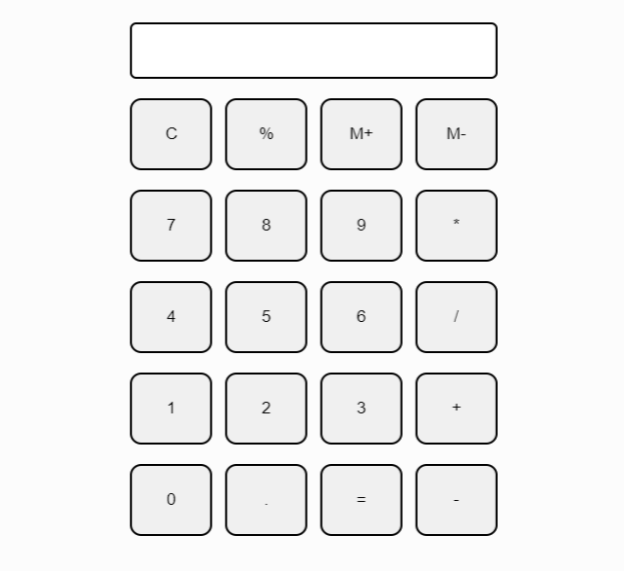
Hands – On Lab

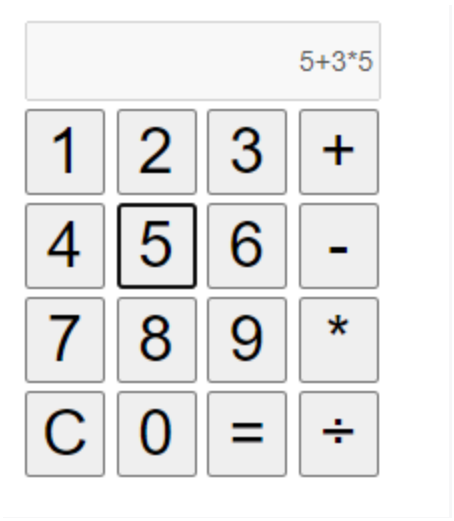
Workshop 4

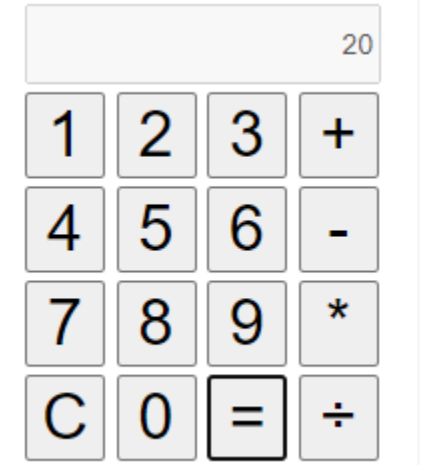
SIMPLE CALCULATOR

Create a UI of calculator using HTML and CSS and perform addition, subtraction, multiplication and division operations. Also handle the errors and exceptions. While clicking on C button, it should clear the textbox.



```javascript
const display = document.getElementById('display');
const button1 = document.getElementById('button1');
const button2 = document.getElementById('button2');
const button3 = document.getElementById('button3');
const button4 = document.getElementById('button4');
const button5 = document.getElementById('button5');
const button6 = document.getElementById('button6');
const button7 = document.getElementById('button7');
const button8 = document.getElementById('button8');
const button9 = document.getElementById('button9');
const button0 = document.getElementById('button0');

button1.addEventListener('click', () => display.value += '1');
button2.addEventListener('click', () => display.value += '2');
button3.addEventListener('click', () => display.value += '3');
button4.addEventListener('click', () => display.value += '4');
button5.addEventListener('click', () => display.value += '5');
button6.addEventListener('click', () => display.value += '6');
button7.addEventListener('click', () => display.value += '7');
button8.addEventListener('click', () => display.value += '8');
button9.addEventListener('click', () => display.value += '9');
button0.addEventListener('click', () => display.value += '0');

const addButton = document.getElementById('addButton');
const subtractButton = document.getElementById('subtractButton');
const multiplyButton = document.getElementById('multiplyButton');
const divideButton = document.getElementById('divideButton');
const equalsButton = document.getElementById('equalsButton');
const clearButton = document.getElementById('clearButton');


addButton.addEventListener('click', () => display.value += '+');
subtractButton.addEventListener('click', () => display.value += '-');
multiplyButton.addEventListener('click', () => display.value += '*');
divideButton.addEventListener('click', () => display.value += '/');

equalsButton.addEventListener('click', () => {

  try {

    display.value = eval(display.value);

  } catch (error) {

    display.value = 'Error';

  }

});



clearButton.addEventListener('click', () => {

  display.value = '';

});
```


