Cross-Platform Development Assessment Item 1 Report

Concept

Overview

Have you ever been in a situation where you looked at the current forecast for your current location or the location you plan to travel to? Only for the weather forecast to be completely wrong, resulting in you getting soaked without your coat to save you! Or perhaps it claims that it is raining, only for it to be sunny, resulting in you having to lug your coat around?

Weatherly is a community based weather app that harnesses the power of users to report back on the actual conditions of weather based on their current locations. This provides real time feedback on the actual conditions of a location rather than just relying on predictions provided by the forecasting services. With Weatherly, gone are the days of guessing on whether the forecasted weather is true or not. Weatherly also provides a forecast just like regular weather apps that provides fast, timely updates to make sure you have all the information you need from multiple sources.

With Weatherly you'll never be caught unprepared again!

Barcode:

Requirements

The weatherly app can be used in a variety of user scenarios, from travelling to a location such as going on holiday, or embarking on a typical daily commute to work. The target group for the app is users of any age group that travel often, stay over in remote locations for a few days, and/or users who prefer to travel with just the most essential items.

As a frequent commuter Omar wants to have a more accurate weather report so he can make an informed decision about what to take with him to university. As a frequent commuter Omar also wants the ability to save his favourite locations so he can keep track of their weather conditions.

Biff the businessman frequently travels to work and takes public transportation to get there, he prefers to travel lighter as it is more comfortable for his journey. As he only takes a briefcase to work it doesn't provide any space for an umbrella or coat for his travels. He often finds that after watching the morning weather forecast on TV before work, he takes his umbrella to work with him only to find that the weather is different than the forecast and as a result he has carried his umbrella around with him for nothing. Biff then searches for a better solution to this problem and stumbles upon Weatherly on the app store. Weatherly helps Biff to make a more informed decision about whether he will need to take his umbrella to work all day. Biff uses weatherly one day, which reports that his work location will be cloudy but dry, Biff then realises that the community report says otherwise, with over 2000 users reporting that there is rain present. With this new knowledge he can now more

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

easily justify taking the extra luggage with him to work as the community has reported rain is indeed present at his work location, he then is able to keep his suit dry and his hair tidy for his big conference meeting.

As such, the functional requirements of this app would be the use of a custom API hosted on a server which the app harnesses to provide the community features for users reporting weather conditions at their location. The app also needs to tap into a weather API to pull information about the location the user is currently situated in, and to provide weather data for remote/added locations. It then needs to store those locations locally, so the app can "remember" the locations added for the user, so they don't need to constantly input them. This data then needs to be displayed onto the app's UI for users to view.

Non-functional requirements would pertain to making the community ratings quick and easy for users to participate in. A community based system that allows users to anonymously post updates on the current weather conditions of their current locations and to receive community updates on other locations.

Competitors

Two apps that Weatherly may compete with are BBC Weather (BBC, 2013) and Met Office Weather Forecast (Met Office, 2013).

The BBC Weather app's main window consists of a full screen background that reacts based on the current weather conditions of a given location. The app provides weather information in detail, ranging from reports of air quality, humidity, and visibility conditions. Because of all the detailed information, the main app's window is densely populated without any logical placement of these elements, this leaves the window looking cluttered and potentially confusing on first use (See Figure 1). One aspect of the main window's UI is the increased emphasis the app puts on critical information, such as the current temperature, weather condition, and location are all larger in size and bold. This assists the critical information in standing out from the rest of the less relevant data in main window, which immediately draws more of the user's attention (See Figure 1).

Figure 1: BBC Weather app's main window of a location's weather condition (BBC, 2013).

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

The app's functionality allows users to search for any location around the world and add it to a personal saved list, which saves the user time and adds to the convenience of the app. One feature that was not apparent was that the UI allows for swiping to the left to reveal an hourly forecast, there is no indication to show to users that this feature exists and is possible.

This app could have benefited from a onetime tutorial that introduces the app's features to its users to get them more accustomed with the UI and functionality of the app. This is also reflected through some of the symbols the app employs that don't entirely indicate what the data may be showing, for example the symbol that is employed for wind direction and speed is not entirely clear, nor does it tell the user what metric is being used. This is a negative element that Weatherly will try to avoid, by making all weather data easily readable and laid out in a logical fashion.

