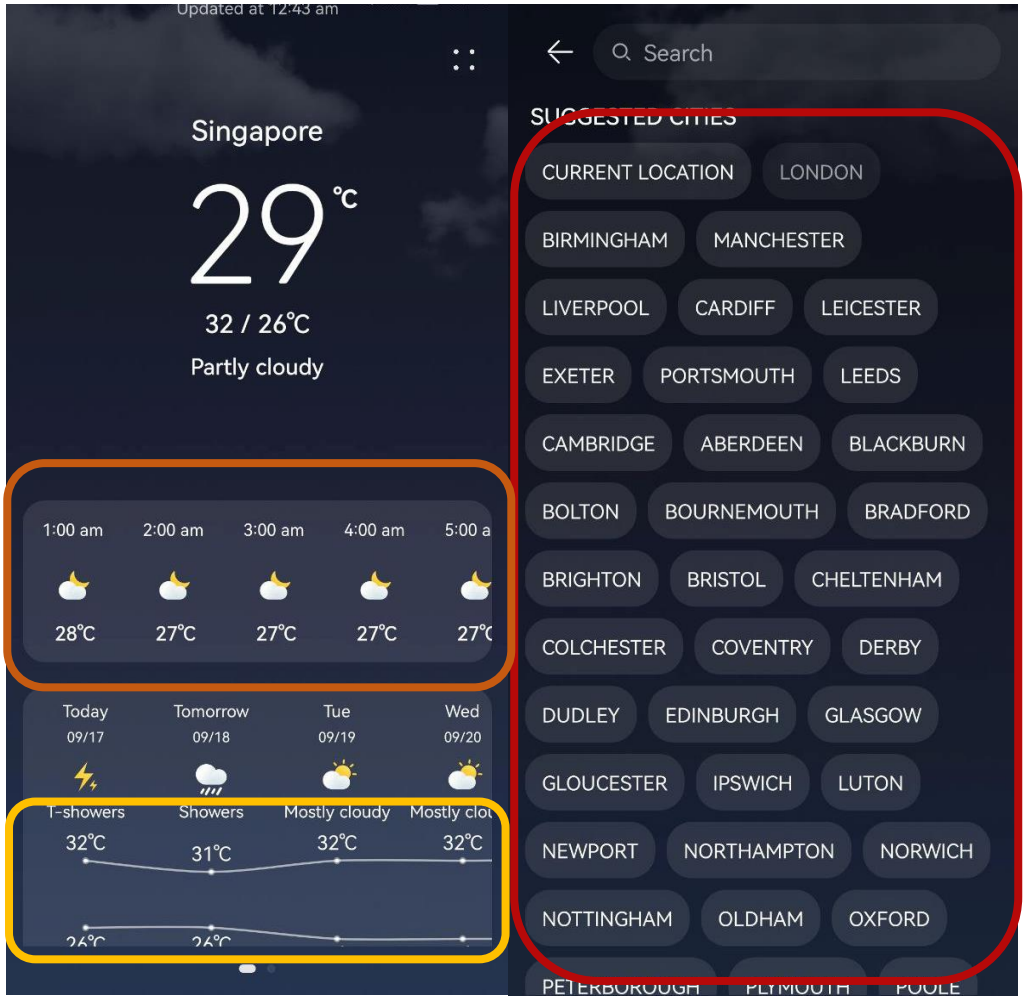
Mobile Development Project Report

Weather App

Contents

Concept Development

Development was done based on the very weather app I use as well as from my own experience of using weather apps and things I would like to see change.



