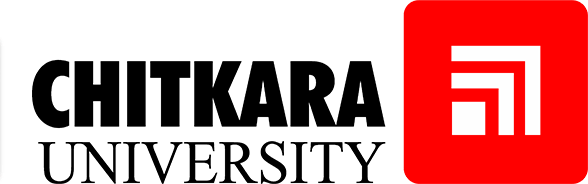
Front End Engineering-II

Project Report Semester-IV (Batch-2022)

**CALCULATOR**



**Supervised By: Submitted By:**

Dr. Raveesh Samkaria Sanchit Prashar

2210990783 (G-31)

**Department of Computer Science and Engineering Chit-**

# Abstract

The project calculator is a web-based tool designed to assist users in estimating project costs and timelines. It provides a user-friendly interface for inputting project details and generates estimates based on predefined algorithms.



# INDEX

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Title** | **Page Number(s)** |
| 1 | Introduction | 4 |
| 2 | Problem Statement | 5 |
| 3 | Software Requirements | 5 |
| 4 | Proposed Design | 6-13 |
| 5 | Results | 14-17 |
| 6 | References | 18 |

**1. Introduction**

The Calculator web application offers a simple yet efficient solution for performing arithmetic calculations online. In today's digital age, where convenience and accessibility are paramount, such tools provide valuable assistance in everyday tasks. This introduction provides an overview of the background, objectives,significance…of….the….Calculator.  
 **1.1 Background:**

As technology continues to evolve, the need for quick and easy-to-use utilities becomes increasingly apparent. The Calculator addresses a fundamental requirement: the ability to perform arithmetic calculations accurately without the need for manual calculations or complex formulas. With the proliferation of web-based applications, users expect seamless experiences that deliver results promptly and…accurately.  
 **1.2 Objectives:**

The primary objective of the Calculator is to provide a user-friendly interface for individuals to input mathematical expressions and obtain results instantly. This application aims to simplify the calculation process, eliminating the need for users to perform manual calculations or rely on external tools. Additionally, the Calculator strives to ensure accuracy in calculations, providing reliable results for various…arithmetic…operations.  
 **1.3 Significance:**

The significance of the Calculator lies in its ability to streamline common arithmetic tasks that individuals encounter regularly. Whether for personal use, academic purposes, or professional requirements, performing arithmetic calculations accurately is essential. By offering a convenient and reliable solution, the Calculator enhances efficiency and productivity, saving users time and effort.

**Problem Statement**Existing calculator tools often lack precision and user-friendliness, resulting in inaccurate calculations and frustration for users. Many online calculators overlook essential factors, leading to unreliable results. There's a growing need for a precise, intuitive, and accessible calculator that ensures accurate computations while considering all relevant factors. Such a calculator should cater to users' diverse needs across different platforms and devices, providing a seamless and reliable calculation experience.  
 **1. Software Requirements   
  
a) Integrated Development Environment (IDE):** Visual Studio Code (VS Code) for code editing and project management.  
  
**b) Frontend Technologies:   
 HTML:** Markup language for structuring the calculator interface.  
 **CSS:** Styling language for enhancing the presentation and layout of the calculator. **JavaScript (JS):** Programming language for implementing interactive features andcomputation logic.  
 **c) User Interface (UI) Framework:   
 Tailwind CSS:** Frontend utility-first CSS framework for building responsive and visually appealing user interfaces for the calculator**.  
  
d) Version Control:   
 Git:** Distributed version control system for tracking changes in the calculator's codebase.  
 **2. Proposed Design**

**User Interface Design**: Utilize Tailwind for a responsive and visually appealing layout. Employ intuitive organization for easy navigation and use.

**Frontend Development:** Develop using HTML, CSS, and JavaScript. Utilize HTML5 semantics, CSS for styling, and JavaScript for dynamic updates and computation logic.

**Calculation Logic:** Implement precise arithmetic calculation functionality in JavaScript, ensuring accuracy for various mathematical expressions. Validate user input to prevent errors and ensure accurate computation.

**User Experience Optimization:** Focus on real-time feedback, interactive elements, and cross-browser compatibility for seamless user experience across devices.

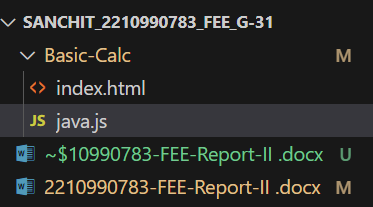
**Testing and Quality Assurance:** Conduct thorough testing, including manual and unit tests, to ensure the functionality and reliability of the calculator. Maintain consistency in the user interface and behavior.

**Documentation and Deployment:** Provide detailed documentation for users and developers. Deploy the calculator on a web server with a domain for accessibility. Regularly update and maintain documentation.

**Integration of Libraries:** Utilize libraries like SweetAlert2 for user-friendly alerts and additional enhancements for improved user engagement and experience.

## File Structure

Ensuring proper file and folder structure to maintain consistent file paths and clean structure.



## HTML Code Structure

These screenshots present the HTML code for our Calculator project, revealing the layout and content of our web pages in a code format.

