Project Report: Simple Calculator

Introduction

- **Project Name**: Simple Calculator (Intermediate Level)
- **Developed By**: Gungun Jain
- **Roll No.**: 2200290120071
- **Branch**: Computer Science (CS)
- **Technology Stack**: HTML, CSS, JavaScript
- **Purpose**: The goal of this project is to develop a simple, responsive calculator that efficiently performs basic arithmetic operations—addition, subtraction, multiplication, and division. The design emphasizes a clean and intuitive user interface, ensuring smooth user interaction.

Project Objectives

- **Basic Features**:
  - Perform core arithmetic operations: addition, subtraction, multiplication, and division.
- **Advanced Features**:
  - **Percentage Calculation**: Enables percentage calculations (e.g., calculating "20% of 200" returns 40).
  - **Square Root Calculation**: Includes a square root function for quick access to square root values.

Project Requirements

- **Frontend**:
  - **HTML**: Structures the calculator layout, providing a well-organized interface.
  - **CSS**: Styles the calculator for aesthetic appeal and ensures a responsive design for various screen sizes.
- **JavaScript**:
  - **Functionality**: Handles inputs, performs real-time calculations, and dynamically updates the display.

  - **Event Handling**: Utilizes event listeners to capture button clicks and trigger the corresponding operations.

System Design

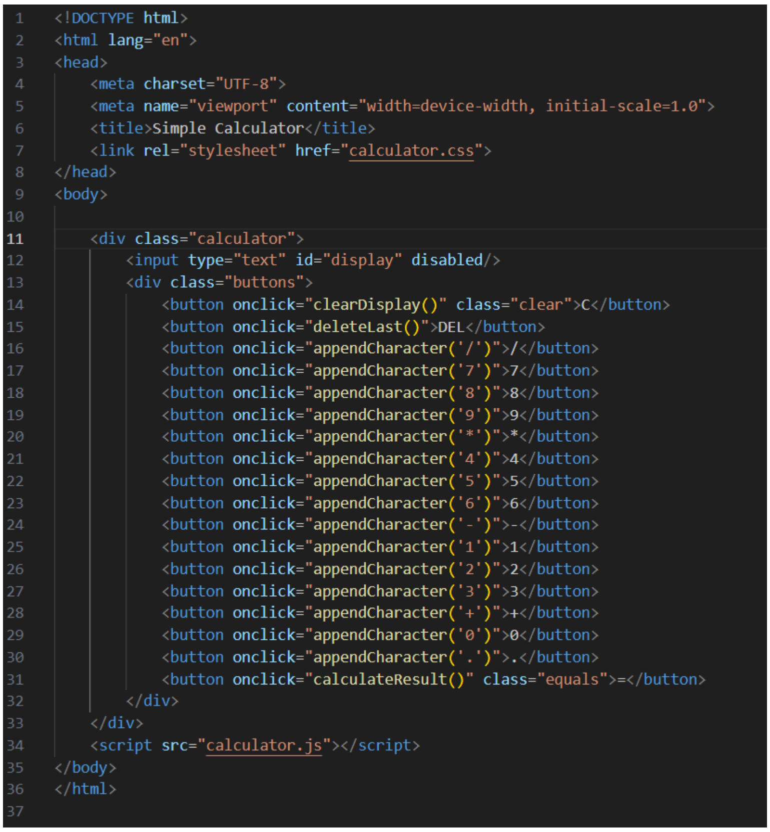
- **User Interface**: A simple, clear layout featuring buttons for numbers (0-9), core arithmetic operations (+, -, *, /), and an interactive display screen.
- **Event Handling**: JavaScript event listeners respond to button clicks, process the user input, and update the display in real time.

Implementation

- **JavaScript Logic**: JavaScript functions perform essential arithmetic operations such as addition, subtraction, multiplication, and division.
- **User Interaction**: User inputs are captured through button clicks, and the results display dynamically, enabling real-time feedback.
- **Testing**: The calculator undergoes thorough testing to verify calculation accuracy and ensure smooth UI interactions.

