WeatherMen by the Weather Boys

Members

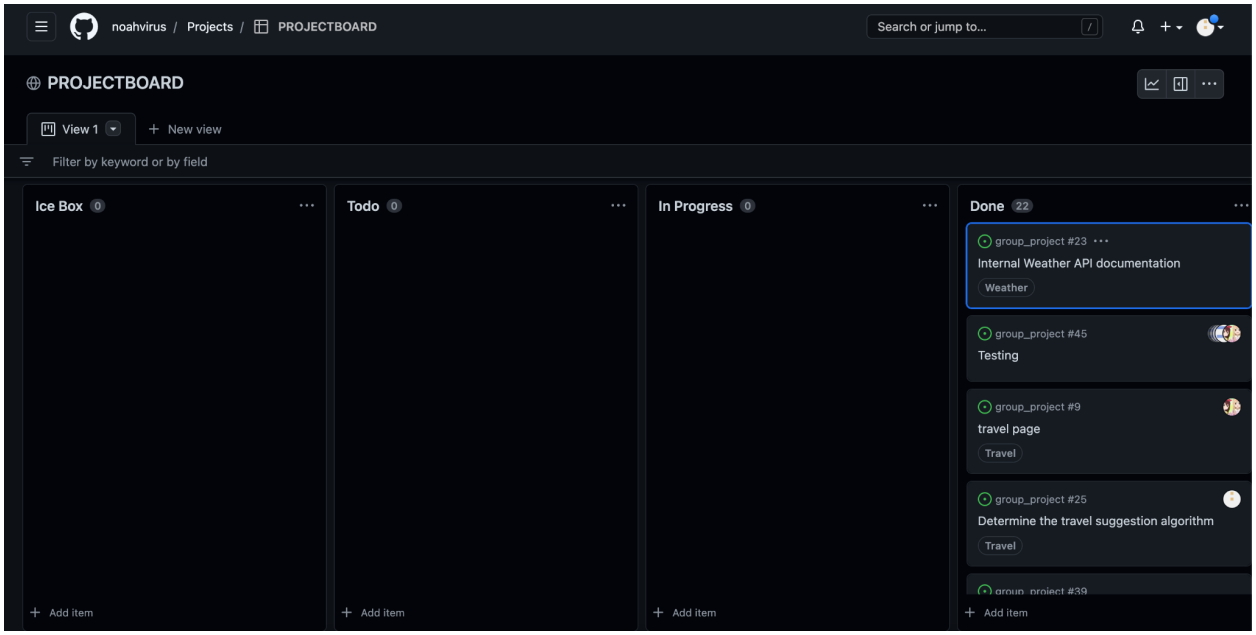
Ian Fernandez
Jay Bentley
Noah Vilas
Matthew No
Gustavo Sanchez-Sanchez
Will Dravenstott

Project Description

This project has generated a weather website where users are able to find weather information for any city around the world, get recommended clothing information, and recommendations for travel based on cities located in the user's favorites. Through the use of a valuable weather api called “weatherapi” we are able to gather lots of real time climate data. This data can then be used to display to users or determine certain outputs, such as for the clothing page. The website is also using a database which enhances functionality and user experience. This enables the profile page to work where tables of data correspond cities to users. This additionally helps the travel suggestion page as cities with similar climates to those in the users profile page are displayed.

Project Tracker

https://github.com/users/noahvirus/projects/1



Video

https://github.com/noahvirus/group_project/blob/main/milestone_submissions/Group6_Demonstration.mp4

VSC

https://github.com/noahvirus/group_project

Contributions

Ian Fernandez:
During the project process I contributed to many different aspects of the website. This included assisting the team, specifically Will, in learning how to use the API calls from the weather api that we chose. Afterwards, I created a search function which explores the weather api and returns a result on a new page. I additionally created a "clothing" button which allows the user to view the recommended clothing depending on the weather of said city. Please note that I did not create the page, I only created the button and the indexjs code. I also implemented the favorite button, which was created by jay, to work through the search function.

