**WRITE A PROGRAM USING ANY PROGRAMMING LANGUAGE OF YOUR CHOICE TO DEMONSTRATE THE USE OF WELL FULLY DEFINED CALCULATOR**

**SUBMITTED BY:**

**GROUP F**

**ABDULAZEEZ ABDULLATEEF O. (2021005314)**

**ADESALA JOHN OBALOLUWA (203221)**

**ADESOJI WILLIAMS OLAMIDE (203914)**

**AJANI GABRIEL IYANUOLUWA (203345)**

**AKANNI FIKAYO ALAGBE (2021007227)**

**IGE PETER ADEYEMI (201942)**

**ODEKUNLE OLALEKAN ABDULRASHEED (204032)**

**OYEDIRAN OLUWATOBILOBA DANIEL (203305)**

**OYEGBAMI SAMUEL (202689)**

**PETER OLORUNFEMI JOSEPH (203143)**

**TELLA OYINKANSOLA (200794)**

**SUBMMITTED TO**

**FACULTY OF COMPUTING AND INFORMATICS (FCI)**

**THE DEPARTMENT OF COMPUTER SCIENCE**

**SUPERVISOR:**

**PROF ISMAILA**

**JUNE, 2024.CREATING A SCIENTIFIC CALCULATOR**

Developing a web-based scientific calculator application, involves several steps. Here is the guide used developing a web-based scientific calculator that includes basic arithmetic operations, trigonometric functions, square roots, logarithms (base 10 and natural log), exponentiation, and factorials using HTML, CSS, and JavaScript.**.**

**STEP-BY-STEP CREATION OF AN ENHANCED WEB-BASED SCIENTIFIC CALCULATOR**

1 Define Requirements and Features

* Include basic arithmetic operations, trigonometric functions (sine, cosine, tangent), square root, logarithms (log and ln), exponentiation, and factorials.

2. Design the User Interface

* Plan the layout to include buttons for the new scientific functions.
* Design the display area to show the current input and result.

3. Set-Up the Development Environment

* Use a code editor like Visual Studio Code, Sublime Text, or any other preferred editor.
* Create a new project folder and add HTML, CSS, and JavaScript files.

4. Implement HTML Structure

* Create the HTML structure for the calculator, including buttons and display areas.

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Web-Based Scientific Calculator</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<div class="calculator">

<h2>GROUP F ( Scientific Calculator) </h2>

<div class="display" id="display"></div>

<div class="buttons">

<button class="btn" onclick="clearDisplay()">C</button>

<button class="btn" onclick="deleteLast()">DEL</button>

<button class="btn" onclick="appendOperator('/')">/</button>

<button class="btn" onclick="appendOperator('\*')">\*</button>

<button class="btn" onclick="appendNumber(7)">7</button>

<button class="btn" onclick="appendNumber(8)">8</button>

<button class="btn" onclick="appendNumber(9)">9</button>

<button class="btn" onclick="appendOperator('-')">-</button>

<button class="btn" onclick="appendNumber(4)">4</button>

<button class="btn" onclick="appendNumber(5)">5</button>

<button class="btn" onclick="appendNumber(6)">6</button>

<button class="btn" onclick="appendOperator('+')">+</button>

<button class="btn" onclick="appendNumber(1)">1</button>

<button class="btn" onclick="appendNumber(2)">2</button>

<button class="btn" onclick="appendNumber(3)">3</button>

<button class="btn equal" onclick="calculateResult()">=</button>

<button class="btn" onclick="appendNumber(0)">0</button>

<button class="btn" onclick="appendDot()">.</button>

<button class="btn" onclick="appendFunction('Math.sqrt')">√</button>

<button class="btn" onclick="appendFunction('Math.sin')">sin</button>

<button class="btn" onclick="appendFunction('Math.cos')">cos</button>

<button class="btn" onclick="appendFunction('Math.tan')">tan</button>

<button class="btn" onclick="appendFunction('Math.log10')">log</button>

<button class="btn" onclick="appendFunction('Math.log')">ln</button>

<button class="btn" onclick="appendOperator('^')">^</button>

<button class="btn" onclick="calculateFactorial()">!</button>

</div>

</div>

<script src="script.js"></script>

</body>

</html>

5. Style the Calculator with CSS

* Add styles to make the calculator visually appealing and user-friendly.