The 2nd app is MetOffice Weather Forecast (MetOffice, 2013), its main window is more minimal in design compared to the BBC Weather app (BBC, 2013) and shows a list of locations first, rather than current location conditions (See figure 2). From this interface the app prompts the user to use the search bar to find locations they may be interested in, which provides the user with direction, an improvement over the BBC Weather app (See figure 2).

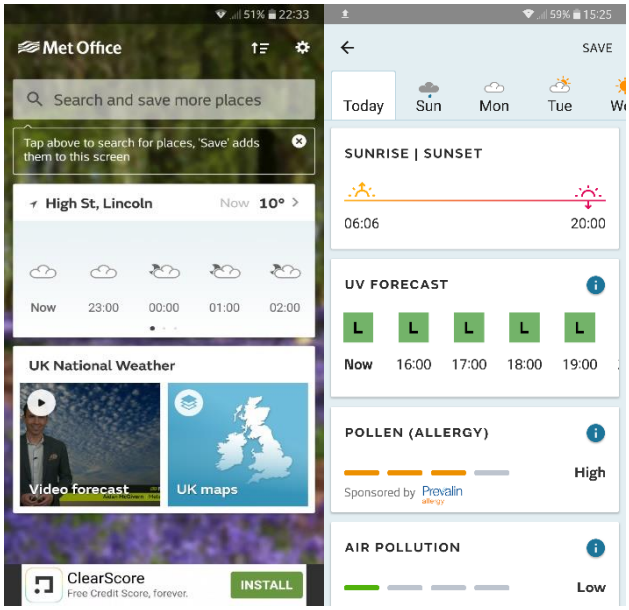

Figure 2: Main window of the MetOffice Weather Forecast app being displayed (left) and the current weather conditions of a location (right) (MetOffice, 2013*).*

The user experience of the app is an improvement over BBC Weather, with better spacing and clearer separation between contents which better breaks up the information and makes the window look cleaner (See figure 2). Weatherly will employ a similar approach of keeping the UI clean and information displayed in a logical fashion.

MetOffice Weather Forecast provides substantial detail, with the app harnessing the power of MetOffice's API to provide rich detail about the current and future conditions of a location. Weatherly will also make use of a forecast API to gather information about a location's current and future weather conditions, combined with the community features this will help users paint a more accurate image of what weather is like at a given location. With both apps, Weatherly distinguishes from them with its unique community ratings feature that hopes to provide a more accurate weather forecast for locations by leveraging the power of the community.

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

Prototyping

Prototyping work was approached through the creation of non-interactive mock-ups using Affinity Designer (Affinity, 2014). These mock-ups show the different windows the user can navigate to when using the app and shows the layout of the user interface. These mock-ups were then presented to fellow colleagues along with a questionnaire to evaluate their effectiveness in areas of design and usability. The testers were also encouraged to suggest improvements and to say what they liked and disliked. This feedback would then be used to iteratively improve the designs of the prototype, hopefully leading towards a sufficient design that would then be used for the final app design and layout.

The first stage of prototyping involved an early concept that had most of the functional requirements situated in a large card that took up most of the display. The user would then swipe the card to flip it over, displaying more detailed information about the weather such as a weekly forecast or wind speeds (See figure 3). Multiple cards can be added to the main window, forming a list of locations the user is interested in seeing.

The current location's card would include a community feature section at the end of the card's content. This allows the user to see the community report on the weather in this given location and provides the user with the option to add to the community report by casting their vote on the current weather conditions of their current location.