Jay Bentley:
I worked on a lot of the planning and framework of the project. I wrote the HTML, ejs, and node.js functionality for the discover, and profile pages. I also planned and implemented the database schema. I created and implemented the favoriting and removing favorites for the user as a button that calls a Node API call to remove the usertocity connection. I was in charge of the release notes and used the releases feature on github for generating those. I created minor features like generating a logout and a login navbar and detecting whether a user was signed in to display the correct navbar using ejs.

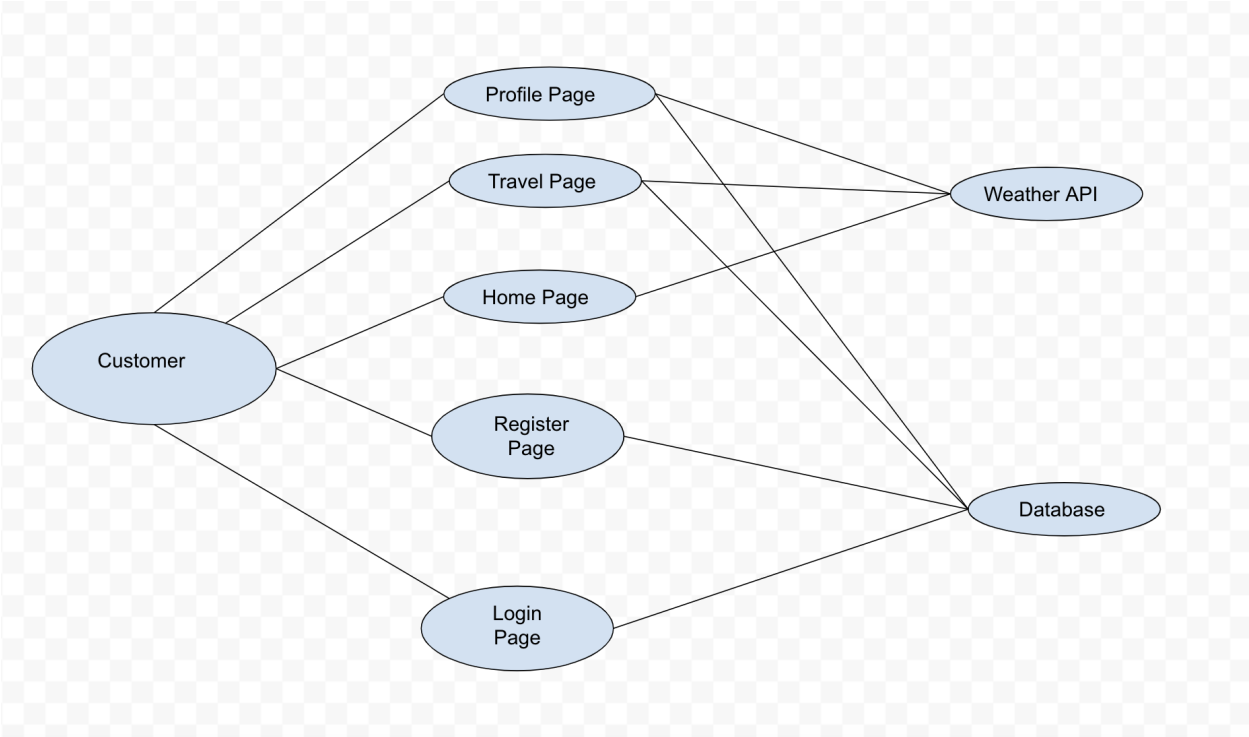
Noah Vilas:
Throughout this project I worked on various aspects of the website, the first thing I accomplished was the login and register pages of the website, followed by assisting my team members with various aspects of the website including multiple javascript functions as well as the creation of many of the other pages. As well as, coding the styling of the website.

