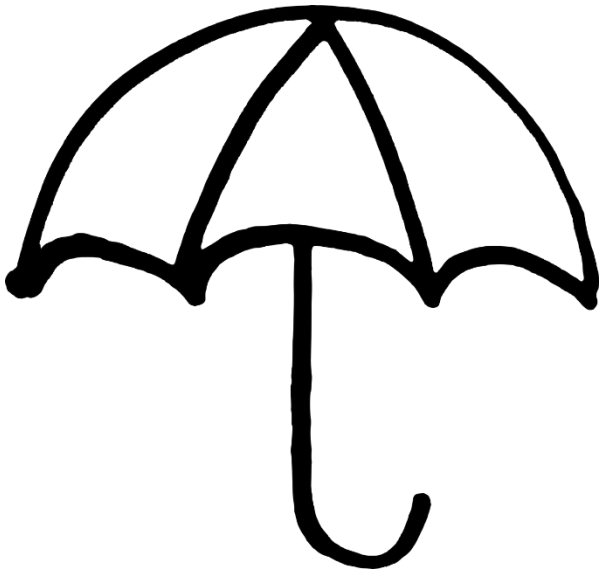
Group 8

BrainStorm Team



Software Requirements Specification

Noah Brayer, Dennis Donoghue, Chad Klinefelter, Kevin Moriarty

1. Abstract

1.1 Summary

This is the requirements document for the Meme Doppler Project. The Meme Doppler is the name for the system that will take in location data from the User Interface, query the NWS for weather data, query our internet meme database, and deliver memes and other data to the User Interface, an Android App.

This has remained unchanged for the final version of our app.

1.2 Scope

The Meme Doppler platform aims to combine the functionality of a weather app with the entertainment of meme culture given the pervasiveness of the meme through the millennial generation.

The primary scope is to provide local weather from reliable sources filtered through the DANK algorithm to produce a memed output ~~as well as allowing users to affect the DANK algorithm to identify the most appropriate meme for a given set of temperature/weather conditions.~~ The software will also alert the user of any emergency weather alerts in their area.

This summary of the scope's only change is that we are not allowing users to affect our algorithm, as feedback was cut from the final design.

In-Scope:

(a) If the user inputs a location, query that location's weather data.
(b) If there is no location input, use geolocation data, if the user consents.
(c) Else, use IP data to find location. This is the most reliable and least accurate option.

(a)-(c) have been implemented through the Google Places API, user consent is not in-scope as permissions will be enabled outside of our app.

(d) Access our database to provide a meme based on the weather data from the best available location data (Allowing for A-B testing and variation).

Implemented with a ZIP code being the final form of location data.

~~(e) Update meme database and delivered meme based on user input.~~
~~(f) Allow users to create and upload their own memes, which will be added to the meme database after team moderation.~~

Eliminated (Part of feedback)

Out-Of-Scope:

(g) Storing data via a user database. I.e. we do not provide functionality for customized data such as favorite locations upon return to the site.
(h) Automatic processing and uploading of user-submitted memes.
(i) Providing a real-time weather forecast radar map (the moniker "Doppler" is misleading in this sense).

This has not changed since our initial document.

2. User Requirements

2.1 Software Interfaces

(a) **NDFD - SOAP**
The National Weather Service maintains the *National Digital Forecast Database*, a service that provides the public, government agencies, and commercial enterprises with data from the National Weather Service's (NWS) digital forecast database. The data we will be using is in the XML format, located on the NDFD's SOAP (*Simple Object Access Protocol*) Server.

Achieved; data is queried using ZIP code which returns lat/long coordinates which are used to requery for the actual weather data for the given area.

(b) **Android- Location Callback**
~~Android implements several ways of acquiring user location data. We will be using the callback and requests to the LocationManager through wifi, GPS, and Cell data. For Cell and GPS data, we will need to request user permission.~~

All functions described under section (b) were implemented automatically using Google Places API

(c) ~~**Google - Cloud SQL**~~
~~Google Cloud SQL is a MySQL-Compatible database service. We would use it to store our meme database and we will query it in order to deliver memes to our UI. Because of the basic discounts with Cloud SQL as well as the $50 dollars we get for Cloud products, we chose this over other services such as Amazon's Aurora.~~

(d) ~~**Android & Google - Cloud SQL Android API**~~
~~We will be using the Google Cloud SQL Android API to communicate between our database and application through the Cloud EndPoints Functionality.~~

Final communication between the database and the application was achieved via image urls being stored in the database which are called and processed via the Picasso image handling library for Android.

2.2 User Interfaces

(a) **Main weather pane**
Upon the user loading the application, the system will load the selected meme for that hour in a pane with the memes for the following hours in separate, smaller panes below.

Implemented as described, the final number of secondary panes was set at six.

