**Software Implementation and Testing Document**

**For**

**Group 20**

Version 1.0

**Authors**:

Betty Tannuzzo

Genevieve Larkins

Madison Vandersluis

Melanie Maguire

Nicole Garcia

# **1.** **Programming Languages**

* HTML Using HTML to create our webpage
  + Format text such as titles and headings
  + Arrange graphics on our webpage (this will be seen when we input our APIs and researched topics)
  + Link different pages within our website
  + Applied cards and containers for aesthetics
* CSS
  + Presentation of the web page(s)
  + Colors, fonts, layouts (coming soon once we get the website template up and working)
  + Adapt to computer screens
* Typescript
  + Functionality of the frontend of the webpage and connection to backend
  + Used for filtering, variables, any sort of action events, and connecting to firebase
* NodeJS
  + Used to make a webserver to host the app locally
  + Helps manage dependencies and gives an angular cli for scaffolding by using npm in node.js

*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).*

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

* Angular
  + We are using Angular to build an interactive, user friendly webpage. This includes the layout of the page, routing, and overall how the page behaves.
  + Uses bootstrap for overall styling
* Google Maps API
  + We are using the Google Maps API to display a searched area with markers to show shelters in that specific location (implementation in progress, currently embedded)
  + Using My Maps to create the emergency shelter markers and overall map
* NOAA API
  + We are using this to get weather updates for our website
* Bootstrap library
  + templates for our website
  + Using this library to auto-generate specific templates for buttons, webpages, and spacing, color schemes, tabs, and styling formats to help generate our website
* Firebase
  + Using this to hold users to have user accounts on the page
  + Can either push a new user into the database or get info of an already existing user
  + Instead of authenticating users like we originally wanted, we also could check the data of a user and compare it to forms in login to make sure the user already existed and the login info was correct
  + It holds missing/found person entries and is where we get the info of an existing entry for people to look through
  + Also holds the forum entries and gets/updates/or deletes the information about the posts

*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).*

# **3.** **Execution-based Functional Testing**

For the home page we tested that all the buttons worked by clicking on them and making sure the linking redirected the site to the corresponding site. The NOAA implementation was tested by searching for different cities and making sure the API was working properly. The find shelter was tested like NOAA to ensure all the features worked on the site and that the images and markers were displaying the right things. For the MIA component we have tested that the new entry link works by clicking on it and adding new entries on the page that it navigates to. We see that it works by checking the Firebase database to ensure that the new entry shows up properly. During this increment we also added the ability for users to include a picture with their entry and we see that that works by checking the database as well as seeing it show up on the MIA page that shows all the missing/found person entries. We also did this for the forum component, testing when a “user” creates and submits a post, it shows up in the database and on the forum page. For the Google Maps, currently you can select the markers and the name and address will pop up on the top left corner. With the NOAA API, the feature is fully working on the weather alerts tab component. For the login and signup pages, we see that all the forms work and that the buttons link to a success page if a user signup/login will be successful. For the help-forum page utilized console.log() messages to print out data to the console to see what events were being called. To check if components were working checked if the correct visuals showed up on screen. Also compared data being entered for creating a post, and what actually showed up in the feed and in the firebase database. For the login/signup pages console.log() was used for almost all testing to check if the data was correct in the object and/or looking at Firebase to make sure the user is created in the database.

This is all checked by creating a local server on “localhost:4200” in either Safari or Chrome using “ng serve” in the angular cli, which showcases all behaviors of the webpage on a server. This allows us to check the functionality of the webapp since it displays and acts how the page would be seen/used by users.

# **4.** **Execution-based Non-Functional Testing**

We have tested the execution based non-functional testing in terms of if and how fast the page takes to go between routing and if the layout makes sense (user friendly). The NOAA API actually loads within an appropriate amount of time. However, the Google maps API sometimes takes a little longer to load; especially after adding 600+ emergency shelter markers to it.

# **5.** **Non-Execution-based Testing**

* In addition to asking each other for advice on certain problems we were having with our individual parts, we sat down and went through each other’s code in order to see if there was anything that could be improved and fixed small issues that we came across.
* We also peer reviewed the documents for this iteration and made sure that everything seemed to make sense