css

/\* styles.css \*/

body {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background-color: #f0f0f0;

margin: 0;

font-family: Arial, sans-serif;

}

.calculator {

border: 2px solid #333;

border-radius: 10px;

padding: 20px;

background-color: #fff;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

.display {

font-size: 2em;

margin-bottom: 10px;

padding: 10px;

border: 1px solid #ccc;

text-align: right;

background-color: #e9e9e9;

border-radius: 5px;

min-height: 40px;

}

.buttons {

display: grid;

grid-template-columns: repeat(4, 1fr);

gap: 10px;

}

.btn {

font-size: 1.5em;

padding: 15px;

border: none;

border-radius: 5px;

background-color: #f1f1f1;

cursor: pointer;

transition: background-color 0.2s;

}

.btn:hover {

background-color: #ddd;

}

.equal {

grid-column: span 2;

background-color: #ff9500;

color: #fff;

}

.equal:hover {

background-color: #e08500;

}

6. Implement JavaScript Functionality

* Write JavaScript functions to handle button clicks, perform calculations, and update the display.

javascript

// script.js

let display = document.getElementById('display');

let currentInput = '';

let resultDisplayed = false;

function appendNumber(number) {

if (resultDisplayed) {

currentInput = '';

resultDisplayed = false;

}

currentInput += number;

updateDisplay();

}

function appendOperator(operator) {

if (resultDisplayed) {

resultDisplayed = false;

}

if (operator === '^') {

currentInput += '\*\*';

} else {

currentInput += ` ${operator} `;

}

updateDisplay();

}

function appendFunction(func) {

if (resultDisplayed) {

currentInput = '';

resultDisplayed = false;

}

currentInput += ` ${func}(`;

updateDisplay();

}

function appendDot() {

if (resultDisplayed) {

currentInput = '';

resultDisplayed = false;

}

currentInput += '.';

updateDisplay();

}

function clearDisplay() {

currentInput = '';

updateDisplay();

}

function deleteLast() {

currentInput = currentInput.trim().slice(0, -1);

updateDisplay();

}

function calculateResult() {

try {

currentInput = currentInput.replace(/√/g, 'Math.sqrt');

let result = eval(currentInput);

currentInput = result.toString();

resultDisplayed = true;

updateDisplay();

} catch (e) {

currentInput = 'Error';

updateDisplay();

}

}

function calculateFactorial() {

if (resultDisplayed) {

resultDisplayed = false;

}

let num = parseInt(currentInput);

if (isNaN(num)) {

currentInput = 'Error';

} else {

currentInput = factorial(num).toString();

}

resultDisplayed = true;

updateDisplay();

}

function factorial(n) {

if (n === 0 || n === 1) return 1;

return n \* factorial(n - 1);

}

function updateDisplay() {

display.innerText = currentInput;

}

7. Test and Debug

* Open the HTML file in a web browser and test all functionalities to ensure they work as expected.
* Debug any issues such as incorrect calculations or input handling errors.

8. Optimize and Enhance

* Optimize the code for better performance and responsiveness.
* Consider adding additional features like parentheses for grouping operations if desired.

