Alec Bielanos
IGME430 – Professor Cody Van De Mark
October 2017
Project 1 – Web API

What your site does and its purpose:

My website and API allow multiple users to share and rate YouTube videos from a list that they choose and collaborate on. Users are able to insert a youtube link and caption of their choosing into the list. They may then refresh the list to see a list of all videos that have been added by all users. Once the list is displayed, the user may then click on any of the video links to open the corresponding video in the embedded player adjacent to the list. Once the video loads, users will have a choice to upvote or downvote the video via the thumbs up and thumbs down buttons below the player; this will affect the public rating shown in the list. It should be noted that the user can only vote once on each video – after they vote, the thumb buttons disappear after being clicked and, if the same video is loaded, clicking the thumb buttons again will not count toward the vote.

What part of your app does the API handle:

The API under the hood of the website handles the storage of video/caption data, the addition/updating of the videos, and handles the voting made on each video. When the user presses the "Add Video" button, the link and caption information will be sent via POST request to the server, which will then parse it and validate it. Currently, only links that begin with 'youtube.com' will be considered valid, or else a 400 (bad request) error will be thrown. If the video already exists, it will be updated with the new information and return the appropriate response codes (201, 204). In any case, the e-tag is updated and a new hash is generated.

When the user wants to get the list of videos, a GET request is sent to the server, which will retrieve the list of videos/ratings ONLY if the e-tag has changed from the one the user currently has.

What went right and what went wrong:

The back end turned out to be really solid. I'm happy with the way the architecture turned out and feel that it's a great base to start adding more functionality onto with very little or no changes to the existing code.

Saving the CSS for the end was definitely a limiting factor in this project's success. It was more of a time management error than something that went wrong in the code – I just underestimated the time it would take to complete.

If you were to continue, what would you do to improve your app:

CSS is definitely the main weakness right now. I didn't have enough time to hone in on the styles or make the best user experience I could have, so more time to do that would certainly help. Additionally, handling the youtube video player error codes would help enrich the user experience. Lastly, I would hook up the video list to a database that wasn't wiped every time the server reset, and potentially look at adding sorting or searching of the list.

How did you go above and beyond:

The addition of cookies was something that we didn't discuss in class at the time of this project. Additionally, the inclusion of the YouTube iframe API was not a requirement of the project but was used to enhance the user experience and add another visual element to the piece.

External Code Dependencies:

This project uses the YouTube iframe API, as mentioned above. This was approved by the professor and includes code available publicly on the API website for setup. All of the code has to be in the global scope for it to work – this was also okayed by the professor