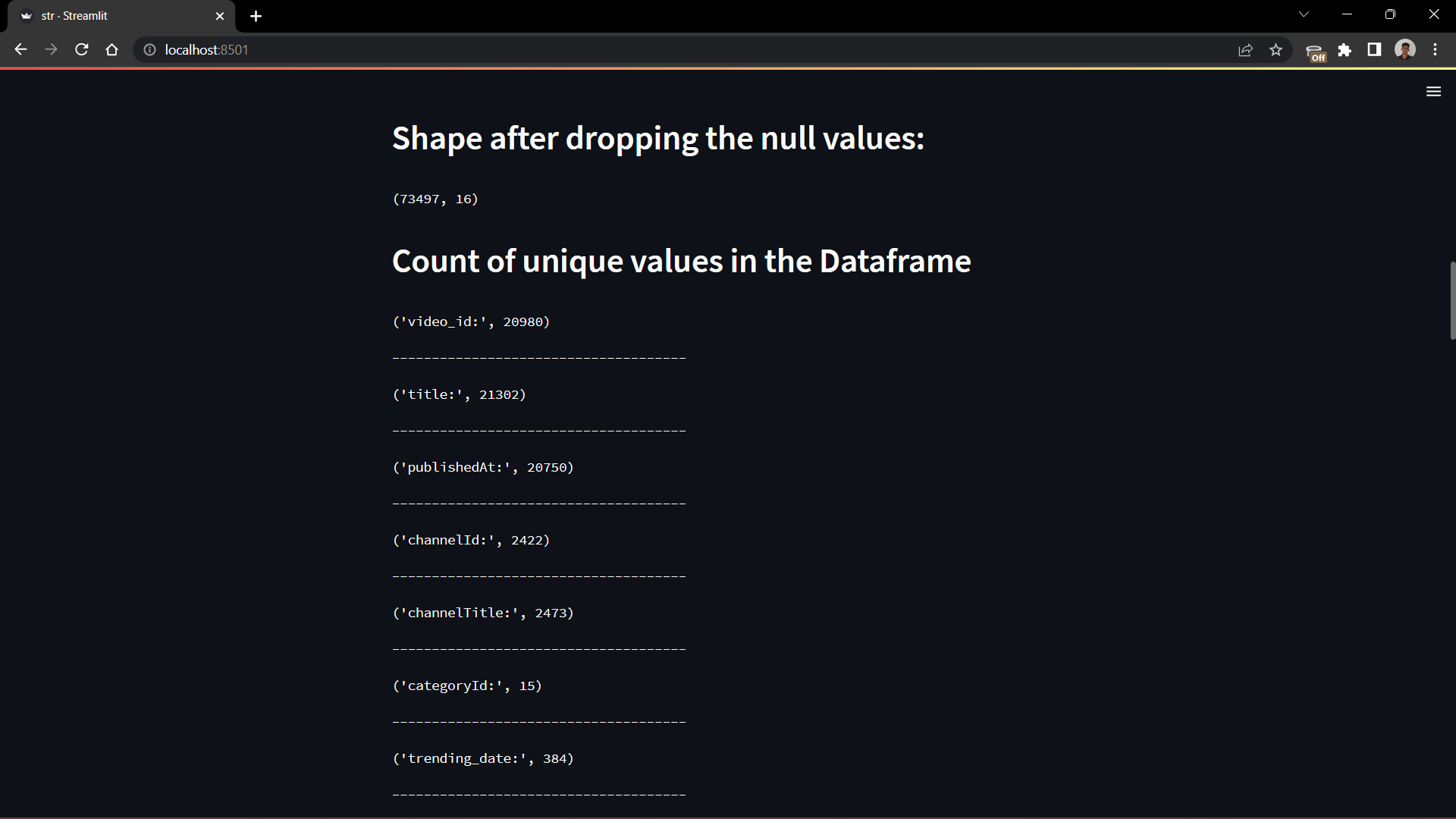