(b) ~~**User feedback**~~
~~Below this will be two buttons, "Dank" and "Not Dank". Users will select a button based upon their opinion of the meme and/or its appropriateness to the weather. If they select "Not Dank", then they will be presented with a myriad of options to report what the meme was lacking. There will be "Wrong meme" if the meme is not appropriate for the weather,~~

~~"Wrong text" if the text does not match the established definition of the meme, "Outdated meme" if the meme has fallen out of vogue, "Inaccurate weather report" if the weather appears incorrect, and "Other", which will present the user with a box to type their comment/complaint. After selecting a button, users will be returned to the original page.~~

User feedback has been eliminated in the final design.

(c) **Location**
The page will include an area to change the location for which the user would like to view the weather. To reiterate, the default location is ~~the user's current location~~ **Amherst, MA**. To view a different location, the user must specify so.

2.3 User Characteristics

(a) Minimal technical expertise will be required to operate the application, and a minimal education level will be required to engage with the application.
(b) The users we are targeting are millenials. As the internet meme was invented and became popular during their lifetime, millennials and younger people have the most experience with them and are therefore the ideal audience.
(c) Previous interaction and experience with Internet memes is helpful in allowing the user to be fully engaged, as many of the memes are dependent upon the established associations of their background image.

These descriptions are indicative of the final design and remain unchanged.

2.4 Assumptions and Dependencies

(a) Functionality of the Meme Doppler depends on a reliable connection to the Internet.
(b) We are assuming the weather data we collect is accurate.
(c) We are depending on the sources to maintain continuously updated databases. This includes the developer's maintenance of the meme database.

These descriptions are indicative of the final design and remain unchanged with the possible exception of developer maintenance.

3. System Requirements

3.1 Functional Requirements

(a) **Current Location Services**
When a user logs into the application, they should expect to see weather for their current location.

The default location is locked to Amherst, MA and currently does not reflect the user's current location until updated.

(b) **Location Input**
A user shall be able to input a zip code or other location information and receive a meme based on the weather at said location.

This has been fulfilled by the design but not yet integrated.

(c) **Location Linking**
A user should be able to copy the url to receive data from the same location. Phrased differently, this means the url should be based on location and location alone. This allows for bookmarking and linking functionality.

(d) ~~**Keep Database Up-To-Date**~~
~~A meme that is found to be out of vogue to often should be removed from database, or limited in such a way that it is displayed less often.~~

(e) ~~**Vary Output**~~
~~Given no change in weather or location, the meme delivered to a user still must vary.~~

~~(f)~~ ~~**User Feedback**~~
~~If the user decides to give feedback on the meme they are currently presented with, their opinion will affect the meme's weather score. A positive response will strengthen the chance of said meme appearing in similar weather. Negative user feedback will be sent to developer for further analysis.~~

3.2 Non-Functional Requirements

3.2.1 Software Quality Attributes

Reliability is important, as users are expecting the application to be functional at all times. If it is not, we will see a rapid decline in usage.

Security is not very important, as there will be no concept of logins or users. Each access will be its own individual access. One possible security concern is the meme database. Only developers and community managers should have access to the meme database.

Our Software Quality Attributes have not changed since the first iteration.

3.2.2 Performance Requirements

(a) **Initial Load Time**
When a user opens the application, the application must load faster than 2 seconds. It should rarely load slower than 1.5 seconds. The goal load time is 200ms. These times should carry over to eventual web development, where they will affect SEO.

(b) **Re-Load Time**
A reload will be any loading and delivering of new memes after the initial application launch. This should be equal to or faster than Initial load times. The app should automatically reload after our minimum forecasting time period passes.

(c) **Minimum User Specifications**
In order to have a good experience with our application, the user must have an android device with Android 4.4 (KitKat) with a Intel® Atom™ Processor Z2520 1.2 GHz, or faster processor, 1.2GB or more storage, and 512MB of RAM.

(d) **Minimum Server Specifications**
~~Google Cloud SQL server dynamically adjusts available storage and throughput based on usage.~~

3.2.3 Design Constraints

(a) **Security**
Our meme database must be secured against malicious entries and deletions. ~~In addition, crashes during collection of user feedback should not corrupt data.~~

As user feedback has been eliminated this section has been relabeled security. Database security is maintained via standard password protection for our server and cPanel.

4. Future Expansions

The scope of our project is limited mainly by time. Our end result will be a fully-functional android application. In addition to this, expansions that we would like to implement if we find the means beyond project completion would include the following.

(a) **Web and iOS Development**
The deliverable for this project is in the form of a android application, which will indeed be able to run on mobile devices. At best, however, an android app will be wildly inconvenient for desktop use, and at worst inaccessible. To gain a larger and more frequent user base, iOS and Web development would be essential.

(b) **Real-Time Radar Map Forecasting**
The most advanced and informative method of weather delivery today is Radar Map Forecasting. In this, a map of the world is overlaid with real time and forecasted weather data. We would like to deliver something similar, where instead of various shades of red and blue, the map would be overlayed in memes.

(c) **Multiple Sessions**
In the future, we would like to be able to deliver, either through tabs or some other method, a way for the user to have multiple sets of individual delivered memes at any given time.

We are still considering these expansions in the future, however at the current time our focus is still on getting the Android application to 100%.