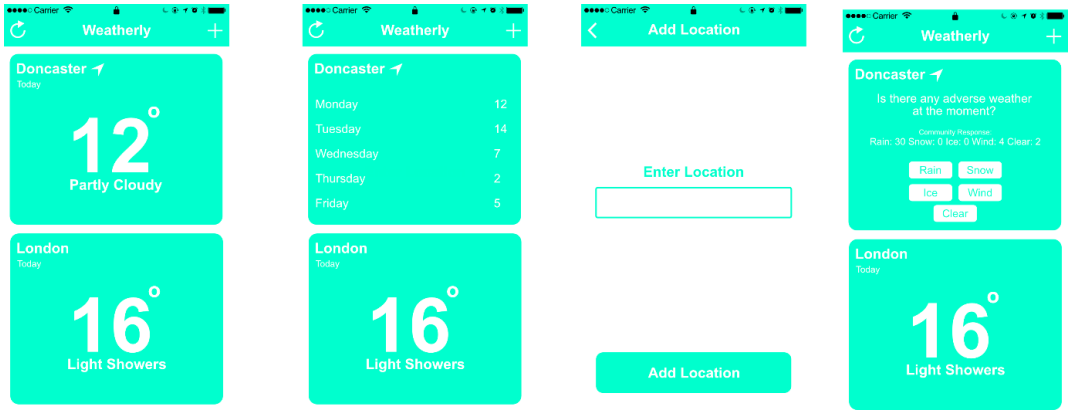

Figure 3: Prototype 1's various windows (Colours were not final).

This design was not well received by user testing participants, as many noted that it was impractical to keep having to constantly flip the card to look between different pieces of information (See figure 3).

The testers did not find it very easy to use because of this feature, this prompted a rethink on how the user interface was to be laid out. Testers did rate consistency very favourably but noted the community feature was not easily understood and could benefit from extra clarity on how it works (See figure 4). The enter location page was specifically acceptable to testers, providing verbal feedback that the placement of the elements on that page were great for one handed usage (with exception to the back button).

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

An iteration on this idea was created for users to test again but they again found that flipping the card to see more data and interact with the community ratings feature was too inconvenient and more awkward than it should be (See figure 5).

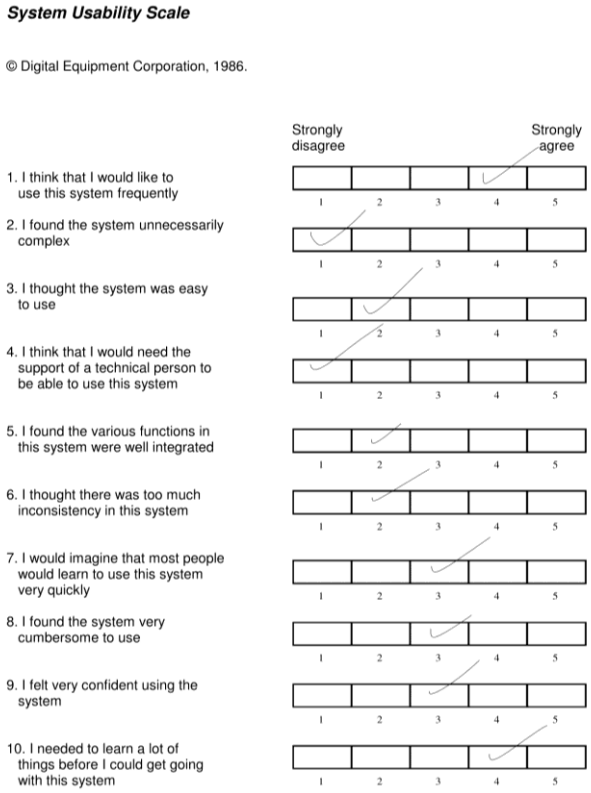

Figure 4: User response to the first prototype via a usability questionnaire.

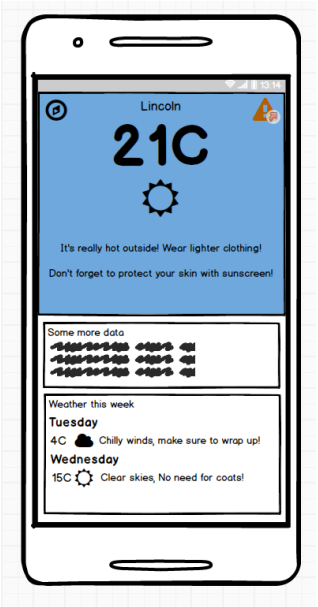

Figure 5: An interactive iteration on the initial prototype that attempts to show more information at once to the user.