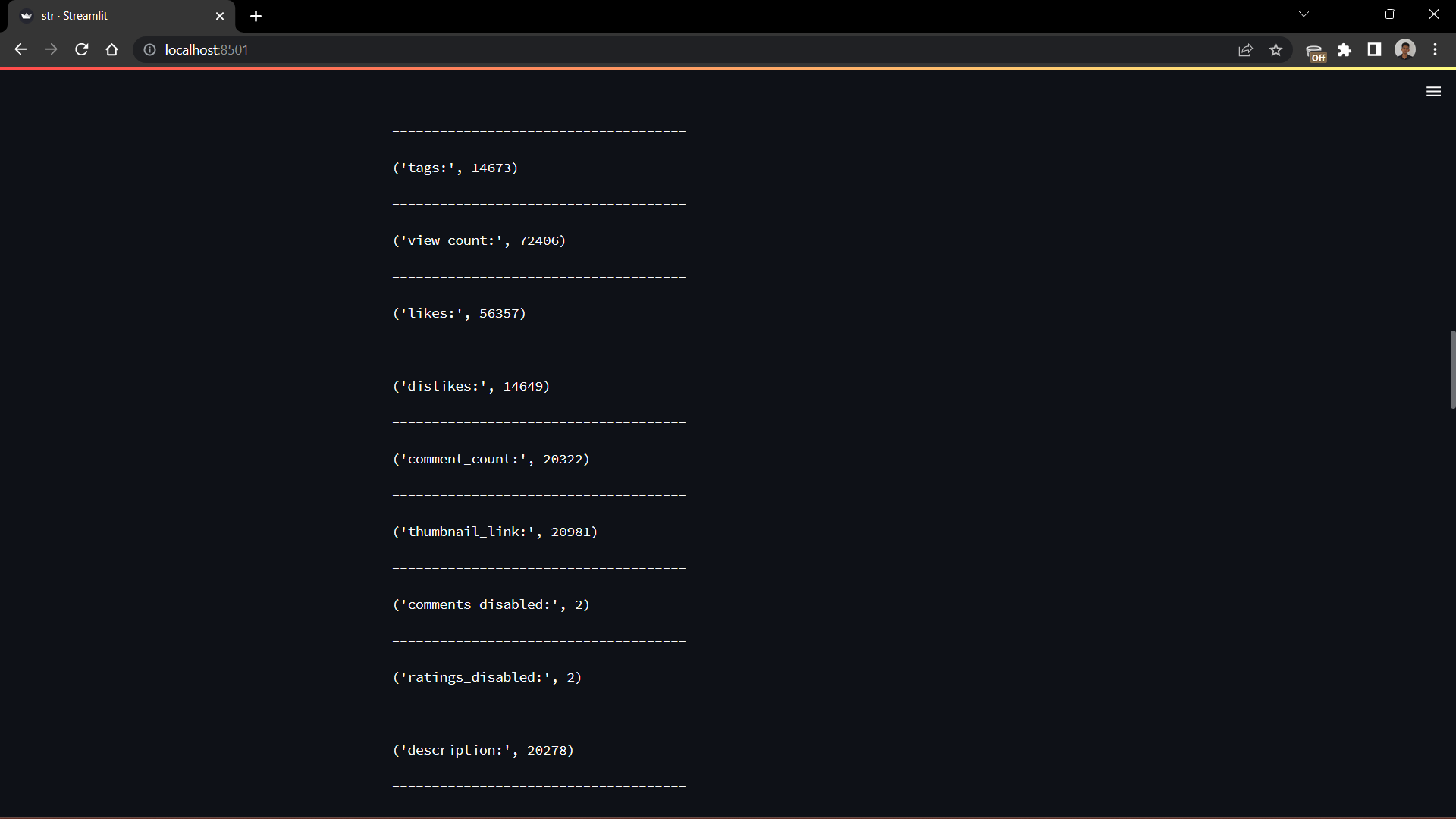
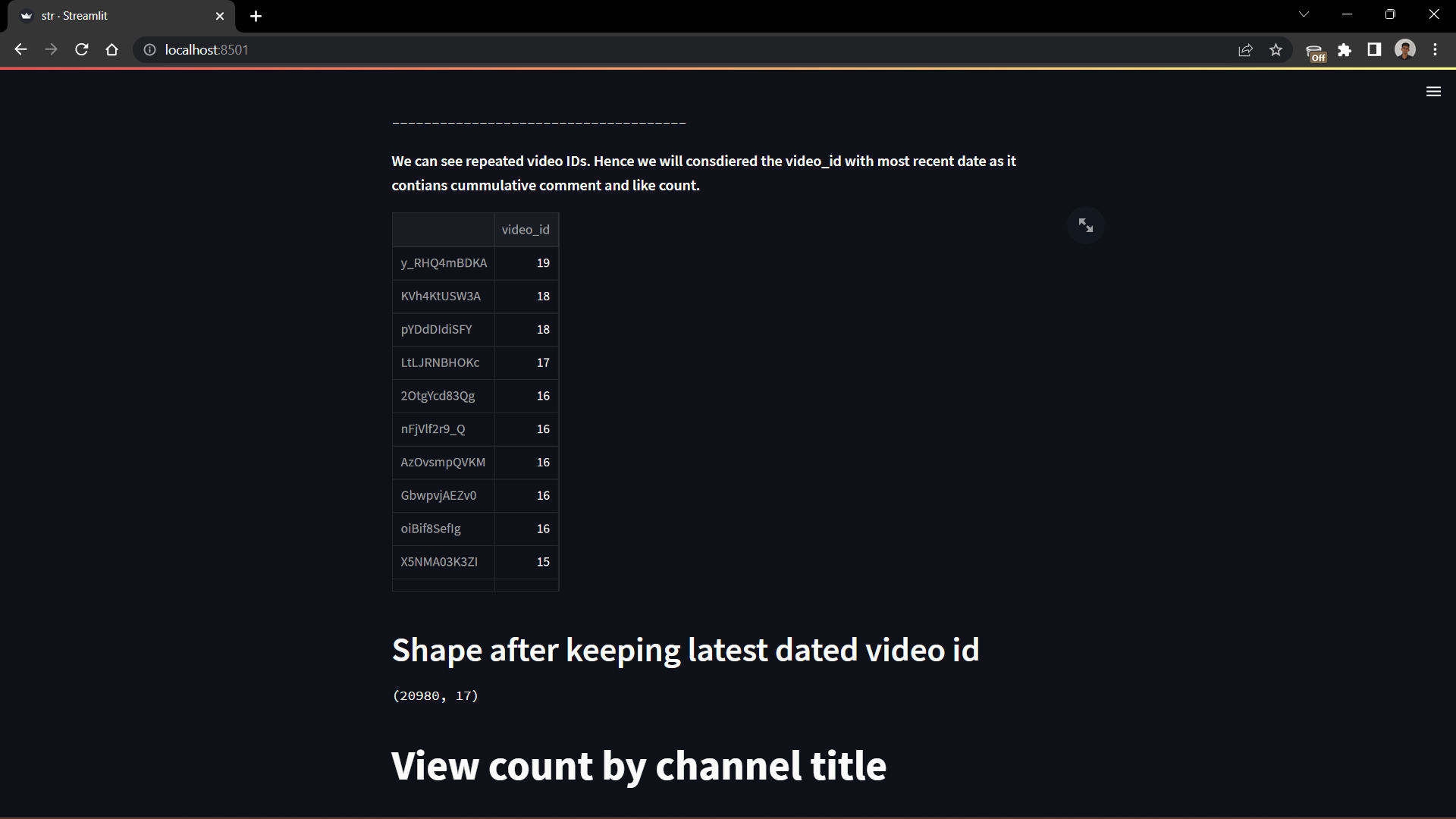