Excessive data and graphs without the data I really want

No indication on the percentage of precipitation

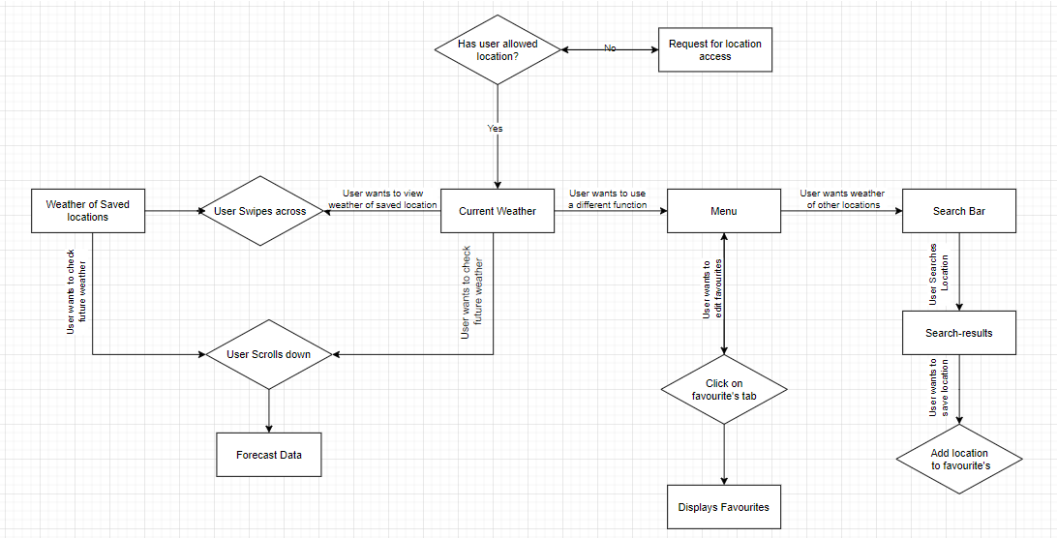
Locations on where I can get the weather from is limited to the provided cities in the country, unable to get the weather of more specific locations

Quite like this design

Search results same issues as home page

User Flow Diagram

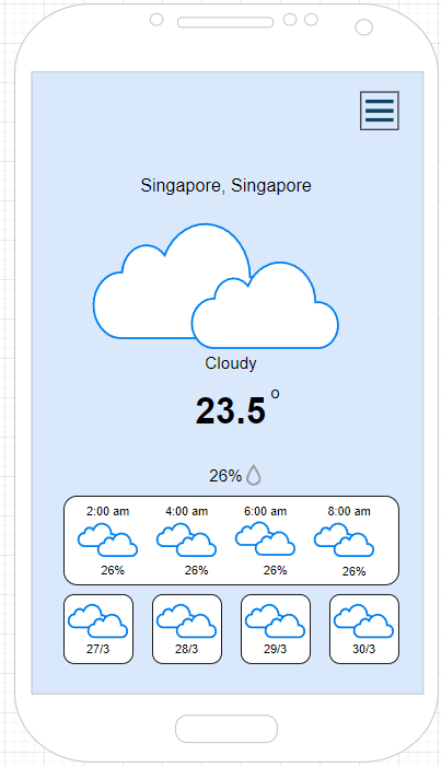
From my research I then created a user flow diagram based on how I would like the flow of actions to occur in the weather app



