

Experiment-04

- **Aim :** Implement an application in javascript using following -
 - (a) Design UI of application using HTML, CSS, etc.
 - (b). Include Javascript validation.
 - (c) Use of prompt and alert window using javascript.

eg - Design and implement a simple calculator using JS. for operations like addition, multiplication, subtraction, division, square of a number, etc.

- a) Interface like text field for input, output, buttons for numbers and operators, etc.
- b) Validate input values.
- c) Prompts/alerts for invalid values, etc.

Theory :

- A calculator application is a fundamental project that demonstrates the use of HTML, CSS and JS. in web development.
- This application enables users to perform basic arithmetic operations including addition, multiplication, etc.

HTML : Structures the calculators interface.

CSS : Enhances visual appeal and user experience.

JS : Adds interactivity, validation, dynamic calculations.

I] UI design and Interface using HTML, CSS -

HTML for calculator structure.

- Input field - Displays user input and results.
- Buttons - Represent digits (0-9) operators (+, -, ×, ÷) and functions like clear (C), equal (=).

Example syntax -

```
<input type = "text" id = "display" readonly >
<button onclick = "appendNumber(1)" > 1 </button>
<button onclick = "appendOperator('+')" > + </button>
```

II] CSS for styling -

CSS improves the visual layout, making the calculator user friendly. It controls -

- Button size and alignment for better usability.
- Colors and backgrounds for a modern look.
- Input field styling - to make output readable.

CSS -

```
# display {
  width : 100%;
  font-size : 20px;
  text-align : right;
  padding : 5px;
}
```



```
button {  
    width : 50 px ;  
    height : 50 px ;  
    font-size : 18 px ;  
    margin : 5 px ;  
}
```

III] Javascript Implementation -

- JS is responsible for processing input, performing calculation and handling errors.

(a) Handling user input and operations:-

Javascript functions capture user input and perform calculations.

Example function to handle button clicks -

```
function appendNumber(num) {  
    document.getElementById("display").value += num;  
}
```

(b) Performing calculations -

Javascript executes calculation when user clicks the "=" button.

```
function calculateResult () {  
    let expression = document.getElementById("display").value  
    document.getElementById("display").value = eval(expression)  
}
```

`eval()` is used for simplicity but should be replaced with a safer alternative in production code.

(c) Input validation - It ensures users enter correct values and prevents invalid operations like division by zero.

```
function validateInput(value) {
    if (isNaN(value) || value === " ") {
        alert("Invalid input!");
        return false;
    }
    return true;
}
```

(d). Using prompts and alerts for User Interaction.

- `alert()` - Displays error messages when invalid input is detected.
- `prompt()` - Accepts user input dynamically eg. for squaring a number.

```
function squareNumber() {
    let num = prompt("Enter a number to square:");
    if (validateInput(num)) {
        alert("The square is: " + (num * num));
    }
}
```

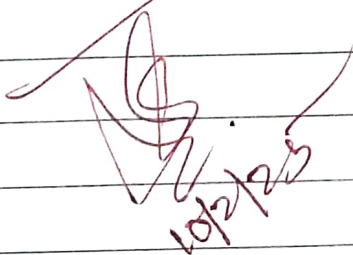
Flow is as follows -

IV) Application flow -

- (1). User enters numbers and operators using calculator buttons.
- (2). Javascript captures the input and updates the display field.
- (3). The equals (=) button triggers the calculation.
- (4). Javascript validates the input to prevent errors.
- (5). Results are displayed in the text field.
- (6). Alerts are shown if the user enters invalid values.
- (7). The clear button resets the calculator for new calculations.

- **Conclusion :** This JS calculator demonstrates how HTML, CSS and JS work together to create an interactive web application.
 - HTML provides the structure.
 - CSS enhances design.
 - JS handles functionality.

This helped us understand DOM manipulation, event handling, user input validation and interactive UI design. Thus we successfully created a JS application.


10/2/25

* * * * *