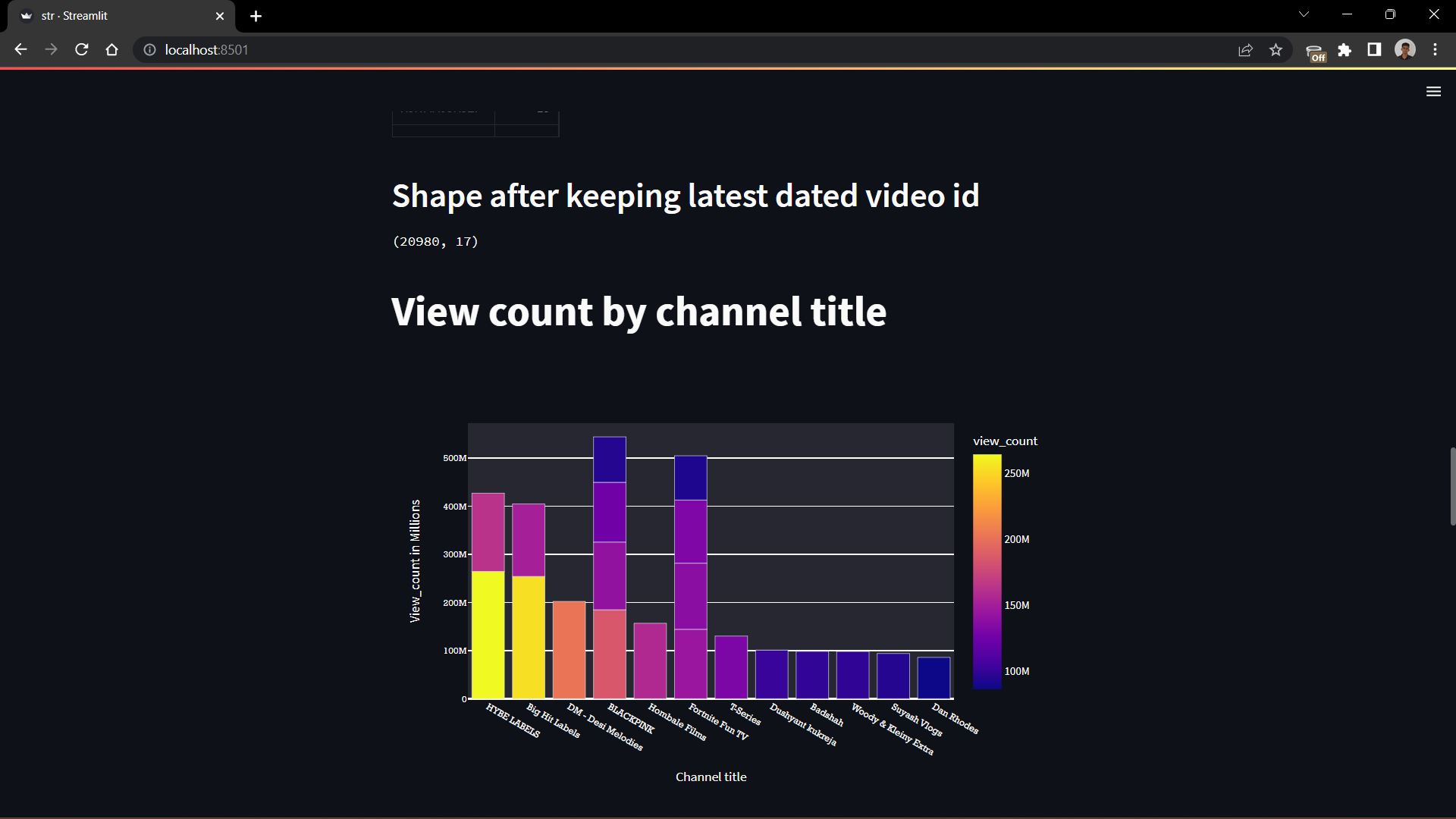