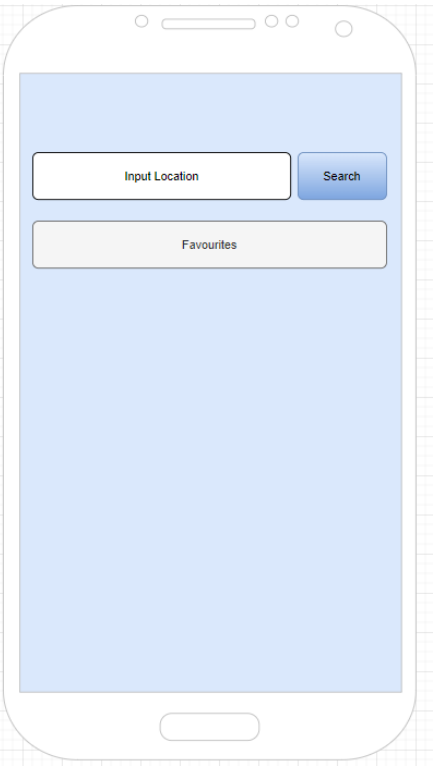
Wireframing

I then created wireframes based on the user flow diagram and research

Home Screen

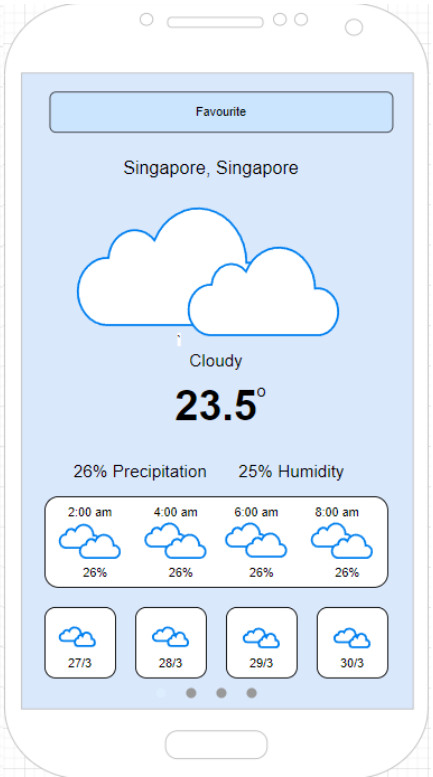


Menu Screen



Favourite's Screen

Search-result screen

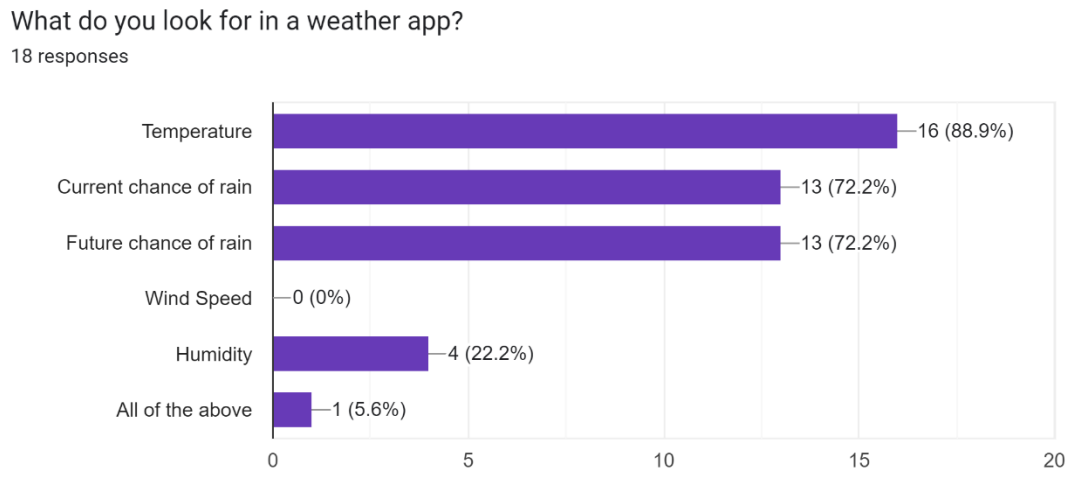


User feedback

I then conducted a survey regarding the wireframes to ensure that the design, features and data provided on the app would match up with consumer needs.

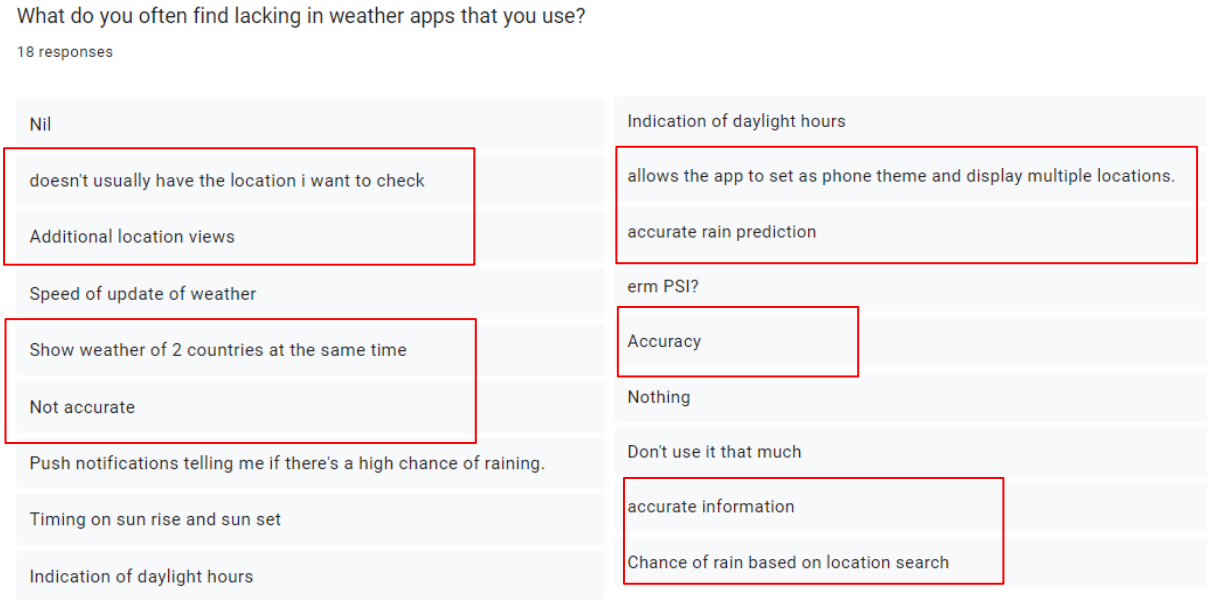
Data changes:

From the survey it can be seen that a majority of consumers use weather apps mainly for the temperature and weather forecast. However as there was a greater deal of user requiring humidity than I expected I made changes to my wireframe to include humidity`



Feature changes:

I also inquired on any additional functions that they often find lacking in weather apps they use to ensure that I don't miss out on important functions



Where about 50% of the answers were concerning on the lack of location accurate weather information which I also agreed with based on my own experience, where the locations offered to check the weather are limited.

Design changes:

Lastly, I enquired about the wireframe in general to open the application up to any criticism, design and detailed changes consumers would like to see.

Observe the image above do you find anything lacking? If so why?

18 responses

no prediction between the hours

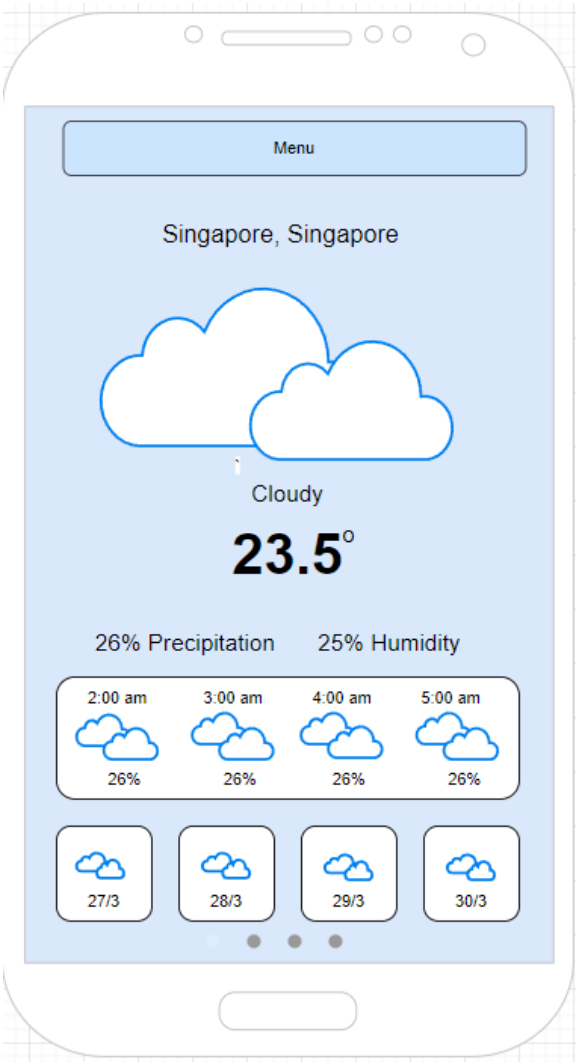
hourly updates. the image shows the updates by 2 hours

Unclear is it chance of rain or humidity level

could be more colorful

Colour

Of the feedback the main repeated ones were regarding the lack of colour and the time gap between the forecast. As the application will have a different background image I made changes regarding the time gap and further changes to make the design more consistent and proceeded to development