Matthew No:
During the duration of the project, I contributed to various aspects of the website. With just starting out on helping with the visualization of the travel page to suggest users on what other users have chosen, keep note that we did not add a button to keep it unique from other pages of the website we have made. I also cleaned up and fixed the partials for our website, just to make sure everything is connected and running when we have started to work on this project. Jay has also assisted me in the travel page from getting data of users rather than getting the data of countries from yourself for recommendations.

Gustavo Sanchez-Sanchez:
For this project I got the opportunity to work on API calls and designing a webpage. I assisted in the use of the technology to get requests for registering and uploading into the database. Another API call that I created was the get request for the homepage. The homepage is one of the notable features that I made since it loads up as a "welcome" page for the user, presenting the weather and relevant information (time, windspeed, etc.) for the top five popular cities around the world.

Will Dravenstott:
For the group project, I was able to work on both the front and back end of the website. For my contribution on the back end, I was able to figure out which api to use first, and then was able to create basic calls to be used on the website, along with Ian. There was much debugging to be done, as the calls didn't initially work, but found a way to make it produce results. For the front end, I was able to complete the clothing page, which shows variable clothing based on the weather outside, and displays it as a picture.

Use Case Diagram



Test Results

- What are the users doing?
- What is the user's reasoning for their actions?
- Is their behavior consistent with the use case?
- If there is a deviation from the expected actions, what is the reason for that?
- Did you use that to make changes to your application? If so, what changes did you make?

Feature 1:
Adding city to favorites
Test Cases

- When a city is favorited, the city should be added to the user favorites page as well as the database correlating with the favorites list.
- The user must be logged in
- The user must be on the discover page
- The user must click the “Favorite” button on the city
- The response should result in either an error or successfully added message.
- The city should be displayed in the user profile page

Test Data

- Selecting a city listed in the discover list to be added and placed in the favorites list.
- This will use the database to store this data with the logged in profile

Test Environment

- The test should be done on the QA environment
- QA database with dummy data to mimic real data

Test Results

- The test should result in one of two possible cases. One is that the city is found, but isn’t able to be favored (due to an error or it is already favorited), and the second is that the city successfully adds the city to the favorites list.

User Acceptance Testers

- Internal tester: Ian Fernandez
- Other user: Kyle Stiebeling, Karston Ferrier

Example

- Boulder, CO -> add [message: “successfully added location”]

Observations:
All users were able to complete this task without deviation from the use case and or confusion. All users were able to add cities by clicking the “favorite” button. With this in mind not all users understood where they might find their favorite page, however, upon further investigation, all were able to find and review them in the profile page. Several users even clicked the “remove” button either whilst in the profile page or upon returning to the discover page. Upon inquiry, many suggested that they simply wanted to see the functionality of said button (if it would work accordingly). Users also found it helpful that there is a message that displays to inform you that the action was performed as requested (“location added”/ “location removed”). After this testing, we added the “favorite” button to the search function to increase more functionality across the board. With this in mind, the primary function of this feature is unchanged from before testing.

Feature 2:
Login/ testing database for validity
Test Case

- When you input username and password the test should check to see if your data is stored and see if you have previously registered or logged in.
- When invalid username or password is entered, it is detected and relayed to user

Test Data

- Input a previous login data to check if successfully logged in (username, password).

- Putting in incorrect login/password to check that login does not work if those two are not correctly inputted.
- The username and password to be compared to will be stored in the database for the dev environment.

Test Environment
- QA environment
- QA database with dummy data to mimic real data

Test Results
- The User should be correctly logged in when correct username and password is entered
- The User should be redirected to the login screen again with error message if username or password was incorrect

User Acceptance Testers
- Internal Tester: Jay Bentley
- Other user: Manal Shami, Brady Kuzinski

Example:

Username: admin1234
Password: PASSWORD
Login Successful!