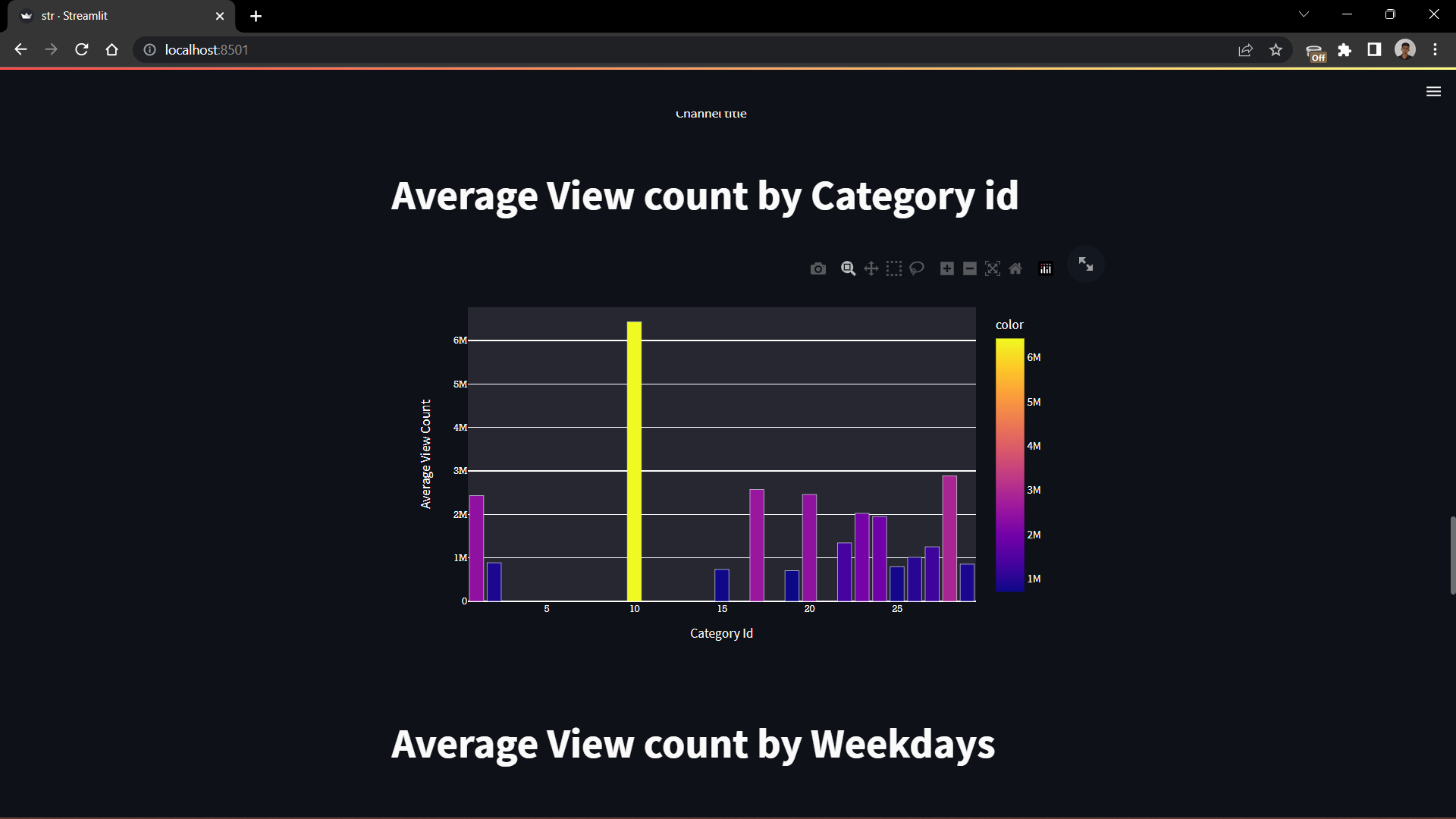
