| **Experiment No. – 2** | | | | |
| --- | --- | --- | --- | --- |
| **Date of Performance:** | **15/1/2025** | | | |
| **Date of Submission:** | **22/1/2025** | | | |
| Program Execution/  formation/  correction/  ethical practices  (06) | Timely  Submission  (02) | Viva  (03) | Experiment  Total (20) | Sign with Date |
|  |  |  |  |  |

**Experiment No. 2**

**Calculator Web application using TypeScript and HTML.**

**2.1 Aim:** To design and develop a simple calculator web application using TypeScript and HTML that performs basic arithmetic operations like addition, subtraction, multiplication, and division.

**2.2 Course Outcome: 3.ITLR0603.2** Understand how TypeScript and AngularJS framework can build dynamic, responsive single-page web applications

**2.3 Learning Objectives:** To learn the basics of building interactive web applications using TypeScript and HTML.

**2.4 Requirement:**

* Visual Studio Code or any code editor
* Web browser (Chrome/Firefox)
* TypeScript compiler (tsc)
* Node.js installed (optional for compilation)

2.5 Related Theory:

**TypeScript** is an open-source programming language developed and maintained by Microsoft. It is a strict syntactical superset of JavaScript, which means any valid JavaScript code is also valid TypeScript. The key advantage of TypeScript is that it introduces **static typing** to JavaScript, allowing developers to define data types for variables, function parameters, and return values. This helps catch errors during development rather than at runtime, making the code more robust, maintainable, and easier to debug.

Key features of TypeScript include:

* **Type Annotations**: Define the type of variables, parameters, and return values.
* **Interfaces and Classes**: Help create modular, object-oriented code.
* **Access Modifiers** (public, private, protected): Improve data encapsulation.
* **Compile-time Error Checking**: Allows early detection of bugs.
* **Modern JavaScript Support**: Includes support for ES6/ES7 features like async/await, arrow functions, and destructuring.

**HTML** (HyperText Markup Language) is the standard markup language used to create the structure and layout of web pages. HTML elements define the building blocks of web interfaces, including input fields, buttons, headings, and containers.

**TypeScript + HTML Integration**:  
 In modern web development, TypeScript is often used in combination with HTML and CSS to create interactive front-end applications. While HTML handles the layout, TypeScript provides the logic and behavior, especially useful for validating user inputs, handling events (e.g., button clicks), and manipulating the Document Object Model (DOM).

In this experiment, we are combining both technologies to build a **calculator web application**, where:

* The **HTML** provides input fields for two numbers and buttons to perform arithmetic operations like Add, Subtract, Multiply, and Divide.
* The **TypeScript** code contains the core logic to read user input, perform the selected operation, and display the result on the screen dynamically.

The TypeScript code is compiled into JavaScript using the tsc (TypeScript Compiler), and the resulting JavaScript file is then referenced in the HTML to bring the logic into action on the web page.

This experiment illustrates how even a simple application can benefit from the clarity and reliability TypeScript offers over plain JavaScript, especially in handling user-driven actions in real-time.

**2.6 Procedure:**

1. Set up the HTML structure for the calculator with input fields and buttons.
2. Write TypeScript code to handle user input and perform calculations.
3. Compile the .ts file to .js using the TypeScript compiler.
4. Link the compiled JavaScript file in the HTML.
5. Open the HTML file in a browser to view and test the application.

**2.7 Program and Output:**

index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Inter:ital,opsz,wght@0,14..32,100..900;1,14..32,100..900&display=swap" rel="stylesheet">