5+3*5
1
2
3
+
4
5
6
-
7
8
9
*
C
0
=
÷


20
1
2
3
+
4
5
6
-
7
8
9
*
C
0
=
÷

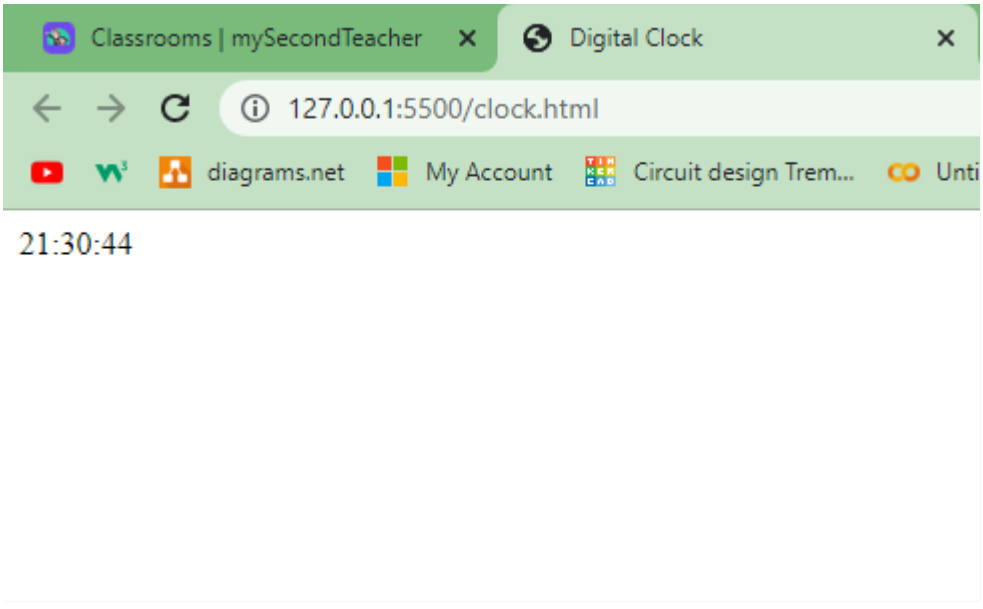
DIGITAL CLOCK

Create a Digital Clock using setInterval and Date function in JavaScript.

Note: Date object can be used to get the date, time, hours and seconds which can be updated using setInterval (). Try to keep the UI good looking and different from each other.

ZZ

```
function updateClock() {
    const clock = document.getElementById('clock');
    const now = new Date();
    const hours = now.getHours().toString().padStart(2, '0');
    const minutes = now.getMinutes().toString().padStart(2, '0');
    const seconds = now.getSeconds().toString().padStart(2, '0');
    const timeString = `${hours}:${minutes}:${seconds}`;
    clock.textContent = timeString;
  }

updateClock();
setInterval(updateClock, 1000);
```

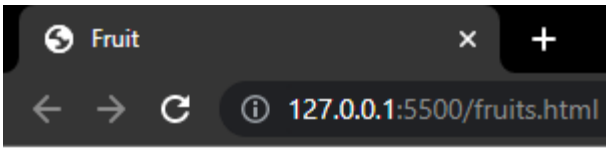


ASYNCHRONOUS JAVASCRIPT

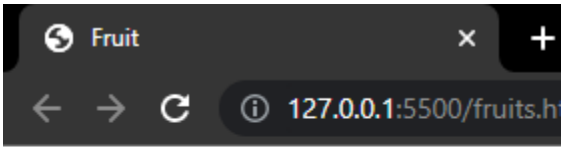
a. Create a function called **getFruit** that takes in a fruit name as a parameter and returns a Promise that resolves after 1 second with a message saying "Here is your [fruit]". If the fruit name is "watermelon", the Promise should reject after 2 seconds with an error message saying "Sorry, we're out of watermelons

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Fruit</title>
</head>
<body>
  <p id = "result"></p>
  <script src="fruit.js"></script>

</body>
</html>
```

```javascript
function getFruit(fruit){
    return new Promise((resolve, reject) =>{
    if(fruit != "watermelon"){
    setTimeout(() =>{
    resolve(`Here is your ${fruit}.`);
    }, 1000)
    }
    else{
    setTimeout(() =>{
    reject("Sorry we're out of watermelons.")
    }, 2000)
    }
    })
    }

    let fruit = prompt("Enter a fruit name: ");
    getFruit(fruit).then(value => {result.innerHTML = value}).catch(err => {result.innerHTML = err});
```



Sorry we're out of watermelons.



Here is your mango.

b. Create a function called **`arrayManipulation`** that takes in an array of numbers and two callback functions as parameters. The first callback function should perform an operation on each element in the array, and the second callback function should filter the resulting array based on a condition. The function should return the filtered array

```
<!DOCTYPE html>
<HTML>
<script>
    function arrayManipulation(arr, operationFn, filterFn) {

  const manipulatedArr = arr.map(operationFn);
  const filteredArr = manipulatedArr.filter(filterFn);
  return filteredArr;
    }

    const arr = [5, 3, 1, 4, 8];
    const multiplyByTwo = num => num * 3;
    const isGreaterThanFive = num => num > 5;
    const filteredArr = arrayManipulation(arr, multiplyByTwo, isGreaterThanFive);

document.write(filteredArr);
</script>
</HTML>

```

15,9,12,24

c. Create an asynchronous function called **`fetchUserData`** that uses **`async/await`** to fetch user data from a JSON API **(https://jsonplaceholder.typicode.com/users)**. The function should take in a user ID as a parameter, and use that ID to fetch the user's data from the API. If the API returns an error, the function should throw an error. If the API returns the user data, the function should return an object containing the user's name and email

d. Fetch data from API (https://jsonplaceholder.typicode.com/todos) and display (UserId,Title and Status) in a browser whose completed status is true and id<=50