Name: Mingwei You (2837632), Rafael Carrillo (2746190)

Date: 12/4/2025

Project Report

CalcWeb

Mingwei You (2837632), Rafael Carrillo (2746190)

1.0 Introduction

In this document, we will report the website "CalcWeb" of:

- The idea
- Setup
- Structure

1.1 Idea of Project

The idea of this project initially was to develop an inner map for CSU. However, after the discussions and thoughtfulness, we decided it is easier and less difficult for us to simply develop a website that contains a calculator and other functions that can be included, so that we learn lessons from the project and know how to set the structure of a website in the future.

1.2 Structure of Project

Since the project was decided to be in a website, we used HTML, JavaScripts, and CSS languages to program the website software. We use these programming languages because it also has a good side: HTML in GitHub can set up a live page, which allows users to view how the pages look without having to download the files and run on local.

1.3 Challenge/Issue

Before starting the project, even though we discussed how the project will progress and accomplish, due to having different schedules, we could not meet and inspect the program in real-time, instead we had to use social media and GitHub to discuss and transfer the codes that we had.

2.0 Interface

In this section, I (Mingwei) will report the elements of "CalcWeb" such as the structure of interface and related functions.

2.1 Structure Design

Initially, the interface design was very simple: white pages and a calculator. After taking another look, I decided that it looked boring. First, I added different elements (e.g. header, sidebar, animation) to the website, so it will look better looking. Second, I added colors corresponding to each element, with a different set of color themes that can be used.

2.2 Calculator Design

After thoughtfulness, I decided to design a traditional looking calculator on the website, nothing fancy. Beside designing the calculator interface, I also added a function that allows users to click on the calculator display bar and can type numbers/operators.

2.3 Other Design

Initially, the buttons for color theme, history, tutorial and others were designed to be on the left side of the calculator. The problem is, the buttons will look squish and tight. So the buttons got redesigned, to be putted in different sections, such as: color theme got its own button (with list of color theme) on the top left corner of the website; history, tutorial, and about were putted inside the "MORE" button next to the color theme button, and can be viewed with a sidebar slide out when clicked.

2.4 Challenge/Issue

A lot of challenges were faced as I was structuring the codes for interface:

1) Usual challenge: code conflict, structure failed.
2) As soon as I finished the color theme, I was going to implement a customizable color theme, but due to the limit of the time, I could not structure the code within enough time and had to remove this idea off from the list.
3) When implementing the calculator frame and advanced function button frame together, the arrangement of both look normal on my end, but after resizing the window, the arrangement override each other. To fix this problem, I had to use *z-index* to set the priority of each element frame, adjust the position of each frame, and check the frame under different screen sizes.

3.0 Arithmetic Function

In this section, I (Rafael) will report the elements that are related to the arithmetic used in the calculator.

3.1 Basic Arithmetic

For the calculator to work, I was thinking of ways that would have the numbers and symbols to display in the calculator while also maintaining how the functions work based on the operators themselves. At first, I wanted each operator to have their own method with their own logic, and have it working in the display. However, I came across issues where I also have to include other things such as decimals and fractions so it became a challenge. What I did instead is search online for resources as someone who is not familiar with javascript. What I did instead is instead of separating the operators into their own method, I instead want them as text based, and have them used as a way to evaluate two or more numbers. When I found this out, it became easier to combine many ideas I have as the text itself can be used to evaluate, which includes the decimal that I mentioned. Because of this, it is possible to use the buttons on the calculator to form a valid operation (such as 5 + 5), and have the operations done in text, which allows the computer to see and form a result based on the numbers and operators being used. I do have to include some codes so that users are allowed to do more than one operator when dealing with large equations.

3.2 Advanced Arithmetic

When we were implementing an advanced menu, we were wondering how we could implement these new symbols to work with the operators and numbers. My partner came up with an idea that these symbols would have extra codes to accommodate with the text, thus making it working. The button I worked on personally is the fraction function, which at the time I had no idea of how to get this to work. My idea was that it should have its own box when being called with its own method that is different from the rest that is used for buttons on the calculator. The user would type in the fraction input for both numerator and denominator so that the user can simply enter a fraction, and the codes in Computation.js would translate the numerator and denominator inputs into text. The rest was done with my panther where all, save for Pi (π), exponent (^), and Euler's number (e), are done in different calls to work with their own logic within these calls.

3.3 Challenge/Issue

One of the challenges we have faced is time constraints as we have other things to work on, it makes it hard to implement all the things we wanted so we have to cut ideas to make sure the website is completely usable. Another thing we have to consider is being able to implement the logic for the advanced functions since they would need extra codes for it to work properly and it would take us a while to get it to work properly. With the aforementioned time constraint, we decided to cut ideas that would take a while to code.

4.0 Unit Conversion Calculator

In this section, we will report the elements that are related to the unit conversion calculator.

4.1 Idea

Initially, this wasn't a part of our plan because we thought what we had was enough, but as we were going to set the status of the project done, we realized any other website already has what we have, users do not even have to use our website. So we discussed and came up with the idea of implementing a unit conversion calculator.

4.2 Conversion & Unit

Before we start to write the code, we knew this is going to take a lot of time, so we tried to look up any possible API to import into our program, unfortunately we did not find one fit our program the best, either because the style is not what we want, or it is beyond of our goal. Also, we have thought about directly implementing these codes into the calculator itself, but that means we have to change everything that we had in the files, and possibly encounter more challenges. So we had to create separated files to make the work lighter.

At the start, we wrote the code in another workspace. Once the calculation and the function works, we have to start to implement all other functions that fit and make life easier. We have come up with many ideas and achieved most of them: unit category, auto change unit category, display result in another bar, clickable overlay (allow close the sidebar upon click on the overlay), and etc. But one of the most important ideas was definitely the auto conversion function. When it comes to testing a new unit that we add, we can just type in the number in the input box and we can immediately see the result without having to click the equal button (this also made us don't have to design an equal button).

4.3 Challenge/Issue

Many challenges were met during our development:

1) Units will not convert to the correct result or correct unit.
2) The result shows an error or infinity no matter what number was in the input.
3) Unit category will not jump to the correct category or keep jumping back even without clicking.
4) The result would not show in the calculation history (we had to substitute this with the copyable result display).
5) Others possible errors or program mistakes.

5.0 Conclusion

We had learned a lot of programming skills and knowledge throughout this project, even though we have a few arguments of what we need/want in the project, we had to eventually balance the idea and take off a few things from the list due to the limited time that we have to develop. We also learned that it is important to have a backup before implementing a new function into an existing file that isn’t backup, while other functions work perfectly fine. And having a shareable real-time files IDE is really important, allowing the files to be edited by each other in real-time without having to upload to the platform and download it.