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

The next prototype had a drastically different layout from the previous, learning from the issues highlighted by the user testers' previous feedback. The cards layout was replaced with a much simpler list of locations, which users then tap on to take them to that location's information page. While containing less weather data, the new layout has an improved design with all information being comfortably displayed in one window (See figure 6).

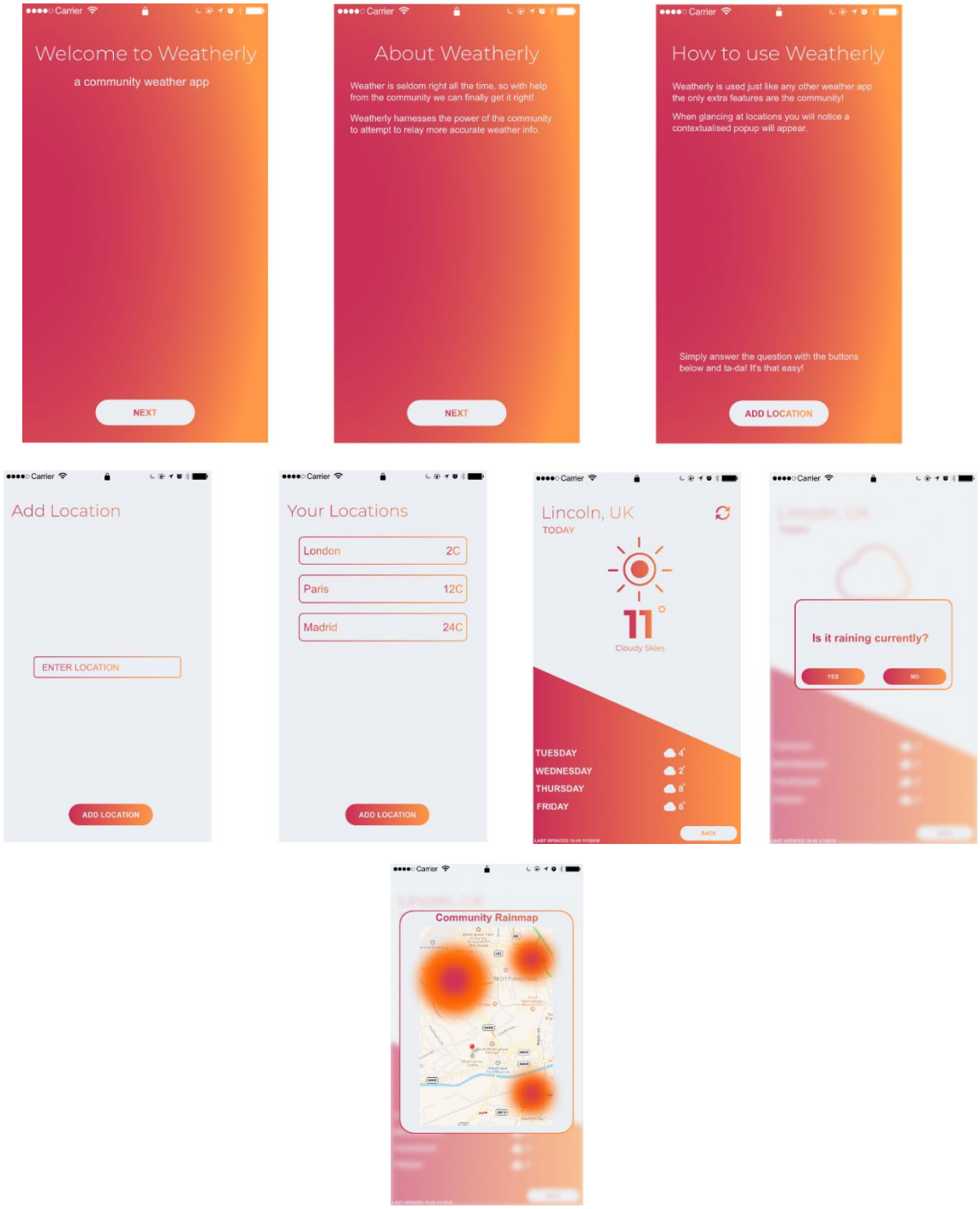

Figure 6: The final prototype/mock-up that inspired the final app design.

