Verbal summary of the implementation:

\*The code retrieves data from the YouTube Data API, stores it in a MongoDB database, and later transfers some of that data to a PostgreSQL database for analysis.

\*Importing necessary libraries, including modules for working with Google's YouTube Data API, MongoDB, PostgreSQL, and Streamlit for building web interfaces.

\*Setting up the YouTube Data API by providing an API key and specifying the API service and version.

## Defining functions for various tasks:

dropdownlist(): Creates a list of channel names for selection.

duration(data): Converts video duration in ISO 8601 format to hours.

get\_channel\_details(channel\_id): Retrieves details about a YouTube channel using the API.

get\_playlist\_details(playlist\_id): Fetches video IDs from a channel's playlist ID.

get\_video\_details(video\_ids): Retrieves details of videos using their IDs.

get\_comment\_details(video\_ids): Gathers comments on videos.

data\_to\_mongo(data): Stores channel, video, and comment details in a MongoDB database.

retrieve(data): Retrieves data from MongoDB and inserts it into a PostgreSQL database.

analysis(data): Executes SQL queries and displays results in Streamlit.

Setting up a Streamlit web application with sections for data collection, data conversion, and data analysis.

Conclusion:

In the data collection section, users can input a YouTube channel ID and click a button to retrieve data from the API, which is then stored in MongoDB.

In the data conversion section, users can select a channel name from a dropdown list and click a button to transfer data from MongoDB to PostgreSQL.

In the data analysis section, users can choose from various predefined questions related to the stored data, and the script executes SQL queries to retrieve and display relevant information using Streamlit's interface.

Overall, this script provides a user-friendly way to collect, store, and analyze YouTube channel data using Python and various APIs and databases. It combines multiple technologies to create a data pipeline from web scraping to data visualization.