**Software Implementation and Testing Document**

**For**

**Group 14**

Version 1.0

**Authors**:

Davion M

Douglass S

Ryan R

Conner F

James H

# Programming Languages

* 1. Javascript (the entirety of the project)
  2. CSS3 (Weather/Forecast page)
  3. HTML5

# Platforms, APIs, Databases, and other technologies used

* 1. React Javascript library (entirety of the web app)
  2. OpenWeatherAPI/AccuWeather API (Forecast/Weather page)
  3. NOAA API (Weather page)
  4. Firebase (login/hurricane data)
  5. Material UI (stylization)
  6. Recharts (statistic visualization)

# 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 current weather report (HIGH)
3. Statistics Graph (HIGH)
4. Firebase functionality (HIGH)
5. User log in /out (HIGH)
6. User reporting (HIGH)
7. Florida Focus (MEDIUM)
8. Social media sharing implementation (MEDIUM)

# 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.

# 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.

Conner implemented the Login functionality of the website.

James and Ryan worked on researching and implementing how Firebase could be used to store past and current weather data, as well as login information, to be displayed on the web application.

DJ and Davion worked on pulling data from various weather API’s to be used in the data analysis for both the Weather files and MainPage file. They also both implemented Weather.js, WeatherForm.js, WeatherComp.js, and MainPage.js