**CEN 4010 Principles of Software Engineering, Spring 2023**

**Group 10**

**Team Name:** Group 10

**Project Name:** Weather App

**Scrum Master:**

Diego Gama - dgama2021@fau.edu

**Product Owner:**

Liam Swank - lswank2020@fau.edu

**Backend Lead:**

Alexander Mendoza - Alexandermen2017@fau.edu

**Frontend Lead:**

Jarod Crush - jcrush2016@fau.edu

**QA Lead:**

Giovanni Smith - giovannismit2016@fau.edu

**Documentation Date:** 3/3/23

**History Table**:

**3/3/23:** First submission of Milestone 1

**3/21/22:** Inclusion of risks and more information regarding high-level system architecture

**Group 10 - Weather App**

**Executive Summary**

Our website wishes to provide its users various information regarding the weather of the current day and every other day throughout the week. This information includes the predicting weather forecast such as rain and snow and the predicted high and low temperature of those days within a given city. While there is a myriad of other competitors such as weather.com and accuweather.com’s weather apps. Our website will be open source for anyone to access the source code and do with it however they please. Our product will also contain zero advertisements as this project is non-profit. Without the help of Weatherwidget.io this website would not be possible.

**Competitive analysis**

Analyzing competitive products available today. Present competitors’ features vs.

your planned ones. First create a table with key features of competitors vs. yours. Only at a very high level, 5-6 entries max. After the table, you must summarize what are the planned advantages or competitive relationship to what is already available.

| **Companies** | **Our product** | **weather.com** | **accuweather.com** | **forecast.weather.gov** | **wunderground.com** |
| --- | --- | --- | --- | --- | --- |
| Open source? | ✓ | X | X | X | X |
| Ads? | X | ✓ | ✓ | X | ✓ |
| Simplicity? | ✓ Only weather info, no nonsense | X News articles all over the page | X News articles all over the page | X A lot of charts not for the common user | ✓ On the simpler side |
| Customizable? | ✓ Even dark / light mode togglable | X Stagnant web page | X Stagnant web page | X Stagnant web page | X Stagnant web page |
| Responsiveness? | ✓ Blazingly fast | X Can be slow due to bloat | X Can be slow due to bloat | X Can be slow due to bloat | X Can be slow due to bloat |

Our product will be open source, this means that we will strive to allow as much customizability and transparency to the public as possible. For many this means respecting their privacy and holding their trust. Since this product is free and open source, we will not add ads to the webpage in order to keep it unobtrusive and true to its use case. The page will be as simple and clean as possible so that wherever you are navigating you are navigating towards more weather information, there will be no random news articles sometimes disguised as ads. We will add a feature to toggle between background colors on the webpage so that you may use it as a home page and customize it to your heart’s content, this will include a dark mode and light mode. Our webpage will be faster than our competitors because it will be simple and to the point, no extra code bloat and motives to get the user to do something other than use the tool as needed.

**Data Definition**

**Key Terms**

**Development Environment –** Regarding the development of this program's backend the use of technology such as node.js will be utilized; to create mock-ups of this project. This is also going to be applied to front-end development. .

**Website / Site** -https://mendoska.github.io/cen4010\_sp23\_g10/

**Server –** Currently being hosted on GitHub.

**Styling –** A big part of this app’s appeal is being able to customize background colors, and themes, based on the user's preference. There will be a Style tab to refer to this.

**Location -** Given by users in the form of GPS or input of area (city, state, zip code, etc.).

**MongoDB -**  A database that will assist with User management.

**User -** An individual who visits the website and/or submits their email, zip code, first, and last name for email notifications.

**Security -** Encryption to protect user safety.

**Weather -** Temperature fluctuations and the other factors that are felt throughout the days of the week.

**Reports -** Breakdowns of the weather sent to users.

**Weather Widget** – An Open-source weather widget that will contain the leading source of information for our website. It will contain the current temperature at a designated location as well as the high and low for that day.