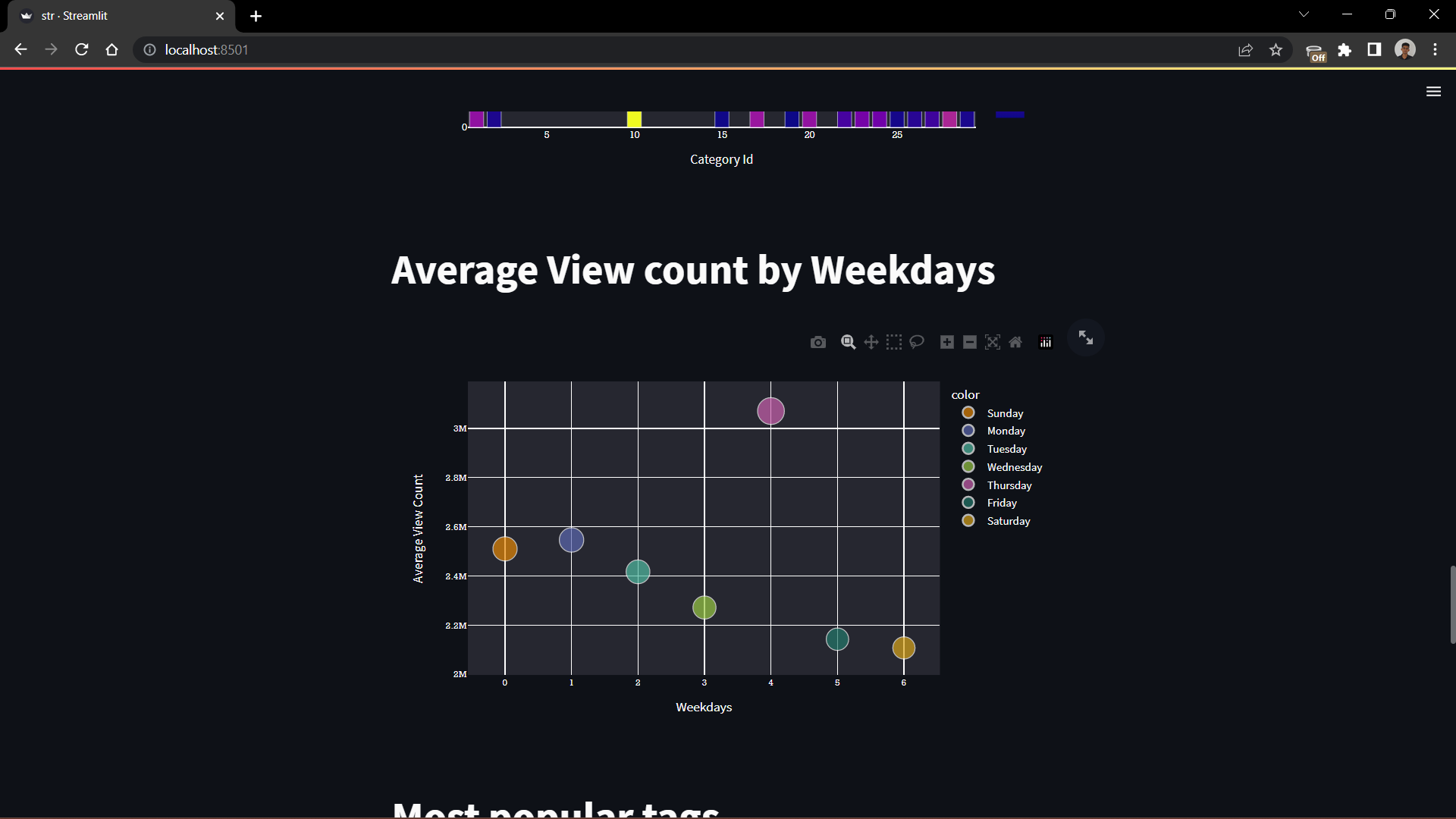
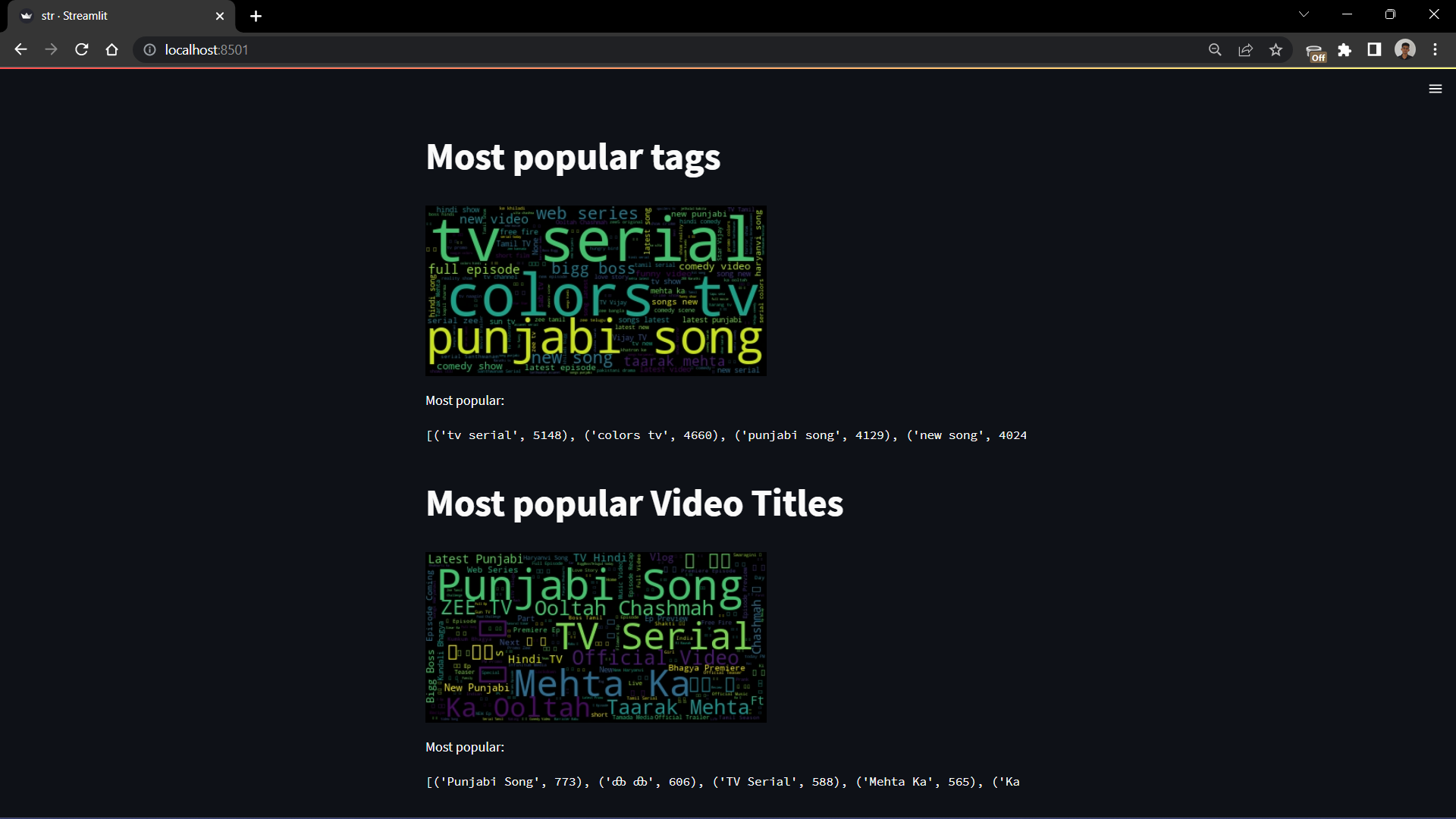
