Software Implementation and Testing Document

For

Group 3

Version 1.0

Authors:

Tommy Chong

David Markel

Victor Marques

Dylan Gire

Abraham Beltran

1. Programming Languages (5 points)

Python: Used to host server/client. We used Python mainly because the Django framework helps establish the DataTweetApi for usage. Which allows us to establish a connection between the backend and the frontend in a simple manner. We use it for the backend to connect to our database and receive incoming requests from the frontend.

HTML: Used for the website page structure.

Javascript: Used to help coordinate website behavior. This is the language used in the frontend which renders HTML segments with React. This builds the UI of our application and is able to connect to the backend to post or retrieve information to/from the database. Mainly picked JavaScript because of its use of the React library which is really powerful for the frontend.

JSX: Specific to react. Though it is similar to HTML there are specific keywords only used in react.

SQL: Used to store data on Users and posts. Used in the backend to make changes to the database. We use this because our application uses the SQLite database which requires SQL.

2. Platforms, APIs, Databases, and other technologies used (5 points)

Backend:

-DataTweet: Django

-DataTweetApi: Django (),rest_framework

-Encryption: Provided by Crypto

-Authentication: Provided by auth0, used for logging in and verifying users

Frontend:

- ReactJS
- Axios component to connect to our backend
- Bootstrap for quick CSS help

3. Execution-based Functional Testing (10 points)

Whenever we pull the code from Github, first we do some manual testing. When we run the code in the backend and frontend, we check if there are any errors in the terminal in the backend, and check if there are any errors in the console in the web browser for the frontend. We do this with all the functional requirements before we write new code. Since everyone tests it, we know that the code is not broken or the feature is not broken since we can all agree on it. In the future we will create unit tests to automate this, in the backend we will use the unittest module to test the functions and in the frontend, we will be using Jest or React Testing library to test the components in the frontend.

4. Execution-based Non-Functional Testing (10 points)

Will consider adding karma tests to each component, Karma test will be run on the build. We will see how much time we have, after all, testing is an important part of software engineering. We will be using the Perf library in React to test for performance.

5. Non-Execution-based Testing (10 points)

Before we merge the code of anyone, we all collectively look at the changes that were requested in the pull request. Since we are 5 members and each one of us looks at it and we all agree there’s nothing wrong. Then, the code should be fine and we can accept the pull request and merge it to the main branch. This allows all team members to stay up to date with the new updates in the codebase and do code reviews to check for any potential errors. We communicate our thoughts on the pull request through our group in Discord. In summary, we take advantage of source control using branches so that we can go back if we need to revert a change.