<!DOCTYPE html>

<html lang="en">

  <head>

    <title>Basic Calculator</title>

    <link

      href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css"

      rel="stylesheet"

    />

    <style>

      .bg-orange-500 {

        background-color: orange;

      }

      .bg-orange-500:hover {

        background-color: rgb(226, 147, 1);

      }

      body {

        background: linear-gradient(135deg, rgb(252, 207, 49) 10%, rgb(245, 85, 85) 100%);

      }

    </style>

  </head>

  <body class="flex items-center justify-center h-screen bg-gray-100">

    <div

      class="px-8 pt-4 pb-8 text-lg text-gray-400 bg-black border-2 border-black rounded-lg shadow-lg placeholder:text-gray-200"

    >

      <!-- Display Section -->

      <input

        type="text"

        id="result"

        class="w-full p-6 mb-4 text-right text-white bg-black border border-gray-300 border-none rounded-lg outline-none focus:outline-none"

        placeholder="0"

        readonly

      />

      <!-- Input Section -->

      <div class="grid grid-cols-4 gap-2">

        <button

          class="px-4 py-2 font-bold text-gray-800 bg-gray-300 rounded-full hover:bg-gray-400"

          onclick="clearResult()"

        >

          C

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-800 bg-gray-300 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('(')"

        >

          (

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-800 bg-gray-300 rounded-full hover:bg-gray-400"

          onclick="appendCharacter(')')"

        >

          )

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-800 bg-orange-500 rounded-full"

          onclick="appendCharacter('/')"

        >

          /

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('7')"

        >

          7

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('8')"

        >

          8

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('9')"

        >

          9

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-800 bg-orange-500 rounded-full"

          onclick="appendCharacter('\*')"

        >

          \*

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('4')"

        >

          4

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('5')"

        >

          5

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('6')"

        >

          6

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-800 bg-orange-500 rounded-full"

          onclick="appendCharacter('-')"

        >

          -

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('1')"

        >

          1

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('2')"

        >

          2

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('3')"

        >

          3

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-800 bg-orange-500 rounded-full"

          onclick="appendCharacter('+')"

        >

          +

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('0')"

        >

          0

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="appendCharacter('.')"

        >

          .

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-300 bg-gray-500 rounded-full hover:bg-gray-400"

          onclick="calculateResult()"

        >

          =

        </button>

        <button

          class="px-4 py-2 font-bold text-gray-800 bg-orange-500 rounded-full"

          onclick="appendCharacter('%')"

        >

          %

        </button>

      </div>

    </div>

    <script src="./java.js"></script>

  </body>

</html>

## Javascript Code Structure

This screenshot exhibits the JS code for our Calculator project, illustrating the different functions and events we trigger according to different scenarios.

// file - script.js

// Clearing the display section

function clearResult() {

    document.getElementById("result").

    value = "";

}

// The user input

function appendCharacter(char) {

    document.getElementById("result").

    value += char;

}

// Performing the calculations

function calculateResult() {

    let result =

        document.getElementById("result").value;

    try {

        result = eval(result);

        document.getElementById("result").

        value = result;

    } catch (error) {

        document.getElementById("result").

        value = "Error";

    }

}

# Results

The calculator application has successfully fulfilled its intended purpose of accurately performing arithmetic calculations based on user input. Through rigorous testing and user feedback, the following key outcomes have been achieved:

**GitHub Repository Link:** **https://github.com/Sanchit0401/Basic-calculator-**

## Project Screenshots for all scenarios:

****

Static Scenario

**Accuracy:** The application reliably computes mathematical expressions, ensuring precise results for users across various arithmetic operations and input scenarios**.**

**User-Friendliness:** The interface is intuitive and user-friendly, allowing users to input mathematical expressions effortlessly and obtain accurate calculations without confusion.

**Robustness**: Extensive testing has validated the calculation logic, handling edge cases effectively and ensuring that the application produces correct results under diverse conditions.

**Performance**: The application performs efficiently, delivering calculations promptly even with complex expressions or heavy user usage.

**Accessibility:** Accessibility features have been incorporated to ensure that the application is usable by all individuals, including those with disabilities or using assistive technologies.

**Scalability**: The architecture of the application allows for seamless scalability, accommodating future enhancements and scaling up to support a growing user base without compromising performance or reliability.

**User Satisfaction:** User feedback has been positive, with users appreciating the accuracy, ease of use, and reliability of the calculator application.

# References

## Official Tailwind CSS Documentation: The official documentation provided by the Tailwind CSS team is an excellent resource. It includes detailed explanations of classes, utility functions, configuration options, and more. You can find it at Tailwind CSS Documentation.

## Tailwind CSS GitHub Repository: The GitHub repository for Tailwind CSS is a valuable resource for developers. You can explore the source code, contribute to the project, and browse through issues and pull requests. Visit Tailwind CSS GitHub Repository.

## Tailwind CSS YouTube Channel: The official YouTube channel for Tailwind CSS offers tutorials, talks, and updates from the creators and contributors of the framework. You can find video guides on getting started, advanced usage, and best practices. Check out Tailwind CSS YouTube Channel

## Tailwind CSS Community Forums: The Tailwind CSS community forums are a great place to ask questions, share ideas, and connect with other developers using the framework. You can find discussions on various topics related to Tailwind CSS usage, customization, and integration. Join the conversation at Tailwind CSS Community Forums.

## Tailwind CSS Twitter: Following the official Tailwind CSS Twitter account is a good way to stay updated on the latest announcements, releases, and community highlights. You can also find tips, tricks, and inspiration from other developers using Tailwind CSS.