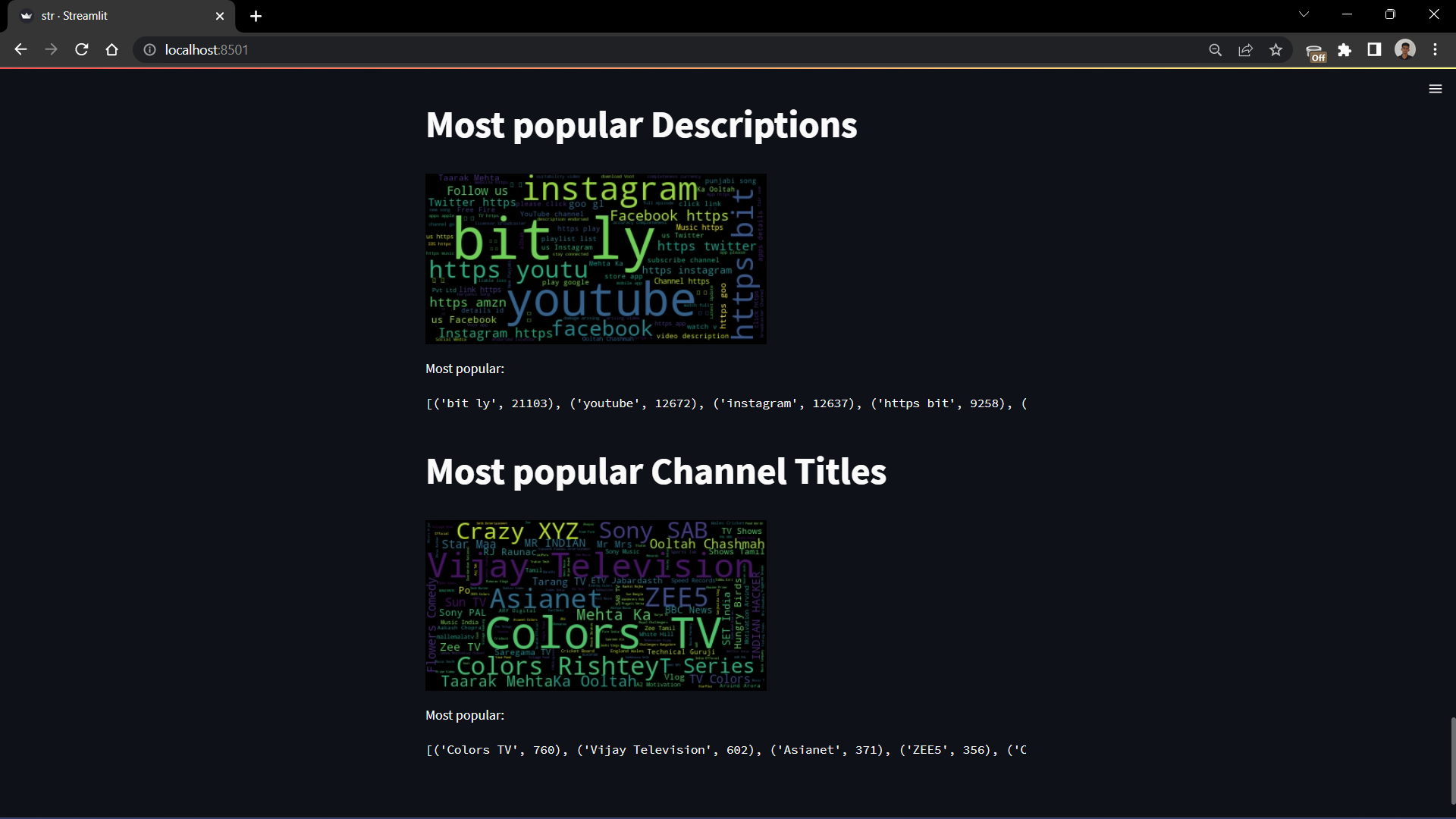
****

