

# LinkedIn Post

Hello Data Folks!


I'm excited to share my data analytics portfolio project, where the user can get any YouTube channel data (worldwide) with just a click\*.

## Problem Statement

School of Data Science, an online learning platform, is nearly it's first cohort and wants to test the students' skills by challenging them to provide interesting insights of their favorite YouTube channels.

Objective is to create a Streamlit web app where the students can provide their own API key and their favorite channels channel id and download the data in a CSV for the students to analyze and provide insights.

## Accessible Worldwide:

With a simple API key and channel ID, you can explore channels from around the world! 

## Tech Stack

Python, Streamlit, Pandas, YouTube API

- **Python** – The project's foundation, harnessing Python's versatility for robust processing and analysis
- **YouTube API** – Fueling data retrieval, enabling seamless interaction with YouTube to extract channel statistics, playlist details and video metadata
- **Pandas** – Powering efficient data manipulation and transformation, essential for organizing and structuring YouTube data.
- **Streamlit** – Elevating user experience, effortlessly converting data scripts into an interactive web app for intuitive exploration of YouTube Analytics.

## Key Features:

- Extracts channel statistics: views, subscribers, and video count.
- Retrieves playlist details and video metadata.
- Streamlit web app for a user-friendly experience.
- Download data in CSV format for your analysis.

## Important Links:

YouTube Tutorial –


Streamlit Web App –

Presentation Slides –

Git Hub –

Article –

## Thanks for the support!

This project was a labor of love, and I'm thrilled to share it with the community. Let's keep exploring the data-driven world together! 

#DataScience #YouTubeAPI #Python #DataAnalytics #PortfolioProject #Streamlit #Pandas

**Video Summary: How to use this app**

# RESUME

## VERSION 1

### YOUTUBE DATA GATHERING PROJECT

**Streamlit Web App Link – [Click here](#)**

- Built a rapid Streamlit web app for global YouTube channel data retrieval in under 15 seconds, enhancing user efficiency
- Solved critical edge cases, including API key authentication and channel ID errors, ensuring robust functionality. Resolved complexities within 2 days, enhancing app reliability
- 30 users reported an average rating of 4.8 out of 5 and highlighted a substantial reduction in data collection time, enabling a more focused approach to critical data cleaning tasks

## VERSION 2

### YOUTUBE DATA GATHERING PROJECT

**[Streamlit Web App](#) | [GitHub](#) | [Blog Post](#)**

- Engineered a rapid Streamlit web app for global YouTube channel data retrieval in under 15 seconds, enhancing user efficiency
- Solved critical edge cases, including API key authentication and channel ID errors, ensuring robust functionality. Resolved complexities within 2 days, enhancing app reliability
- Demonstrated proficiency in data analytics, Python programming, YouTube API integration, and user-friendly data interface development

## VERSION 3

### YOUTUBE DATA GATHERING PROJECT

**[Streamlit Web App](#) | [GitHub](#) | [Blog Post](#)**

- Architected an efficient Streamlit web app for global YouTube channel data retrieval
- Resolved critical edge cases, ensuring robust functionality and better user experience
- Showcased expertise in data analytics, Python, YouTube API integration, and user-friendly app

# INTERVIEW QUESTIONS

## 1. Tell me about this YouTube Analytics Project, you mentioned in your Resume?

- YouTube Analytics Project is a primarily a data collection project.
- The main objective of the project is to streamline the data collection process of any YouTube Channel using API and convert the JSON data provided by the API into a CSV file, ready for the analyzing it and extracting insights.
- I created an interactive Streamlit web app where users can provide their own API key and the favorite channel ID, and download the data in a CSV Format.

## 2. Can you explain the data retrieval process?

I used 3 API end points from the YouTube API, specifically channels end point, playlist end point, & videos end point.

I wrote separate python functions for every end point.

- a. I created a function for channels end point where the API key & channel id are provided as input parameters and return channel statistics like total views, total subscribers, total number of videos uploaded and playlist id as a dataframe.
- b. The Playlist id from the previous function is used as an input for another function and returned all the unique video ids of every video of that channel in a list.
- c. The third and final function takes the video ids in a comma separated string format as input and returns the video meta data like views, likes, comments, title, duration etc., of every video as a pandas dataframe.
- d. Finally, the data frame is converted int to a CSV file.

## 3. Can you describe the main features of Streamlit Web App

- The web app is very minimalistic in its usage, the API key & channel ids are embedded in a Streamlit form by utilizing st.form function from Streamlit library.
- After entering the valid API key and channel id, the user is able to download the data as a CSV.
- I also dealt with the edge cases like
  - incorrect API key,
  - incorrect channel id,
  - no API key,
  - no channel id and
  - also, channel with no videos
  - and an appropriate error message will be provided to the user in these cases.

#### 4. What Challenges did you face while working on this project and how did you overcome it?

- **Data Retrieval Limitations:** The YouTube API limits data retrieval to a maximum of 50 items at a time, presenting a challenge when dealing with channels containing more than 50 videos or items.
- **Next Page Token Utilization:** To overcome the limitation mentioned above, the concept of a “*next page token*” provided by the YouTube API was utilized. This allowed for the retrieval of additional details beyond the initial 50 items
- **Dynamic While Loop:** As the number of items varies from channel to channel, a ``while`` loop was employed during the data retrieval process. This loop ensured comprehensive data collection while handling varying item counts.
- **Exception Handling:** Robust exception handling was implemented using ``try`` and ``except`` blocks. This covered scenarios such as incorrect API keys, missing API keys, incorrect channel IDs, missing channel IDs, and cases where a channel has zero videos.