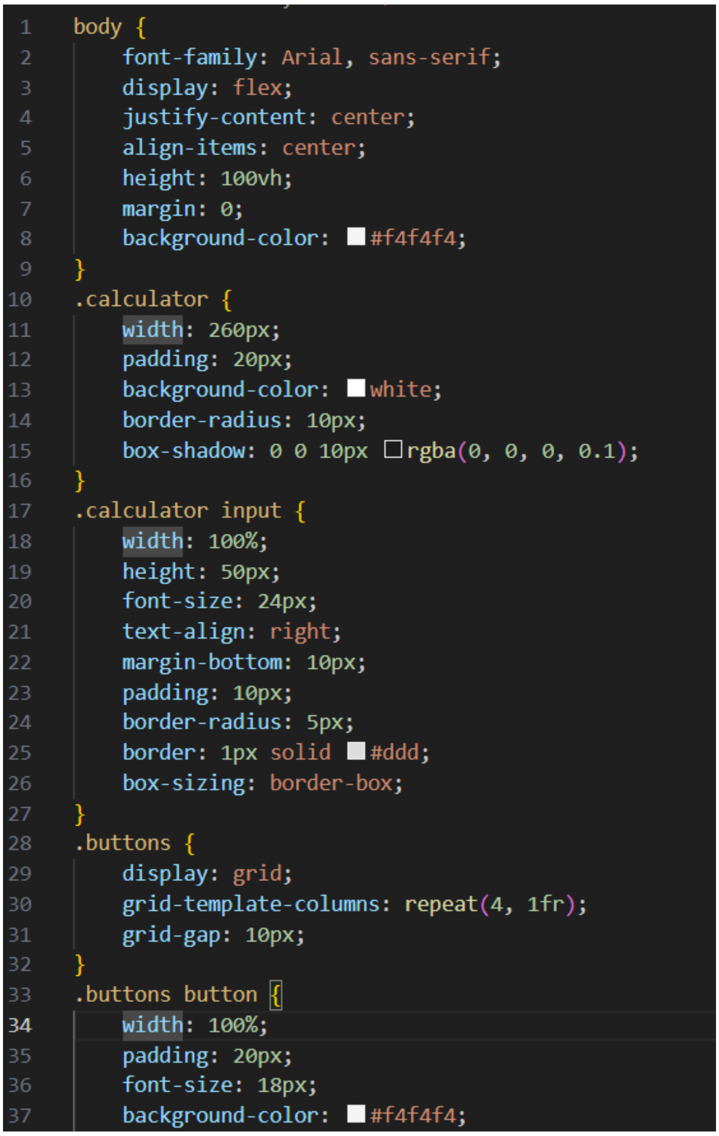
Code Implementation

HTML

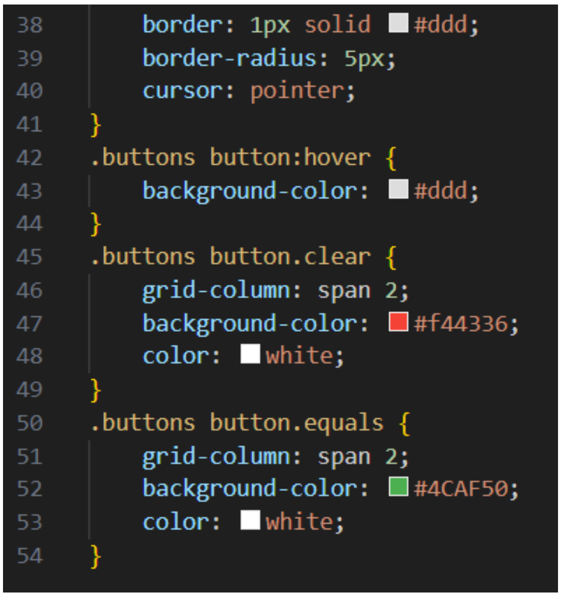


```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Simple Calculator</title>
    <link rel="stylesheet" href="calculator.css">
</head>
<body>

    <div class="calculator">
        <input type="text" id="display" disabled/>
        <div class="buttons">
            <button onclick="clearDisplay()" class="clear">C</button>
            <button onclick="deleteLast()">DEL</button>
            <button onclick="appendCharacter('/')">/</button>
            <button onclick="appendCharacter('7')">7</button>
            <button onclick="appendCharacter('8')">8</button>
            <button onclick="appendCharacter('9')">9</button>
            <button onclick="appendCharacter('*')">*</button>
            <button onclick="appendCharacter('4')">4</button>
            <button onclick="appendCharacter('5')">5</button>
            <button onclick="appendCharacter('6')">6</button>
            <button onclick="appendCharacter('-')">-</button>
            <button onclick="appendCharacter('1')">1</button>
            <button onclick="appendCharacter('2')">2</button>
            <button onclick="appendCharacter('3')">3</button>
            <button onclick="appendCharacter('+')">+</button>
            <button onclick="appendCharacter('0')">0</button>
            <button onclick="appendCharacter('.')">.</button>
            <button onclick="calculateResult()" class="equals">=</button>
        </div>
    </div>
    <script src="calculator.js"></script>
</body>
</html>

