Weather-Fy: Documentation

**a. Introduction: Introduce your web app and its purpose.**

Our web app is for university students and working adults to receive daily updates on the weather in their area. Our app provides the user with up-to-date weather conditions from humidity percentages to the likelihood of rain. Our web app accurately displays these data in charts that are easy to interpret by the everyday person. We know how busy people can be and so we aimed to create a web app that would allow its users to come on, find what they need, and then leave.

**b. Usability goals: Describe the usability goals you set for your web app and explain how you addressed each goal.**

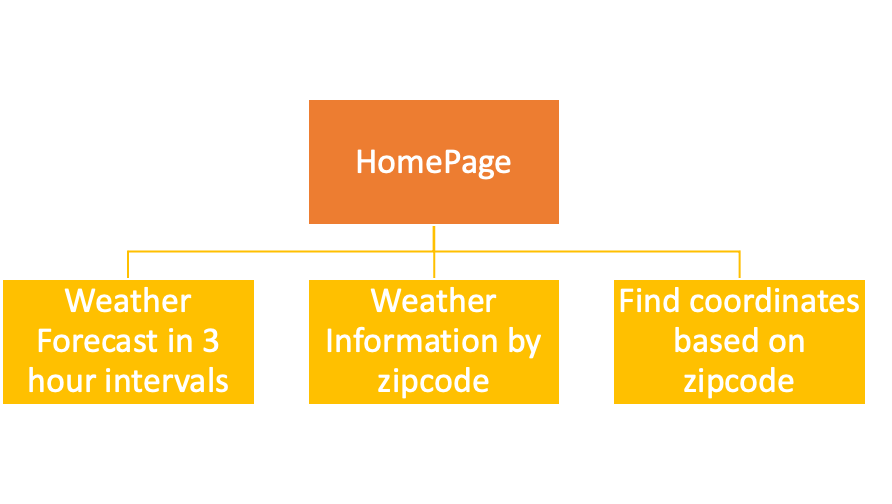
The usability goals for our web app are focused on its effectiveness, efficiency, and utility. Here we want to be able to present an app that does its job and does it well. With its effectiveness, we are able to effectively search around different cities in the US for weather updates whenever we want to. Additionally, we also want to be able to effectively present these weather metrics through various APIs with clear and concise data to the users. Next, is that we also aim to be able to do this efficiently with the use of various APIs and the utilization of Streamlit’s library, making it simple and easy to code and edit in the long run. And lastly, aim for the applications’ utility to be fully realized as we ultimately want to display the data and charts that we need, and be able to display coordinates, locations, and zip codes without the fluff, and nothing but straight-up necessary data.

**c. Design process: Discuss your design process, from sketching to implementation.**

- Goal Identification: Weather-Fy goal is to provide university students and working individuals with accessible weather information with minimal effort, so that they can take the appropriate precautions.

- Scope: After the group was formed, we decided to meet twice a week via Discord to share our updates on the final project. Starting date: July 5th. End Date: July 27th.

- Sitemap: With the intent of not having overwhelmed users, we decided to divide the web application is the 4 pages shown below:



- Content creation: Since the web app’s essential information is the weather, we opted for keeping the text as short and informative as possible while making use of maps and interactive widgets to make the web application engaging.

- Visual elements: We opted for including a map, images, and charts in order to avoid making the essential information boring.

- Testing: Due to the nature of the framework used to make the web application, testing was successfully done during the development/implementation phase.

**d. API integration: Explain how you utilized the APIs and discuss any challenges or limitations encountered.**

The trickiest part was searching for an ideal API that would allow us to retrieve weather information with as many calls per day as possible. At first, we found an API with good access to weather data for free, but we ended up having to discard its use due to the very limited number of calls allowed. Ultimately, we decided to use an API with not as much data but essential information and a significant increase in the number of calls allowed per day.

When working with the API, there were no major challenges other than finding the URLs that our free API key would have access to. After the appropriate URLs were found, it was a matter of understanding how the data retrieved was organized in order to be able to extract it and display it in the web application.

**e. Interactive widgets: Describe the widgets you incorporated and their purposes.**

The widgets that we incorporated are as follows:

* Textbox: to be able to find cities, and search around through their ZIP codes as needed
* Color Picker: to change the line color for different metrics in the web app.
* Radio button: to be able to change into either Celsius or Fahrenheit
* Button: to search a city or a coordinate, based on ZIP code.
* Checkbox: to check sunset and sunrise times
* Dataframe: to display min and max temp

**f. HCI design principles: Discuss how your web app adheres to HCI design principles.**

Provide clear visual cues and feedback to guide users. Use prominent buttons, icons, and calls to action to make important elements stand out. Offer feedback for user actions, such as showing loading indicators during processes or success messages after completing tasks.

**g. Testing and feedback: Summarize the results of your usability testing and any changes made in response to feedback.**

Once we had a mockup ready, we decided to share the results with the rest of the team in order to get some feedback on any weaknesses or strengths in the design of each page. There were many changes that had to be made mainly in response to having an interactive web application in order to make it more engaging. This is why the following were added: legend, map, images, and selection widgets (such as a select box).

In terms of functionality, each developer made sure to test all parts of their assigned page to ensure the system correctly responded every time progress was made. Finally, we decided to make the navigation between pages intuitive and simple by having a sidebar on the left side of the web application. When merging the pages each developer worked on, appropriate testing took place in order to verify the navigation experience was on point.

**h. Conclusion: Reflect on your experience and discuss potential future improvements.**

Overall, the making of a weather web application with the usage of Streamlit and OpenWeatherMap was a very enriching experience. From being able to build the design of the application in a relatively short number of steps to researching the API availability and accessibility, this project allowed the team to be self-driven and gain experience with API data manipulation.

Potential future improvements:

- Implement more information by considering the usage of other available APIs.

- Potentially explore more widgets available in Streamlit to benefit from better user interaction with the web application.

- Make more testing with a broader group and not limit the usability testing to any potential bias that may arise from having the development team make this testing.