Software Implementation and Testing Document

For

Group 14

Version 1.0

Authors:
Davion M
Douglass S
Ryan R
Conner F
James H

1. Programming Languages

List the programming languages use in your project, where you use them (what components of your project) and your reason for choosing them (whatever that may be).

a. Javascript (the entirety of the project)
b. CSS3 (Weather/Forecast page)
c. HTML5

2. Platforms, APIs, Databases, and other technologies used

List all the platforms, APIs, Databases, and any other technologies you use in your project and where you use them (in what components of your project).

a. React Javascript library (entirety of the web app)
b. SQL/MongoDB (login/hurricane data)
c. OpenWeatherAPI/AccuWeather API (Forecast/Weather page)
d. NOAA API (Weather page)
e. Leaflet.js
f. Express.js
g. Node.js MySql (login/hurricane data)

3. Execution-based Functional Testing

Functional testing for GustBuddy was conducted by setting up each feature we outlined in our functional requirements. Each feature was preliminarily developed with Javascript, and actual functionality gradually implemented. The team began with developing the main page of the web app, which currently displays hurricane information including a monetary damage graph, as well as links to information of past hurricanes. Each team member took responsibility for multiple features to develop them in the same functionality and uniform style of the others. The team began with developing the main page and then divided up the other tabs (pages) to be developed.

1. Day/Week/2 Week weather forecast (HIGH)
2. On-this day past weather report (HIGH)
3. Hurricane Probability (HIGH)
4. Chat-based Weather Assistant (HIGH)
5. Florida Focus (MEDIUM)
6. Social media sharing implementation (LOW)
7. User log in (LOW)
8. Planet Forecasts (LOW)

4. Execution-based Non-Functional Testing

One instance of non-functional testing is comparing API output on our site to the website that hosts the API. This allowed us to see if our output represented the actual data. Additionally, the team decided against using various bootstrapping technologies as the packages would have been too slow and hindered performance. In the future our non-functional testing will include securing user's usernames and keeping their passwords anonymous, writing efficient code, and attention to minimize performance lag.

5. Non-Execution-based Testing

Davion, Conner, and Ryan worked on researching and implementing state and components with the react-router-dom to the functioning Dashboard. From there, the code was reviewed by each team member and inspected before the changes were merged.

James worked on researching and implementing how SQL databases could be used to store past and current weather data to be displayed on the web application.

DJ worked on pulling data from various weather API's to be used in the data analysis for both the Forecast and Hurricane report components.