```

CSS

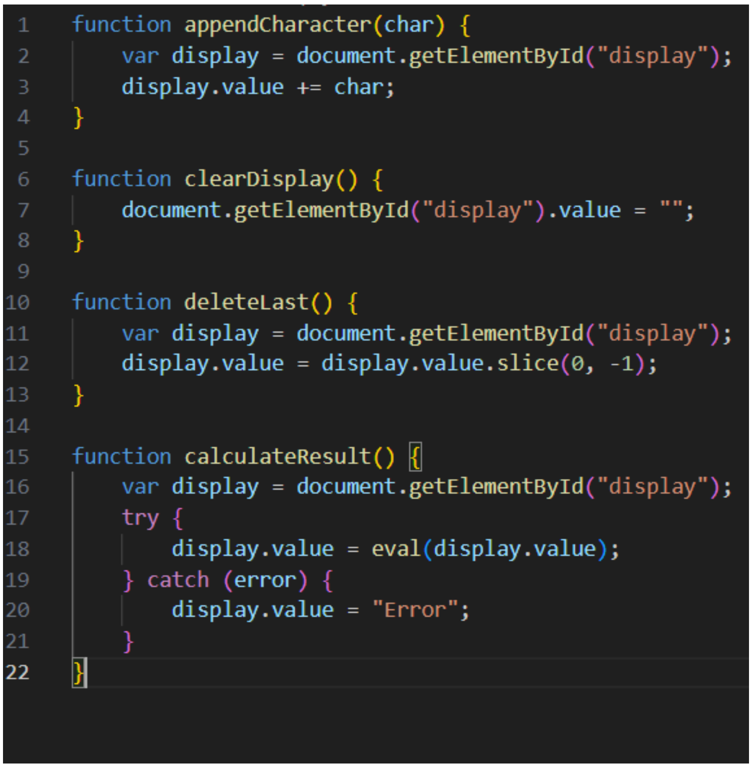


```css
body {
    font-family: Arial, sans-serif;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    margin: 0;
    background-color: #f4f4f4;
}
.calculator {
    width: 260px;
    padding: 20px;
    background-color: white;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}
.calculator input {
    width: 100%;
    height: 50px;
    font-size: 24px;
    text-align: right;
    margin-bottom: 10px;
    padding: 10px;
    border-radius: 5px;
    border: 1px solid #ddd;
    box-sizing: border-box;
}
.buttons {
    display: grid;
    grid-template-columns: repeat(4, 1fr);
    grid-gap: 10px;
}
.buttons button {
    width: 100%;
    padding: 20px;
    font-size: 18px;
    background-color: #f4f4f4;
```