**Type of users:**

* An everyday person who wants to check the weather in their area.
* Organizers of outdoor activities (sporting, concerts, community)
* Pilots
* Operators of sea vehicles
* Programmers / Developers due to open-source nature

**Actors**

Concerning the users, *all users are seen as actors* in this system of our website. When using the system, the person, or actor in this case, will access our website and depending on the location, will be given information about the weather. To access this website, the person(actor) will use their computer (actor) and communicate with the database system (actor) of the server (actor) on which this site is located.

**Overview, scenarios and use cases**

This section describes the project overview (in much more details) and likelihood usage scenarios of your product from end users’ perspectives. Focus only on main 2 use cases. Simple text format is OK and preferable – tell us a story about who and how is the application used? Focus on WHAT users do, their skill level, not on HOW the system is implemented. You can expand use cases provided in high level document in future milestones.

The likely usage scenarios for our product are simple at face value because of the manufactured simplicity of the product yet gets more complex the more the end user is considered. One instance is a man who spends his weekends fishing. He relies on the simplicity and reliability of our product in order to check the weather conditions for his hobby seamlessly. He has a laptop and checks the weather on it first thing in the morning to see if the week’s predictions were wrong (rain all week). He sees that the percentage chance of rain is slightly reduced and becomes hopeful. He leaves the page up on his computer and customizes the background to a much more appealing shade and then does chores around the house. He comes back every so often checking if the weather lets up so he can fish, and it eventually does. He is ecstatic and packs his things for fishing and closes his laptop. He pulls up our product on a webpage on his phone so he can check while fishing and on the go in general. Nothing hampered his experience through unnecessary stories or ads because our users want to use this app to check the weather and that's it.

A second use case is with a woman who is stuck in her office building overnight because of a sudden snowstorm that made the roads not drivable. She relies on our product to let her know when it's safe to go home, nothing gets in the way, and it gives her the safest experience and reliable responsiveness. She changes the color to a less energy intensive one so that she may save any battery she can since she does not have a charger. The lack of ads and other bloat also allow for maximum battery efficiency when it is absolutely necessary. Our product is created to assist the user and act as a tool. We do not want the user to do something or push something on the user as we know how valuable the weather information can be for most.

**High-Level Functional Requirements**

1: Below the weather forecast for the current hour and day you are able to drag your mouse across the bar below to see the forecast for the other hours throughout the day.

3: Clicking on the *i* in the upper right corner will allow the user to access a link that can take them to another webpage made by weatherwidget.org that asks the user to download their app on their Android devices.

2: The link below everything sends the user to the sign-up page in which the user can input their first and last name, email, and zip code. This information is then stored into our MongoDB database.

3: On each page the user will have access to a light mode/ dark mode switch and when clicking it will change the mode accordingly through simple css changes.

**Non-Functional Requirements**

1: Encrypting each account to ensure user safety

3: Eventually only allow certain IP’s

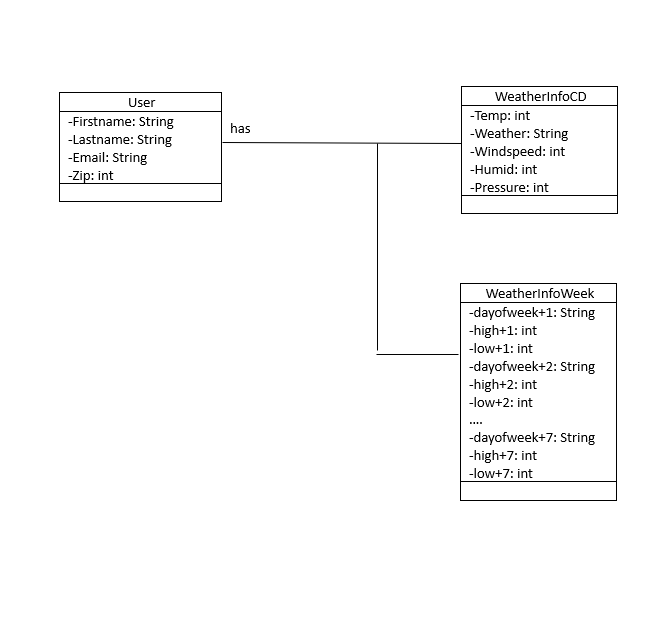
1: None of the back end will be visible when inspecting pages so the connection to the database is secure.

