Matthew Fong

Rich Media 2 Project 3 - Tweeter Documentation

**Site purpose:**

For this project, I made a Twitter clone app. It is a small social media app experience where users create an account and can ‘tweet’ or post messages that other users can see or reply to.

**Profit method:**

My profit method for my app is through the use of ads. I display the ads on the right-hand side of the app so that the main app content is still displayed in the center of the screen.

**Templating language/React Use:**

In terms of templating languages, I used handlebars and React. I am using handlebars as the very basic skeleton of the app pages (login, app, and notFound). For React, I am rendering React components throughout the app experience. On the login screen, when I user switches tab views, the page is actually rendering React components for both the login and signup window. On the app page, if the user switches to the password change tab, the page is also rendering a React component to render the Password Change form onto the page. Otherwise, most, if not all, the user interaction features on the main app page uses React to render different views onto the page. Examples include, making/tweeting a message, clicking on the image icon to insert an image file, viewing tweets, clicking the dropdown icon on a tweet to edit/delete that tweet, actually editing a tweet, which will render a change tweet form, deleting a tweet, which renders two more buttons onto the page, and finally, replying to tweets.

**MVC:**

In terms of MVC, I used express, which maps specific requests from the client browser to different functions on the server. Each specific request that comes from the client browser is received on the server-side controllers (Account or Tweet). Then the controller taps into the models to access the model’s database. After modifying an entry in the database, the controller returns the proper data/view back to the client browser to render onto the screen.

**Mongo:**

I used Mongo to store accounts and tweets. Each account has a username, displayname, password, createdDate, following array, followers array and salt properties. On the other hand, each tweet consists of the tweet message, the owner (account), the owner’s displayname, an image data property, favorites array, createdDate, and array of comments (or replies) to that tweet. The TweetSchema has a findAll(), findById(), and findByIdForAll() functions. The findAll() function is used to retrieve all existing tweets in the mongo database for any logged in user to view. The findById() function is used specifically to find a tweet by id that belongs to said user. This is mostly used when the user wants to edit/delete one of their own tweets. The findByIdForAll() function is used mostly for replying or favoriting a tweet. Any user can do either on a tweet but the server still needs to know which tweet to look for in the database, thus, the tweet’s id is necessary.

**Above and beyond:**

For above and beyond features, again, I wanted to focus on implementing features that are realistic and exists in real social media platforms. Since I continued from my project 2, I focused on adding more features and continue polishing the overall app. In terms of polish, I fixed up the ‘favoriting a tweet’ feature. Before, users can favorite a tweet multiple times by refreshing the page. However, instead of storing the favorites as an integer count in mongodb, I made it an array. As a result, when a user favorites a tweet, their account will actually be saved into the array so that the next time they try to favorite the same tweet, the server will check if the incoming user has favorited the tweet before. If so, do nothing. Another polish feature I worked on was the infamous image data upload. As mentioned in my previous documentation and due to my stubbornness, I wanted users to be able to upload image files, rather than image links. I am using multer to help with file uploading from the browser client to my server, which the file is then stored in an imgData property in mongodb. Furthermore, I added the dark/night mode feature. Users can toggle the overall page view of the app with a simple click of a button! Finally, I added the feature in which account users can have followers and follow other accounts. Once a user successfully makes an account, they can search for other existing accounts and follow them, and likewise, vice-versa (And no, users cannot follow themselves, who does that?! [I am checking that]). With the following and followers system, users can only view tweets posted by themselves and the accounts they are following. Searching for accounts to follow and general profile stats (number of followers/following) is displayed on the left-hand side of the main app, as that area was empty in my project 2.

\*Note: Images may be removed due to Heroku dropping the server/app every 30 minutes.