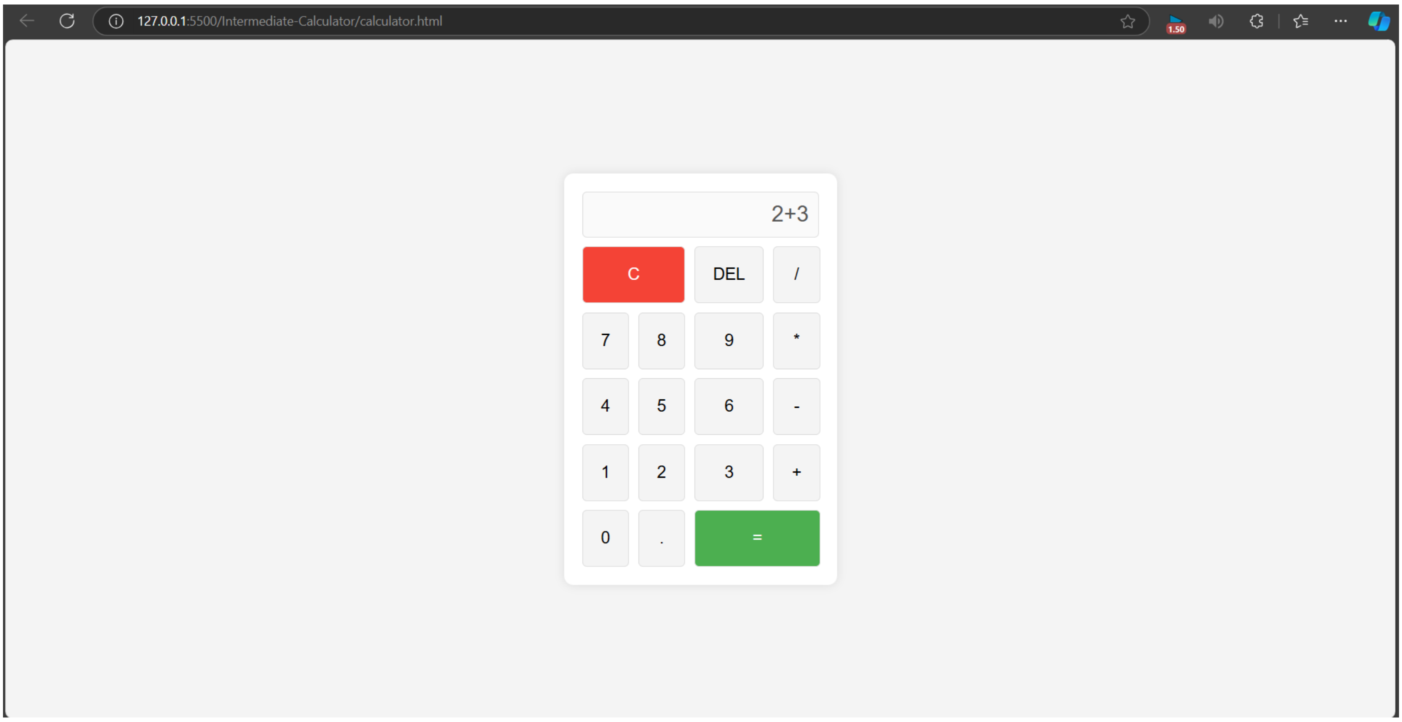
```css
    border: 1px solid #ddd;
    border-radius: 5px;
    cursor: pointer;
}
.buttons button:hover {
    background-color: #ddd;
}
.buttons button.clear {
    grid-column: span 2;
    background-color: #f44336;
    color: white;
}
.buttons button.equals {
    grid-column: span 2;
    background-color: #4CAF50;
    color: white;
}
```

JAVASCRIPT



```
function appendCharacter(char) {
    var display = document.getElementById("display");
    display.value += char;
}

function clearDisplay() {
    document.getElementById("display").value = "";
}

function deleteLast() {
    var display = document.getElementById("display");
    display.value = display.value.slice(0, -1);
}

function calculateResult() {
    var display = document.getElementById("display");
    try {
        display.value = eval(display.value);
    } catch (error) {
        display.value = "Error";
    }
}
```

Output :-



Features

- **Basic Arithmetic Operations**: Performs addition, subtraction, multiplication, and division.
- **Clear Button**: Resets the calculator display for new calculations.

- **Real-Time Dynamic Display**: Updates the display as the user inputs numbers and selects operations.

Challenges Addressed

- **Input Management**: Prevented errors from chaining operations (e.g., avoiding consecutive operations like "++" or "--").
- **Responsive Design**: Ensured UI consistency and usability across multiple screen sizes and devices.

Future Enhancements

- **Advanced Functionalities**: Add features like square root, percentage, and memory functions to expand calculator capabilities.
- **UI Customization**: Introduce customizable themes and display options to enhance user experience.

Conclusion

This project successfully implements a functional and user-friendly calculator with the foundation for additional features. Its clean design and responsive interface make it a practical tool for basic and advanced calculations, providing an excellent platform for future development.