#link to open the website



**OUTPUT:**

****

****         

**8.2 SAMPLE CODINGS:**

app.py:

import streamlit as st

import numpy as np

import pandas as pd

import os

import matplotlib.pyplot as plt

import seaborn as sns

from nltk.corpus import stopwords

from nltk.tokenize import word\_tokenize

import nltk

#Visualization libraries

from plotly.offline import init\_notebook\_mode, iplot

import plotly.express as px

import plotly.graph\_objs as go

import plotly.offline as py

# Disable warnings

import warnings

from wordcloud import WordCloud, ImageColorGenerator

from PIL import Image

warnings.filterwarnings('ignore')

st.markdown("<h1 style='text-align: center; color: white; font-size: 60px;'>Youtube Content Creation Recommendation Platform</h1>", unsafe\_allow\_html=True)

code=['Brazil','Canada','France','UK','India','Japan','US','None']

country\_= st.radio("Choose country",code, index=7)

if country\_ =='Brazil':

country='BR'

elif country\_ == 'Canada':

country='CA'

elif country\_ == 'France':

country='FR'

elif country\_ == 'UK':

country='GB'

elif country\_ == 'India':

country='IN'

elif country\_ == 'Japan':

country='JP'

elif country\_ == 'US':

country='US'

else:

st.write('Choose a country')

st.title("")

st.title("")

st.title("")

st.title("")

st.title("")

st.title("")

st.title("")

st.title("")

st.title("")

st.title("")

st.title("")

st.title("")

st.title("")

data=pd.read\_csv("C://Users//soora//Downloads//Countrywise\_youtube\_trending\_data//"+country+"\_youtube\_trending\_data.csv")

st.header('Statistics of Dataframe')

st.write(data.describe())

st.header('Header of Dataframe')

st.write(data.head())

st.header('Shape of Dataframe')

st.text(data.shape)

st.header('Null values:')

st.write(data.isna().sum())

st.header('Shape after dropping the null values:')

data=data.dropna()

st.text(data.shape)

st.header("Count of unique values in the Dataframe")

for i in data.columns:

x=i+":",len(data[str(i)].value\_counts())

st.text(x)

st.text("-------------------------------------")

st.markdown("\*\*We can see repeated video IDs. Hence we will consdiered the video\_id with most recent date as it contians cummulative comment and like count.\*\*")

data["video\_id"].value\_counts()[:20]

data['trending\_date']= pd. to\_datetime(data['trending\_date'])