Following on with suggestions pointed out by user testers evaluating the last prototype, a tutorial was introduced into this prototype which helps users understand the basics of the app and how the community ratings feature works. The tutorial was evaluated by fellow colleagues to determine its

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

effectiveness, which was met with positive responses. Testers noted that the tutorial better helped them understand the community ratings feature, how it works, how to use it, and how the user may potentially benefit from this feature. A wireframe was produced that mimicked the layout and design of the initial prototype (See Figure 7).

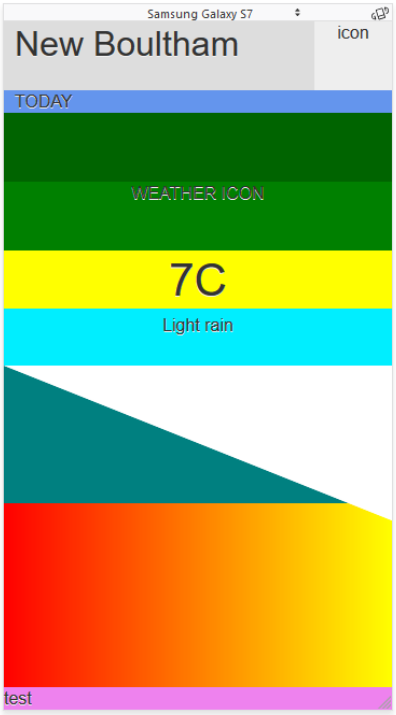

Figure 7: The initial wireframe layout with the weather API built in and successfully fetching data about the current weather conditions.

Another improvement over the original design is how the community ratings feature has been implemented. The feature now provides the user with a popup asking them what the current weather conditions are like outside compared to the forecasted prediction. This method, while arguably more intrusive, means that the likelihood of users contributing and using the community ratings feature will increase significantly. Another feature was also proposed in this prototype, a heatmap layered over google maps allows for users to see exactly where adverse weather has been reported.

The popup improves over the original design by making it easier and quicker for the user to answer, rather than making users swipe to the end of the card to find the feature.

As the app could be used in a variety of public/outdoor environments, extra care was given to one handed usage of the app, by making sure all critical elements are reachable even on larger display devices which was well received by testers as they noted usability was much better than the previous prototype. This design then matured into the final look of the app shown below (See figure 8).