**High-level System Architecture:**

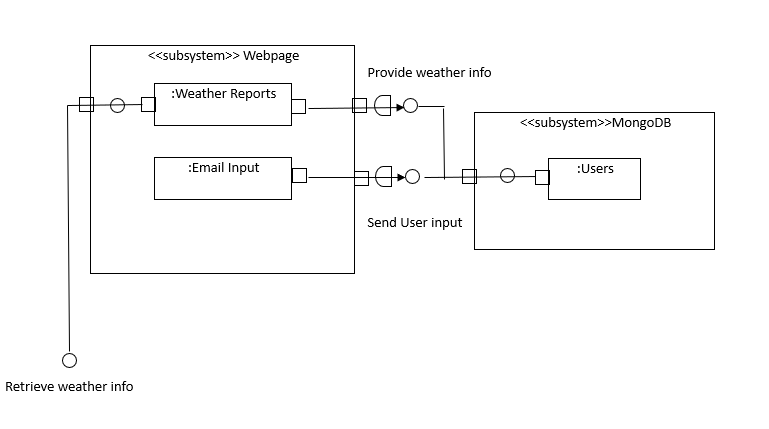
The languages we will be using for this project are HTML for the website's core structure, CSS/Bootstrap for the client side of the website, and Javascript for any other aspect of the page that may need a scripting language. We will be using two software products, Github and Github Pages, the former for storing out code and the latter for displaying our website. Brackets will be used as our main IDE for everything with the website. Github Pages works with Apple Safari, Google Chrome, Microsoft Edge, and Mozilla Firefox, thus our website will also run on these browsers. Weather widget at weatherwidget.org will be used to actually track the weather information. Our database will be hosted on Mongodb via cloud, which will only have one collection titled “Users” with each document containing the User ID (primary key), first name, last name, email address, and their zip code. All of the images, videos, and audio will be stored in the file systems rather than the DB since our website will not contain mass amounts of such data. The only implementation of searching/filtering within our project will be Ajax which obtains a user’s information from the frontend to send them emails. After the information is obtained from the user, the POST request API will send the information to the DB. When the information has reached the DB the user’s information will be given a primary key for easier identification and converted to a document in .json format.

**High-Level UML Diagrams**

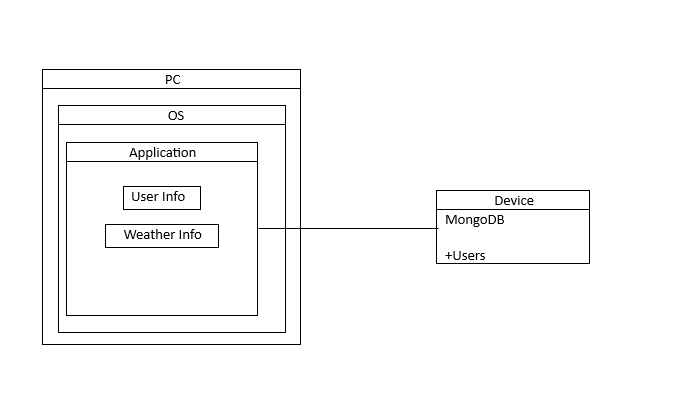
High-level UML class diagram:



Component Diagram:



Deployment:



**Risks**

Skills - Two to three of our group members don’t have a lot of experience with web development. The members that do have experience have been helping them learn the processes that go into such a field.

Schedule/Teamwork - We have one member who has an extremely tight schedule and thus has trouble finding the time to actually help out. We resolve this by communicating with this member ahead of time to plan out when they are able to find the time to work on the project. This member also is lacking in web development skills so it’s not too much of a concern when they have trouble finding time.