9. Documentation and User Guide

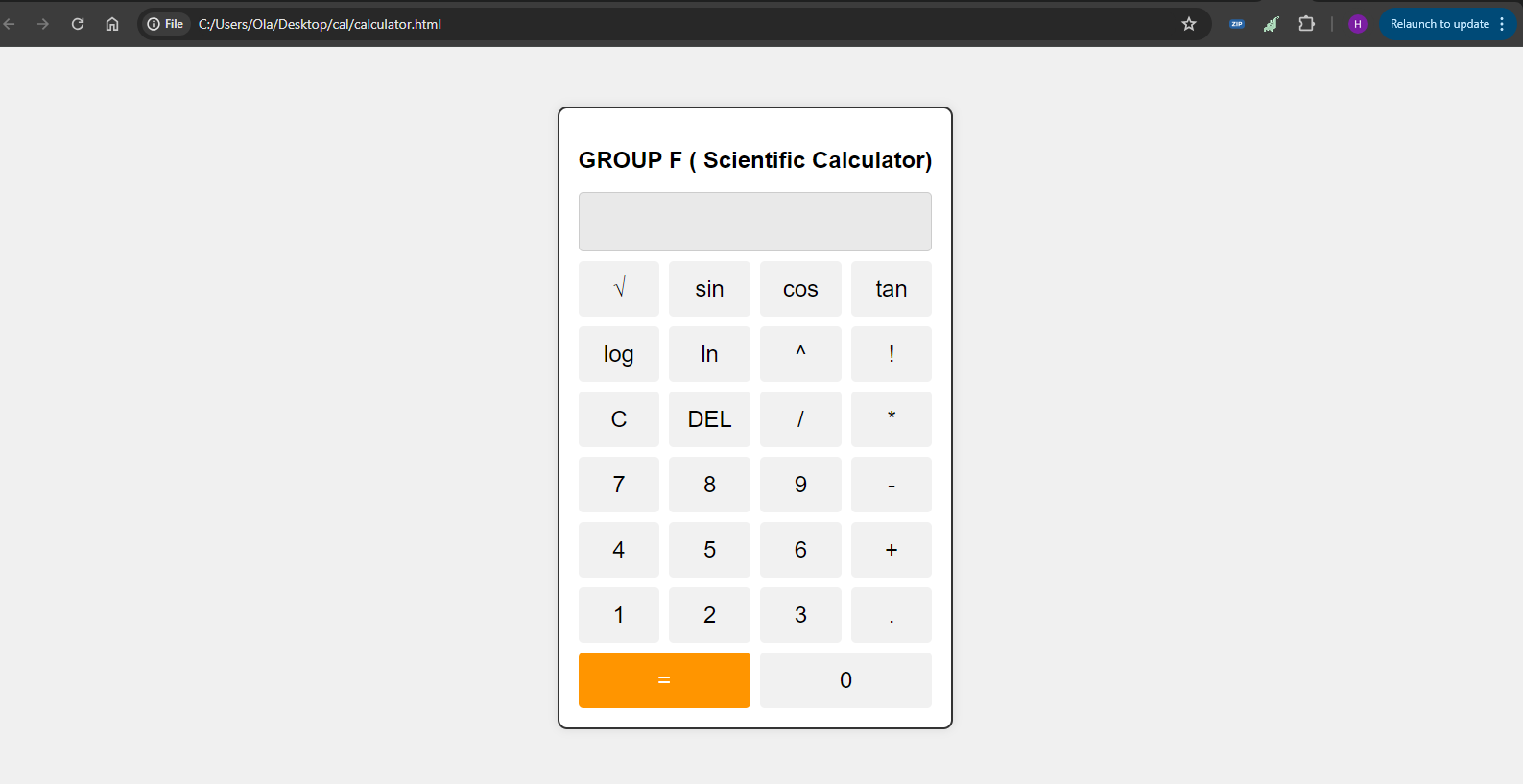
* Provide documentation on how to use the calculator.
* Include a user guide to explain each function and how to perform calculations.

10. Deployment

* Host the web-based calculator on a web server or use services like GitHub Pages for deployment.
* Share the link with users to access the calculator online.

By following these steps, you can create a functional and user-friendly web-based scientific calculator with basic arithmetic operations, trigonometric functions, square roots, logarithms, exponentiation, and factorials using HTML, CSS, and JavaScript.

**GRAPHIC USER INTERFACE**



ARTHIMETRIC OPERATIONS  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
OUTPUT  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
SQUARE OPERATION  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
TRIGONOMETRIC FUNCTIONS