data["month"]=pd. DatetimeIndex(data["trending\_date"]).month

data["day"]=pd. DatetimeIndex(data["trending\_date"]).day

data["week"]=pd. DatetimeIndex(data["trending\_date"]).week

data=data.drop(columns=["thumbnail\_link","channelId"],axis=1)

fdata=data.sort\_values(by="trending\_date").drop\_duplicates(subset=["video\_id"], keep="last")

st.markdown("## Shape after keeping latest dated video id")

st.text(fdata.shape)

data=fdata.sort\_values(by=['view\_count'],ascending=False)

fdata=fdata.sort\_values(by=['view\_count'],ascending=False)

st.title('View count by channel title')

top20views = fdata[:20]

fig = px.bar(top20views, x='channelTitle', y='view\_count',color='view\_count', hover\_data=['view\_count',"title"])

fig.update\_xaxes(title\_text='Channel title',title\_font = {"size": 14},tickfont=dict(family='Rockwell', size=10))

fig.update\_yaxes(title\_text='View\_count in Millions',title\_font = {"size": 14},tickfont=dict(family='Rockwell', size=10))

st.write(fig)

st.title('Average View count by Category id')

cat\_count=fdata.groupby("categoryId")["view\_count"].mean()

fig = px.bar(cat\_count, x=cat\_count.index, y=cat\_count.values ,color=cat\_count.values)

fig.update\_xaxes(title\_text='Category Id',title\_font = {"size": 14},tickfont=dict(family='Rockwell', size=10))

fig.update\_yaxes(title\_text='Average View Count',title\_font = {"size": 14},tickfont=dict(family='Rockwell', size=10))

st.write(fig)

st.title('Average View count by Weekdays')

fdata['publishedAt']= pd. to\_datetime(fdata['publishedAt'])

fdata["Published\_day"]=pd. DatetimeIndex(fdata["publishedAt"]).day

fdata["Published\_week"]=pd. DatetimeIndex(fdata["publishedAt"]).week

fdata["Published\_time"]=pd. DatetimeIndex(fdata["publishedAt"]).time

fdata["Published\_weekday"]=pd. DatetimeIndex(fdata["publishedAt"]).weekday

day\_count=fdata.groupby("Published\_weekday")["view\_count"].mean()

color\_code=["Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday"]

fig = px.scatter(day\_count, x=day\_count.index, y=day\_count.values, color=color\_code,color\_discrete\_sequence=px.colors.qualitative.Vivid,size=day\_count.values\*10)

fig.update\_xaxes(title\_text='Weekdays',title\_font = {"size": 14},tickfont=dict(family='Rockwell', size=10))

fig.update\_yaxes(title\_text='Average View Count',title\_font = {"size": 14},tickfont=dict(family='Rockwell', size=10))

st.write(fig)

stop\_words = set(stopwords.words('english'))

if country=='BR':

stop\_words = set(stopwords.words('spanish'))

elif country=='FR':

stop\_words = set(stopwords.words('french'))

fdata['title'] = fdata.title.apply(lambda x: word\_tokenize(x))

fdata['title'] = fdata.title.apply(lambda x: [w for w in x if w not in stop\_words])

fdata['title'] = fdata.title.apply(lambda x: ' '.join(x))

fdata['tags'] = fdata.tags.apply(lambda x: word\_tokenize(x))

fdata['tags'] = fdata.tags.apply(lambda x: [w for w in x if w not in stop\_words])

fdata['tags'] = fdata.tags.apply(lambda x: ' '.join(x))

fdata['description'] = fdata.description.apply(lambda x: word\_tokenize(x))

fdata['description'] = fdata.description.apply(lambda x: [w for w in x if w not in stop\_words])

fdata['description'] = fdata.description.apply(lambda x: ' '.join(x))

#fdata = pd.read\_csv('file1.csv')

pop\_words=[]

def WordCloudfunction(title,text):

cloudtext=' '.join(fdata[text].tolist())

sns.set(rc={'figure.figsize':(16,10)})

wordcloud = WordCloud().generate(cloudtext)

plt.figure()

plt.imshow(wordcloud, interpolation="bilinear")

plt.axis("off")

plt.margins(x=0, y=0)

plt.title(title,size=24)

st.image(wordcloud.to\_array())

text\_dictionary = wordcloud.process\_text(cloudtext)

# sort the dictionary

word\_freq={k: v for k, v in sorted(text\_dictionary.items(),reverse=True, key=lambda item: item[1])}

#use words\_ to print relative word frequencies

rel\_freq=wordcloud.words\_

#print results

pop=list(word\_freq.items())[:5]

pop\_words.extend(pop)

st.markdown("Most popular:")

st.text(pop)

st.title('Most popular tags')

WordCloudfunction('Tags','tags')

st.title('Most popular Video Titles')

WordCloudfunction('Video title','title')

st.title('Most popular Descriptions')

WordCloudfunction('Description','description')

st.title('Most popular Channel Titles')

WordCloudfunction('channelTitle','channelTitle')