Community Response

RCOS 24 Spring Project

Vansh Reddy Cheguri

Introduction

Community Response is a chrome extension that analyzes a Youtube video comment section and shows interesting statistics:

- the most frequently used words
- the overall sentiment of the comments,
- the most active commenters
- the geographical distribution of viewers
- Word cloud of the most frequent words

This semester the project had the following objectives:

- Design the overall system architecture
- Complete working on the Backend and setting up Routes
- Design a robust DB schema
- Design the frontend and do preliminary work before the next semester
- Research Hosting options

Technology Stack

Frontend:

React JS

chakra

Backend:

Flask Hugging Face

Database:

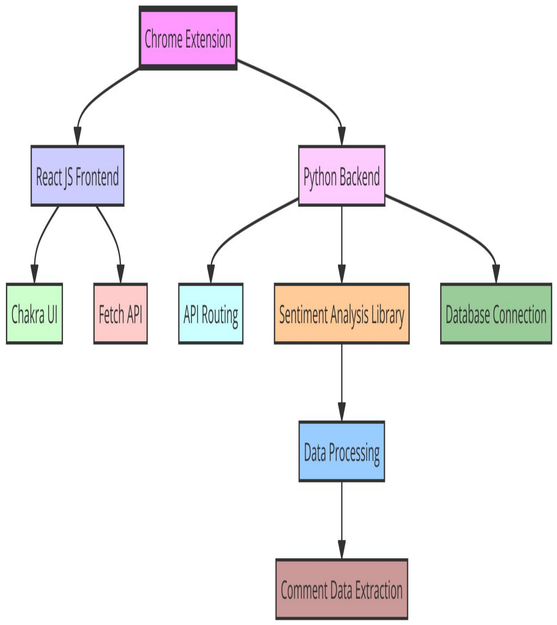
mongoDB®

Utils:

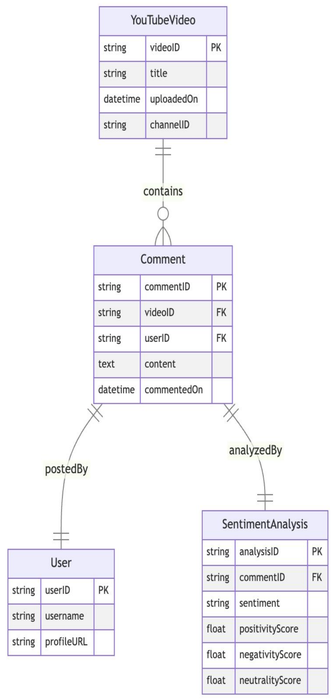
GitHub Poetry

System architecture

The following diagram shows the high level architecture of the project:

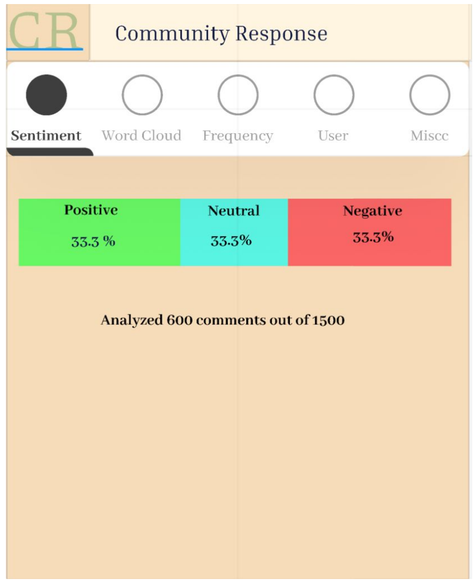


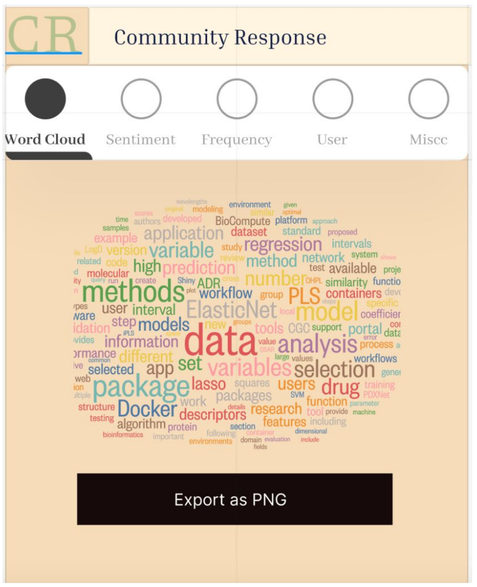
Database Architecture



Frontend Semester Progress

This semester we focused on designing the UI and slowly working on implementing the Frontend. We were able to generate the following using Figma:





Data Extraction

This semester we were able to find multiple ways to extract the Youtube comments.

- Our First method relied on extracting the data by accessing the HTML content from the page. However, this turned out to be more difficult as comments are lazily loaded which meant you cannot extract all the comments. The API currently can extract upto 100 comments using this method.
- The second method relied on utilizing the youtube API. This method proved to be more simplistic but the Youtube API is a paid service. This service is not feasible in the long term.The API supports using the Youtube API temporarily.
- In the future, we are planning on moving to a system where we can load all the comments using HTML method.

Backend Routes

- This semester we developed several backend routes but these are the main routes used and tested:
- **Endpoint:** /api/comments
- **Method:** GET
- **Description:** This route fetches the comments for a specified YouTube video. It might require query parameters like the video ID or pagination details (like page number or comment limit per request).
- **Parameters:**
  - videoId (required): The ID of the YouTube video.
  - page (optional): Page number for pagination.
  - limit (optional): Number of comments per page.
- **Endpoint:** /api/sentiment
- **Method:** POST
- **Description:** Receives a batch of comments and returns the sentiment analysis. This can be designed to accept multiple comments at once to reduce the number of API calls.
- **Body:**
  - comments: An array of comment texts to analyze.

- **Endpoint:** /api/video

- **Method:** GET

- **Description:** Retrieves details of a specific video such as title, author, and description.

Future Objectives

We have the following future objectives in the upcoming semester:

- Develop a better developer experience for working on the Frontend
- Develop a CI/CD pipeline
- Work on training a custom network to perform sentiment analysis based on youtube data for more accurate results.
- Implement the Frontend designs
- Improve Backend performance
- The bottleneck in the process is the inference time, improve the inference time