<title>Calculator</title>

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

</head>

<body>

<div class="calc-body">

<div class="calc-screen">

<div id="calc-operation">0 </div>

<div id="calc-typed">0</div>

</div>

<div class="calc-button-row">

<button class="ac" onclick="clearAll()">AC</button>

<button class="opt">&#43;&#47;&#8722;</button>

<button class="opt">&#37;</button>

<button class="opt" onclick="getValue('/')">&#247;</button>

<button onclick="getValue('7')">7</button>

<button onclick="getValue('8')">8</button>

<button onclick="getValue('9')">9</button>

<button class="opt" onclick="getValue('\*')">&#215;</button>

<button onclick="getValue('4')">4</button>

<button onclick="getValue('5')">5</button>

<button onclick="getValue('6')">6</button>

<button class="opt" onclick="getValue('-')">&#8722;</button>

<button onclick="getValue('1')">1</button>

<button onclick="getValue('2')">2</button>

<button onclick="getValue('3')">3</button>

<button class="opt" onclick="getValue('+')">&#43;</button>

<button onclick="getValue('0')">0</button>

<button onclick="getValue('.')">.</button>

<button onclick="removeLastDigit()">&#9003;</button>

<button class="opt" onclick="solve()">&#61;</button>

</div>

</div>

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

</body>

</html>

calculator.css

\*{

font-family: 'Inter';

}

body {

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

margin: 0;

background: #2C3E50;

}

.calc-body {

width: 275px;

margin: auto;

min-height: 400px;

box-shadow: 0 8px 50px -7px black;

background: #3A4655;

}

.calc-screen {

padding: 1rem;

}

#calc-operation {

font-size: 1.3rem;

text-align: right;

color: #727B86;

padding-bottom: .5rem;

}

#calc-typed {

font-size: 2rem;

text-align: right;

color: #fff;

}

.calc-button-row{

display: table;

}

.calc-button-row button {

display: table-cell;

width: 25%;

background: #425062;

color: #fff;

height: 65px;

font-size: 1.3rem;

border: none;

border-color:#3C4857;

border-width: 1px 1px 0px 0;

border-style: solid;

}

.calc-button-row button.ac {

color: #ff7665;

}

.calc-button-row button.opt {

color: #ffbc56;

}

/\*The border has been managed in every fourth button so that the design will not distract.\*/

.calc-button-row button:nth-child(4n){

border-right: none;

}

.calc-button-row button:active {

position: relative;

top: 1px;

}

.calc-button-row button:hover {

background: #3e4b5c;

}

calculator.ts

const input\_val = document.getElementById('calc-operation') as HTMLDivElement;

const ans = document.getElementById('calc-typed') as HTMLDivElement;

function getValue(val: string) {

if (input\_val.innerText != '0')

input\_val.innerText += val;

else

input\_val.innerText = val;

}

function solve() {

const equation = input\_val.innerText;

try {

const result = eval(equation);

ans.innerText = result.toString();

} catch (e) {

input\_val.innerText = "Error";

}

}

function clearAll() {

input\_val.innerText = "0";

ans.innerText = "0";

}

function removeLastDigit() {

if (input\_val.innerText.length > 1)

input\_val.innerText = input\_val.innerText.slice(0, -1);

else {

input\_val.innerText = "0";

}

}

calculator.js (Compiled from calculator.ts)

var input\_val = document.getElementById('calc-operation');

var ans = document.getElementById('calc-typed');

function getValue(val) {

if (input\_val.innerText != '0')

input\_val.innerText += val;

else

input\_val.innerText = val;

}

function solve() {

var equation = input\_val.innerText;

try {

var result = eval(equation);

ans.innerText = result.toString();

}

catch (e) {

input\_val.innerText = "Error";

}

}

function clearAll() {

input\_val.innerText = "0";

ans.innerText = "0";

}

function removeLastDigit() {

if (input\_val.innerText.length > 1)

input\_val.innerText = input\_val.innerText.slice(0, -1);

