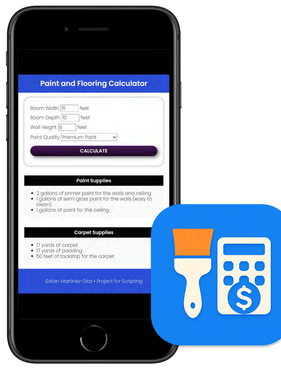
Painting and Flooring Cost Calculator App: Designing a Practical Tool for Everyday Use

Building a simple, interactive calculator that helps users estimate painting and flooring costs quickly and accurately.

BY ALAN MARTINEZ-DIAZ



Introduction

This project was an assignment I completed in Mr. Cheney's class where the goal was to create a small but functional web application using HTML, CSS, and JavaScript. I wanted to make something that felt practical, something that a person could actually use in their day-to-day life. That is how I came up with the idea for the Painting and Flooring Cost Calculator App, a simple tool that helps users estimate the cost of materials for home improvement projects.

I chose this concept because I liked the idea of designing something that combined usability, accuracy, and visual simplicity. Many online calculators can be cluttered or difficult to read, and I wanted to avoid that. My goal was to create a user-friendly interface that provided quick and reliable results without overwhelming the user with unnecessary elements. This project also gave me the chance to apply the fundamentals of front-end web development, such as creating clean layouts, styling consistent components, and writing functional JavaScript logic that responds to user input in real time.

The Process

Developing in Visual Studio Code

I began this project by outlining what the app needed to do. The main idea was to create two calculators in one: one for estimating painting costs and another for flooring costs. Each calculator needed to take in a few user inputs, perform the appropriate calculation, and display the result immediately on the same page.

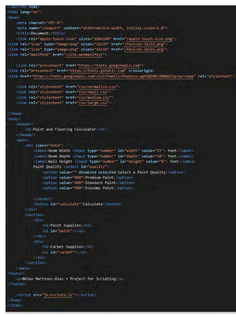
Since this was a coded project, I started directly in Visual Studio Code and focused on writing the HTML first. I built a clear and organized structure by dividing the page into sections. Each section had a heading, labeled input fields, a calculate button, and an area to display results. This structure helped ensure that everything felt balanced and easy to navigate. I made sure that the form layout followed a logical flow from top to bottom so users could read, fill, and calculate naturally.

Once the layout was set, I moved to CSS. I wanted the app to look calm and professional while still feeling friendly. I used a light color palette with shades of blue and white, giving the interface a clean and trustworthy appearance. Rounded edges and soft shadows made the design feel approachable, while padding and spacing gave the elements room to breathe. I also paid attention to alignment and used flexbox and grid to make sure the calculator remained responsive across different screen sizes.

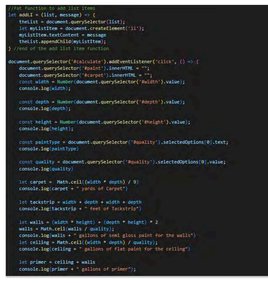
When the layout and style were complete, I started building the functionality using JavaScript. This was the most rewarding part of the project because I got to see everything come to life. I wrote functions that retrieved user inputs, performed calculations, and then updated the page with the total cost. For example, the painting calculator multiplies the entered area by the cost per square foot and the number of coats, while the flooring calculator uses just area and cost per square foot. To make the app more reliable, I included simple input validation so the app would notify users if they left fields empty or entered invalid values. This small detail made the experience smoother and showed me how important user feedback is in web design.

HTML Structure

The HTML sets up the layout of the calculator. It has two main sections, one for painting and one for flooring. Each includes input fields, labels, and buttons so users can enter their information easily and view results right below. This structure keeps the app simple and clear, making it easy for anyone to use.



JavaScript Functionality



The JavaScript handles all the logic behind the calculator. It takes the user's input from each field, performs the calculation, and then updates the page with the result. I used simple math formulas and added input checks so that the app responds clearly if something is left blank or entered incorrectly. This made the calculator more reliable and user friendly.

Styling

Each of these code examples represents a different piece of the app working together, from the structure of the HTML to the logic in JavaScript and the visual style created with CSS. Together they form a complete and practical web tool that feels responsive and functional.

Conclusion

Working on this project taught me a lot about building functionality through code and balancing it with a user-friendly design. I enjoyed the process of planning how users would interact with the app and making sure the results were accurate and displayed instantly. I was proud that I was able to create a working calculator that looked clean, felt responsive, and actually served a purpose.

If I were to revisit this project, I would like to expand it by adding more advanced features such as a materials selector that allows users to choose different paint or flooring types that affect pricing. I would also like to include a print option or a small summary report so users could save their estimates for later.

Overall this project showed me how important it is to think about the user first. The goal was to make something that worked well and was easy to understand, and I believe I accomplished that. It was also one of the first projects that made me realize how powerful JavaScript can be in connecting design with interaction. I left this assignment proud of what I built and more confident in my ability to design and code a fully functioning web application from scratch.