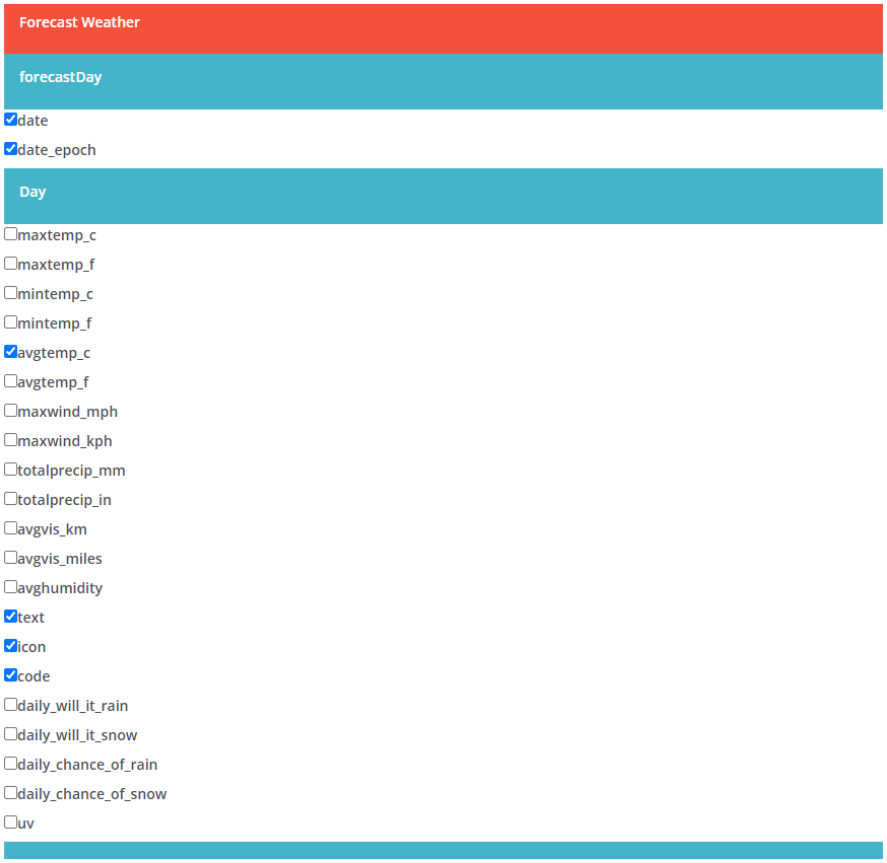


Development

I broke down development into its respective pages and the functions required by each page with the addition of information on the API used and the calls I used.

API

The API I chose was weatherapi.com as I found its ability to edit the response fields beneficial to ease data parsing.



In order to make a call to the API the statement required a query that could intake coordinates or location.

In my project I made two different calls one for coordinates and one for location in the home page and search-results page respectively.

```
//api call
await fetch(`http://api.weatherapi.com/v1/forecast.json?key=9c2171a0d096437db48101007230209&q=${(props.props.coords.latitude)},${(props.props.coords.longitude)}&days=3`,{
method: "GET",
headers: {
    Accept: 'application/json',
    'Content-Type': 'application/json'
}
```

```
await fetch(`http://api.weatherapi.com/v1/forecast.json?key=9c2171a0d096437db48101007230209&q=${props.props}&days=3`,{
method: "GET",
headers: {
    Accept: 'application/json',
    'Content-Type': 'application/json'
}
```

Home Page

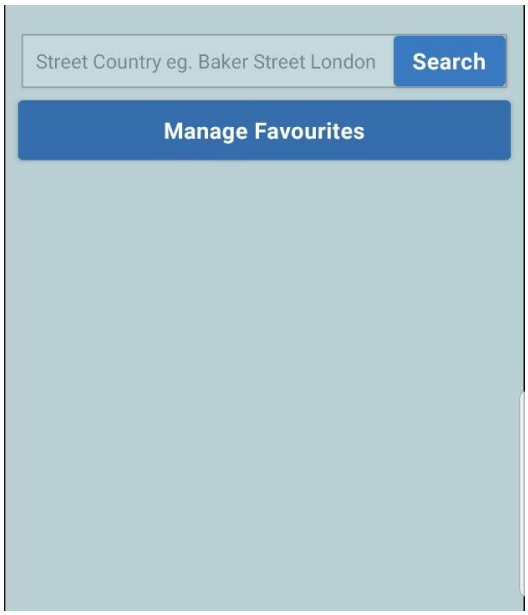
Swipe Function

The home page contains three scrollable components indicated by the arrows: the page itself, the hourly forecast, and the daily forecast. In order to execute this the page scrolling utilises Swiper from 'react-native-swiper' whilst the hourly and daily forecast utilises ScrollView from 'react-native-gesture-handler' This is to prevent a clash between the two different types of scrolling due to the swiper function being based on the ScrollView function from react-native.

As for the data in the various components: when the API returns the data, it is passed onto a hook where the data is parsed into the respective data required and displayed in its respective components as seen below. The data parsed includes all weather data as well as the icons used.

```
            console.log('hometemplatedata fetched');
            // Assign to hook
            setApi(json)

        })
        .catch((error) => {
            console.log('failed to fetch data')
            console.log(error);
        })

    })();
    }, []);

if(apidata != null) {
    //Parse Data

    const current = apidata.current;
    const date = new Date()
    const forecast = apidata.forecast.forecastday;
    const hourForecast = apidata.forecast.forecastday[0].hour;
    const location = apidata.location.country;
    const dayForecast = [];
    for (i=0;i<forecast.length;i++){
        dayForecast.push(apidata.forecast.forecastday[i].day);
    }

    const hourly = []
    for (i=0;i<hourForecast.length;i++){
        var time =  new Date(hourForecast[i].time) - new Date();
        if(time>0){
            hourly.push(hourForecast[i]);
        }
    }

    const days = ['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday'];
```

Menu Page



Search Function

In order to provide accurate weather information, the location input by user is the location used in the API call.