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

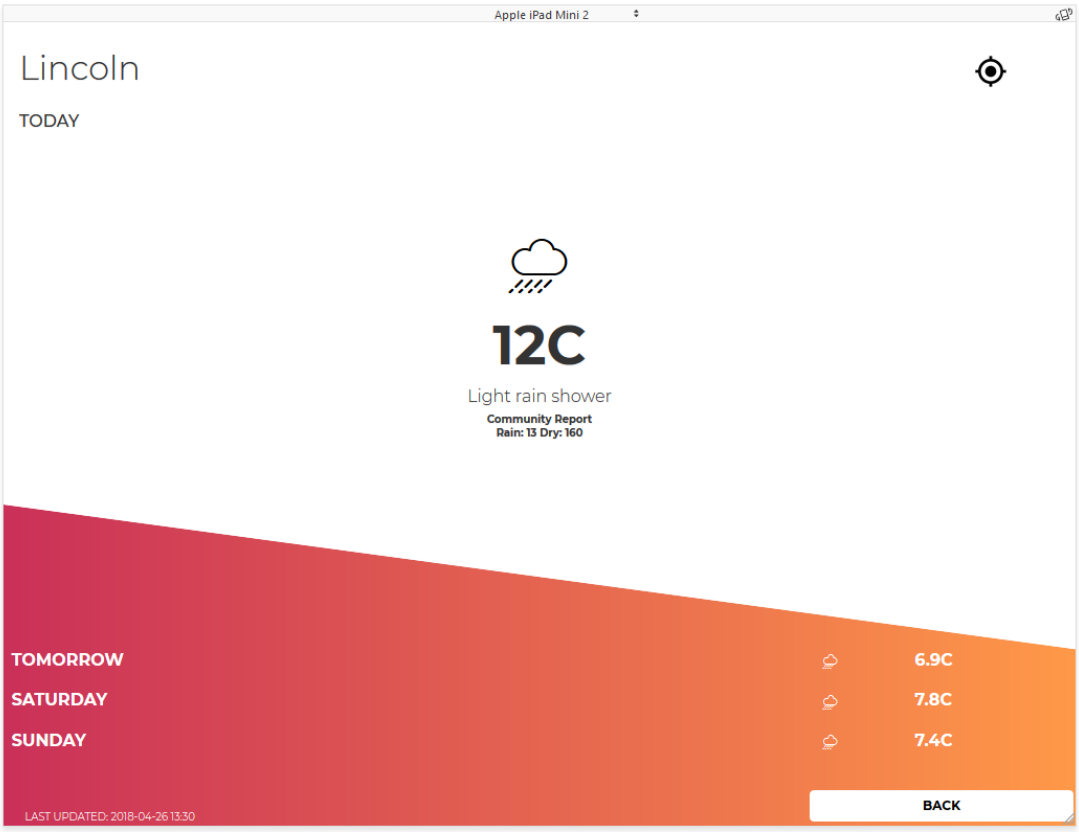

Figure 8: The final look of the app shown on an Apple iPad Mini 2 tablet in landscape mode.

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

Final App

The final version of the app adhered strongly to the final prototype mock-up except for the lack of the slant on iOS devices due to lack of support on Safari (See figure 9).

Figure 9: The final app working on both iOS (left iPhone SE (1136 x 640)) and Android (right LG V30 (1440 x 2880; 18:9)) devices.

The community ratings feature and the weather API are two highlight features of the Weatherly app.

The community ratings feature provided ample challenge for the development process of this app, the feature was implemented into the app using a custom API hosted on an external server which proved difficult to work with. The external server is hosted using a service called Heroku (Heroku Inc., 2007) which stores all the API's data for all devices using the app to access. You can see the raw data here:

https://secret-meadow-93624.herokuapp.com/ratings

Creating a custom API for the app to communicate with was a new experience, this feature was achieved through using a boilerplate via Yeoman to create the server backend files automatically and create the API service (Yeoman Inc., 2012). The MongoDB plugin was added to the server so the API data could be stored correctly for the app to send GET and PUT requests to (MongoDB Inc., 2009). The structure/schema of the API can be seen below (See figure 10).

Using a custom API allows for data to be freely shared and updated across devices running the app quickly and easily, this system helps to facilitate the community requirements of the app. The ratings data is then taken from the API based on the location in question via a GET request, this data is then displayed subtly underneath the conditions of the current weather showing a user if others reported the area was raining or dry.

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

The nature of the API allows it to be highly expandable and account for all other weather types in the future by replacing the yes and no fields in the schema to types of extreme weather such as high winds, snow, ice, hurricanes ect. Then replacing the popup's yes and no buttons with a drop down list of all possible extreme weather options for users to report and contribute to the ratings.

The UI would then be updated to show the most voted weather condition rather than show all responses at once as this would look cluttered, users would be able to view all responses by tapping on the community report div to bring up a popup. Lee *et al.,* (2014) notes the importance of using simplicity in an apps design which greatly improves its overall usability and design. This logic is applied within the Weatherly app, with simplicity and ease of use being key factors for the requirements of the app, as it is intended to be used to quickly acquire knowledge of weather conditions while on the move.

```
const communityratingsSchema = new Schema({
  location: {
    type: String
  },
  yes: {
    type: String
  },
  no: {
    type: String
  }
}, {
  timestamps: true,
  toJSON: {
    virtuals: true,
    transform: (obj, ret) => { delete ret._id }
  }
})
```

Figure 10: The structure/schema of the JSON data for my custom API with fields for location, yes and no as well as id and timestamps of when the data was published to the server.

