Team Sanguine Thirst / reelFeels / https://github.com/mmacquarrie/reelfeels

Members: Zack Bebbington, Megan MacQuarrie, Matt Rubin, Humad Syed

Team Write Up

Application Overview

The idea is to have a platform where content creators, similar to those on YouTube or Twitch, can either upload or share their content for users to view. Apart from the usual metrics such as user views and likes/dislikes, the content creators will be able to see (either in graph form or via some other visual representation) how the users felt while viewing content. This is extremely valuable information for content creators as it allows them to determine weaknesses in their content and equips them with the data necessary to produce better content. We have not deviated from our proposal.

Design Overview

Our application has four major dynamic views: search, explore, user profiles, and individual videos. The search page displays a list of videos whose titles and/or descriptions match the query read from the url. The explore page displays a list of new videos by checking if their upload date was within the last week. It also displays a list of the most popular videos. This is done by ordering all videos by the number of views field and limiting the number of results.The title, uploader, top emotion, and number of views are all pulled from fields in the Video model and displayed for each video on the search and explore pages. Both of these pages are filterable by top emotion, which is calculated by a function in the Video model.

Individual videos are accessed by the url ‘video/{id}’. Here, we pull the video title, description, and uploader from the Video model and the set of comments associated with that video from the Comment model. The global stats come from some calculations involving the set of view instances for that video, and the individual sessions are session/user-based, to be done later. Individual users are accessed by the url ‘profile/{id}’. Here, we pull the user’s username, profile picture, top emotion, and overall emotions from the User model. We then display a list of their uploaded videos by querying videos that match their uploader id.

Problems/Successes

There were three main problems that we encountered while working on the second portion of the project. The first two issues surrounded the content that we wished to display to our users. We were unsure as to how to formulate the maths needed to determine a user’s/video’s top emotion, the algorithm that we were going to utilize to determine which videos should be flagged as ‘popular’, and how we should get a users’ emotion values for ‘happiness, sadness, disgust, anger, and surprise’. Aside from the math/algorithm design, we also hit a rough spot in coordinating the management and addition of mock data to the database file.

The greatest success that we had in project two was the implementation of our views.py and url.py files. Thanks to the mozilla documentation and online resources, it was easy for us to get these ready to interact with our database models. On top of this, the problems that we did encounter were easily handled thanks to the great amount of communication that our team continues to have. Readily responding to others questions on slack outside of class made it made it easy to progress through even the tough parts of project 2.

Individual Write Ups

Zach Bebbington: 25%
I edited the views.py, urls.py and the user-profile.html files so that the profile views would display dynamic data. The user profile page displays some of the user's personal information, such as their username, profile picture, and all of the videos that they've uploaded. Aside from this I also worked to touch up our models, to account for changes that needed to be made while we progressed through project 2. I also tested each of our pages to ensure that the views were working and that there we properly interacting with our database.

Megan MacQuarrie: 25%
I worked on turning the explore view into a dynamic page, which involved changes to views.py and the explore template. This page displays a list of the newest videos and a list of the most popular videos. I also wrote a small script to filter videos by top emotion and a method to grab youtube thumbnails given the video url, which will be used on other pages. Along with everyone else, I worked on finalizing the models, adding mock data, and testing our views.

Matt Rubin: 25%
The bulk of my programming work for Project #2 was done in setting up the dynamic view for the individual video page. Most of this work was done by both writing a view function in views.py and by editing the video template to use the Django template tags/code. I also made miscellaneous other changes/additions to some other project files, such as models.py and settings.py, to get various specific features to work properly along the way.

Humad Syed: 25%
I worked on creating a dynamic view for the search page. The views.py file had to be modified to accept a search query, and also handle cases when a search query was not provided. The latter was changed to render the explore page instead of the search page. The search page currently displays videos that contain the user's query. I also added mock data and tested that they worked with the views.