Manage Function

Navigates the user to the favourites page where they can view their favourited locations

Favourites Page

Display Favourites Function

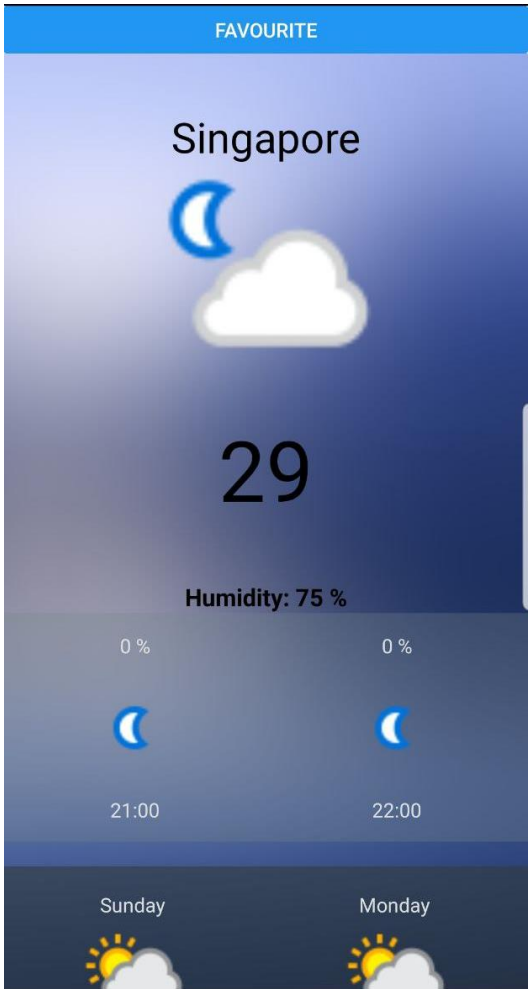
In order to display users favourite location the page retrieves all the data in the storage and maps it to each display and delete component as seen in the code below

Delete Function

The favourites pages utilise the delete function to allow users to remove locations they have saved by utilising AsyncStorage.removeItem(key) to delete the data of the respective key

```
fav.map((f,index)=>{
    console.log(f)
    return(
        // display them
        <View
        key = {index}
        style={styles.locContainer}>
            <Text style = {styles.favLocation}> {f}</Text>
            <Pressable style={styles.deleteBtn} onPress={()=>removeData(f)}>
                <Text style={styles.deleteText}>Delete</Text>
            </Pressable>

        </View>
    )
})
```

Search-results Page



Favourite Function

The search-results displays data similarly to the home page but instead utilises a favourite button instead of a menu button in order to allow users to add the location to their storage. This is done by utilising AsyncStorage.setItem(key,value) which saves the location input by the user to allow for retrieving when the application is open. By storing the data and the key to be the same it prevents the user from being able to store the same location multiple times as well as aid with data retrieving. This can be seen in the code below

```
<Button
title = "favourite"
onPress={()=> storeData(search,search)}/>
```

Evaluation

Whilst writing this programme the main difficulties was the usage of using multiple scrolling components on the home page, ensuring that hooks were being correctly used due to their heavy participation in data handling and navigation handling.

Though I feel that I was able to execute the functions and design I wanted limitations from the API such as the inability to call for bulk data and the day limit on the forecast function only allowing data from the next 3 days to be able to be called prevented certain improvements I would have like to implemented.

An improvement I can see being done include a change in the weather icons. I utilised the weather icons provided by the app in order to maintain data consistency and ensure that my own personal interpretation of each weather icon will not affect the data accuracy. However, I feel the icons are low in quality and bring down the visuals of the application.

Another improvement I would carry out would be to also add the percentage of precipitation at the current time as seen in my wireframe. However, it had to be removed due to the API not providing that data.

In conclusion, though I have successfully executed the design and functionalities I had sought out, I feel that with the removal of certain limitations further improvements could be executed.

References:

Research images taken from Weather app

API images taken from https://www.weatherapi.com/signup.aspx