The 2nd core feature of this app is the API weather service that is used to get forecasted weather from a traditional source. This API returns detailed data about the current and future weather conditions of the location requested, and displays that data to the UI for the user to see.

The weather data is laid out in the app with careful consideration to the layout, the current temperature draws the most attention with the large, bold font it employs. The icon attempts to denote the current conditions of the weather, but this may not be universally translatable so current conditions are also displayed in plain text to improve usability. As the community ratings feature is intended to be complementary data to the primary weather data, the font size for this UI element is smaller as a result.

The heatmap feature proposed in the prototype was not included in the final app due to this feature not adding anything meaningful to the overall experience and this was reflected through some initial testing. Hoehle and Venkatesh (2015) state that apps with poor usability tend to not prioritise the most "*essential aspects*" of the app and content is therefore "*ineffectively presented*" which negatively affects a user's experience.

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

Reflection

Cross-Platform technologies are greatly assisting app developers in making it easier to deploy their apps for multiple platforms, with the advantage of maintaining a single codebase for all deployed platforms. This improves development time when creating apps and allows developers to reach out to as many users as possible by making their app accessible to all platforms. This is reflected through my personal experience when developing the Mobile Computing assignment which took a similar amount of time to develop for but only works on the android platform.

This approach is not without its disadvantages, this method of development results in a final app that is likely to be slower overall in terms of performance when compared to native apps and cannot access some 1st party plugins.

Choosing a cross-platform method would completely depend on the proposed app's requirements and due dates. If an app requires a heavy reliance on hardware features, a native app will have much better support and stability over a web app as a native app is tailored for that specific platform. It can also access more specific hardware elements that web apps can't such as proximity sensors and ambient light detectors. However, apps that only leverage internet connectivity to provide content to users such as a weather app, it makes more sense to use a web app for its advantages of single codebase and faster development speeds.

Companies with lower budgets can also opt for using cross-platform technologies as they are generally cheaper to produce than their native equivalents. This is due to native requiring different developers as the programming languages differ and iOS specifically requires a mac machine to develop with, which all increase expenses.

Name: Daniel Garry Cumbor
Student ID: CUM 14547879

References

Affinity. (2014) *Affinity Designer* [vector graphics editor]. Available from https://affinity.serif.com/en-gb/ [accessed 25th April 2018].

BBC Media Applications Technologies Limited. (2013) *BBC Weather* [app]. Available from https://play.google.com/store/apps/details?id=bbc.mobile.weather&hl=en_GB [accessed 23rd April 2018].

Heroku, Inc. (2007) *Heroku* [cloud platform as a service]. Available from https://www.heroku.com/ [accessed 25th April 2018].

Hoehle, H. and Venkatesh, V. (2015) Mobile Application Usability: Conceptualization and Instrument Development. In: *MIS Quarterly*. 39(2). 435-472. Available from http://www.vvenkatesh.com/wp-content/uploads/dlm_uploads/2015/11/Hoehle-and-Venkatesh-2015-MISQ.pdf [accessed 25th April 2018].

Lee, D., Kim J.Y. and Yi, Y.M. (2014) Antecedents and consequences of mobile phone usability: Linking simplicity and interactivity to satisfaction, trust, and brand loyalty. In: *Information & Management*. 52(3). 295-304. Available from https://www.sciencedirect.com/science/article/abs/pii/S0378720614001463 [accessed 25th April 2018].

Met Office. (2013) *Met Office Weather Forecast* [app]. Available from https://play.google.com/store/apps/details?id=uk.gov.metoffice.weather.android&hl=en_GB [accessed 25th April 2018].

MongoDB Inc. (2009) *MongoDB* [database plugin]. Available from https://www.mongodb.com/ [accessed 25th April 2018].

Yeoman Inc., (2012) *Yeoman* [development tool]. Available from http://yeoman.io/ [accessed 25th April 2018].

Name: Daniel Garry Cumbor
Student ID: CUM 14547879