Observations:
This task was very straightforward for many users. All users were first required to register an account so that this information can be put into the user database (this could also be done through manual input upon database creation). During the login process, there were some events where users were confused as to what happened. We had not implemented error messages or other functions to the feature, therefore, upon incorrect inputs, the users were redirected to the register page and were confused. After the round of testing, we added the error features so that the user is aware of what went wrong. There are now three outcomes that can occur during the login process. Firstly all information can be correct and the user will login, being redirected to the home page. Secondly, the user can input the username correctly but the incorrect password which then results in an error message stating “Incorrect username or password,” redirecting the user back to the login page. Finally, the user can input all the wrong information. This redirects the user to the register page with an error message stating “username not registered.”

Feature 3:
Testing api call for displaying city information - search bar.
Test Case
- User must be logged in.
- User must be located in the discover.
- User will then input a city name in the search bar and hit “search”/ enter.
- User will see cities that fit the corresponding input name.

Test Data
- Testing various cities around the world
- Test using incorrect city names
- Test using another search function that isn’t a city name (i.e. postal code)

Test Environment

- QA Environment
- QA database with dummy data to mimic real data

Test Results:

- When a user searches for a city with a real name, the user is redirected to a new page. On this page should be a list of information regarding the weather of the city. This city should also be the same city as the name that was inputted into the search bar.
- When inputting an incorrect city name, there should be an error message displayed to the user. The user should also be redirected to an error page.
- When using a search function that is not implemented, such as a postal code, another error should be displayed, redirecting the user to the error page.

User Acceptance Testers

- Internal Tester: Will Dravenstott
- Other user: Christian Mcginty, Marcelo Dimatteo

Example:

Search: Paris
Enter
Results displayed with correct city

Observations:
All users were able to complete the tasks with no issues. There are no real notable observations regarding the operation of these tasks because of the simplicity in user execution. Users did find the error messages odd where an incorrect city name would result in a message, “api call failed.” This seems like an easy fix, however, for the purpose of the developers understanding what is going wrong and when, this message is unchanged. This is the same for “database failure” which occurs when the api call works but it cannot find the item in the database. This did not disrupt the users from effectively completing the task assigned. At the current moment we are unsure about creating an error page, where the user can be redirected to upon error, to separate the user and make them aware that something went wrong. *update* This error page was added to the website for the purpose stated in the last sentence.

Feature 4:
Logout success and saving of data through database.
Test Case

- While the user is logged in, they should be able to see their profile page.
- User will logout of the website.
- The user will click on discover, profile, and travel page links in the nav bar (of which all require a login) to determine if they can enter the pages.
- User will log back into website with the same credentials as earlier.
- User will view their profile page to determine if their cities are still the same favorited ones.

Test Data

- Testing database saving for users.
- Testing logout function is correctly implemented.

Test Environment

- QA Environment
- QA database with dummy data to mimic real data

Test Results:

- When the user logs out, they should not be able to enter the discover, profile, or travel pages.
- When the user logs out and logs back in the user should be able to see the same cities they favorited earlier still saved.

User Acceptance Testers

- Internal Tester: Matthew No
- Other user: Riley Smith, Sebastian Net

Example:

N/A

Observations:

Before users logged out, we requested that each user favorite any three cities that they wanted to and to view them on the profile page. All users were then able to complete this user task by clicking the logout button. No user was able to access any of the three web pages that were listed above while they were logged out. The error message that was displayed did assist in user understanding of why they could not access said page that was being clicked on. There was no feedback given regarding this error message. After logging back in, all users were able to see their favorited cities once again in the profile page. There are no noteworthy observations for this test. It is only significant that all expected functions of this feature were able to work correctly. Nothing was then changed after these tests.

Deployment

In order to run this report, you must have docker installed along with all of the corresponding code. Before running docker, you must input the following command into the command line: rm -rf node_modules. After this is completed, you are able to start up docker and go to localhost:3000 through a web search and use the site.