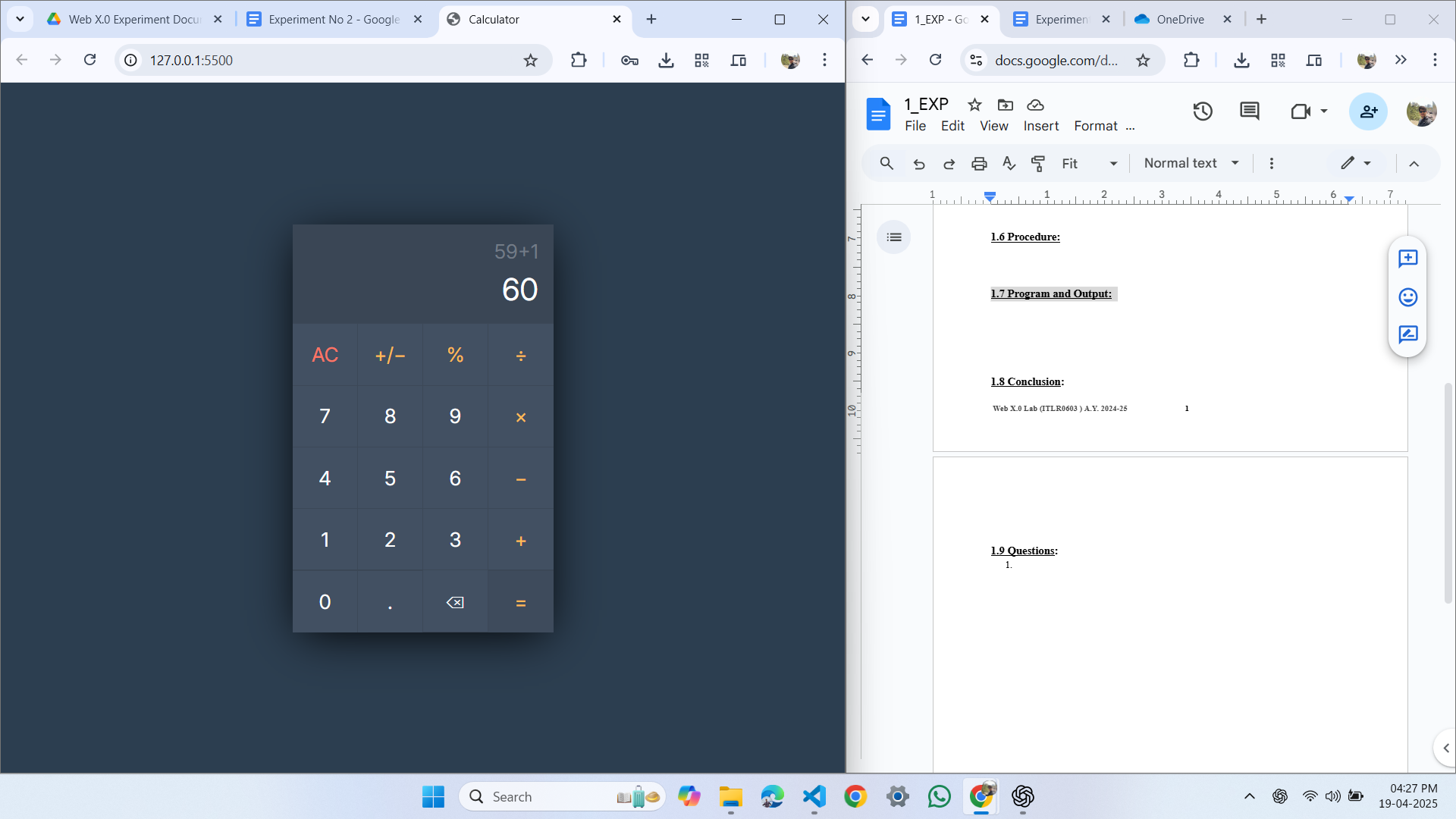
else {

input\_val.innerText = "0";

}

}

**Output:**

****

**2.8 Conclusion:**: The experiment successfully demonstrated the use of TypeScript and HTML to design a basic web application. It helped in understanding the integration of logic (TypeScript) with user interface elements (HTML) and how to compile and link TypeScript into functional JavaScript.

**2.9 Questions:**

1. **What is TypeScript?**

TypeScript is a typed superset of JavaScript that compiles to plain JavaScript for better error handling and code maintainability.

1. **What are the steps to compile and run a TypeScript program?**

Write your TypeScript code in a .ts file, compile it using tsc filename.ts to generate a .js file, then include that .js file in your HTML using <script src="filename.js"></script> to run it in the browser.

1. **What are the advantages of using TypeScript in web development, and how is it integrated into HTML?**

TypeScript improves readability, offers type safety with early error detection, and is integrated into HTML by compiling .ts files to .js and linking them using the <script> tag.