**Name: Kunal Baghele**

**Asssignment -1**

**Task -1 (Button task)**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet"

integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL"

crossorigin="anonymous"></script>

<title>Button Task 1</title>

<link rel="stylesheet" href="../Style.css">

</head>

<body>

<div class="container text-center">

<div id="head">

<h1>Button Task 1</h1>

</div><br>

<div>

<strong>Click on buttons to change the colors</strong><br><br>

<button class="btn btn-light" value="red" onclick="changeColor(this)" >Red</button>

<button class="btn btn-light" value="green" onclick="changeColor(this)" >Green</button>

<button class="btn btn-light" value="yellow" onclick="changeColor(this)" >Yellow</button>

</div>

</div>

<script>

function changeColor(button) {

const color = button.value;

// console.log(color); //---> Color value from value attribute of button tag will come here

const buttons = document.getElementsByTagName("button");

// console.log(buttons); //---> It will return the HTMLCollection of buttons

//Iterating the collection to set background color of button

for (let i = 0; i < buttons.length; i++) {

buttons[i].style.backgroundColor = color; // ---> Assigning the color value to all buttons while iterating

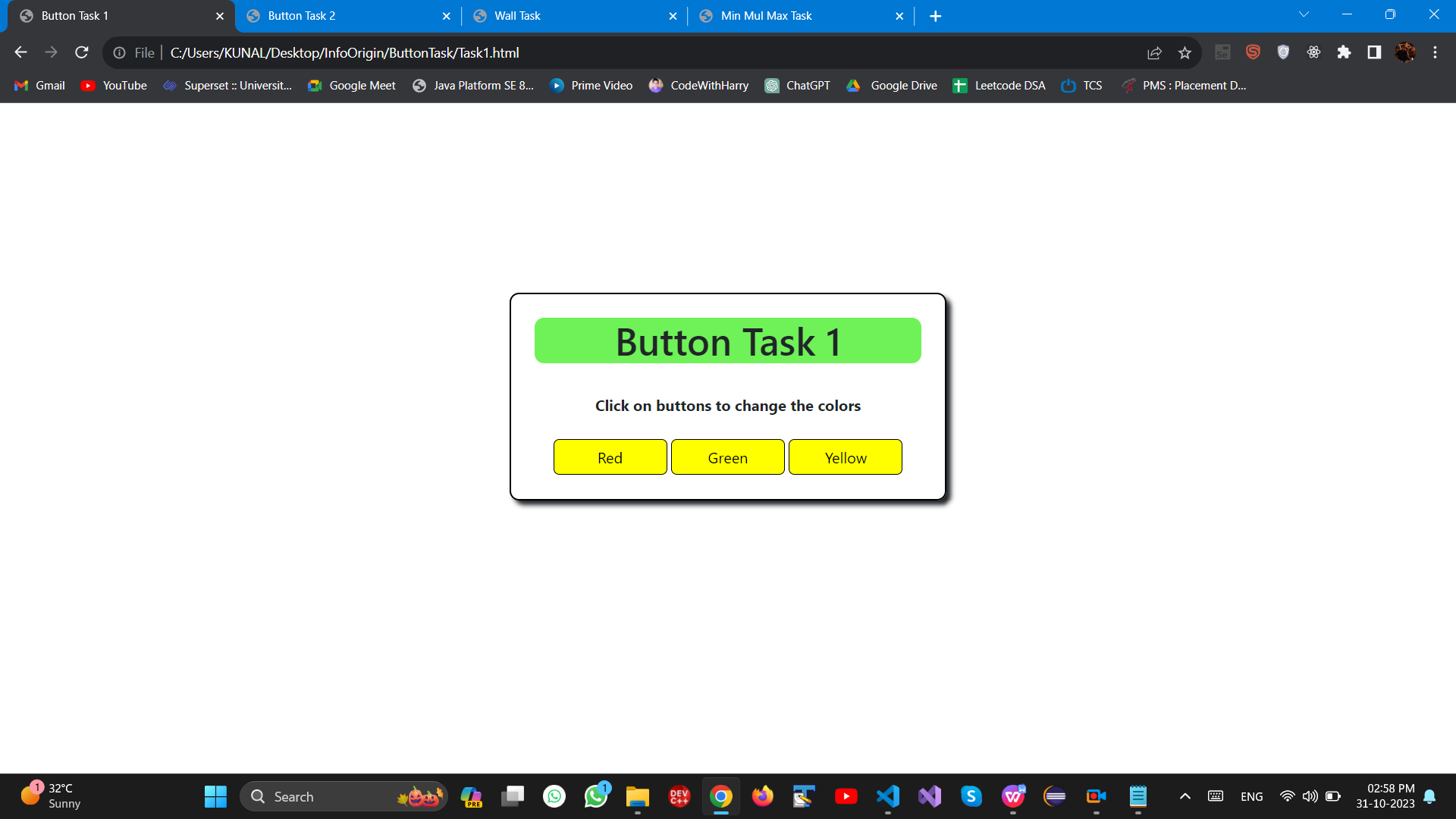
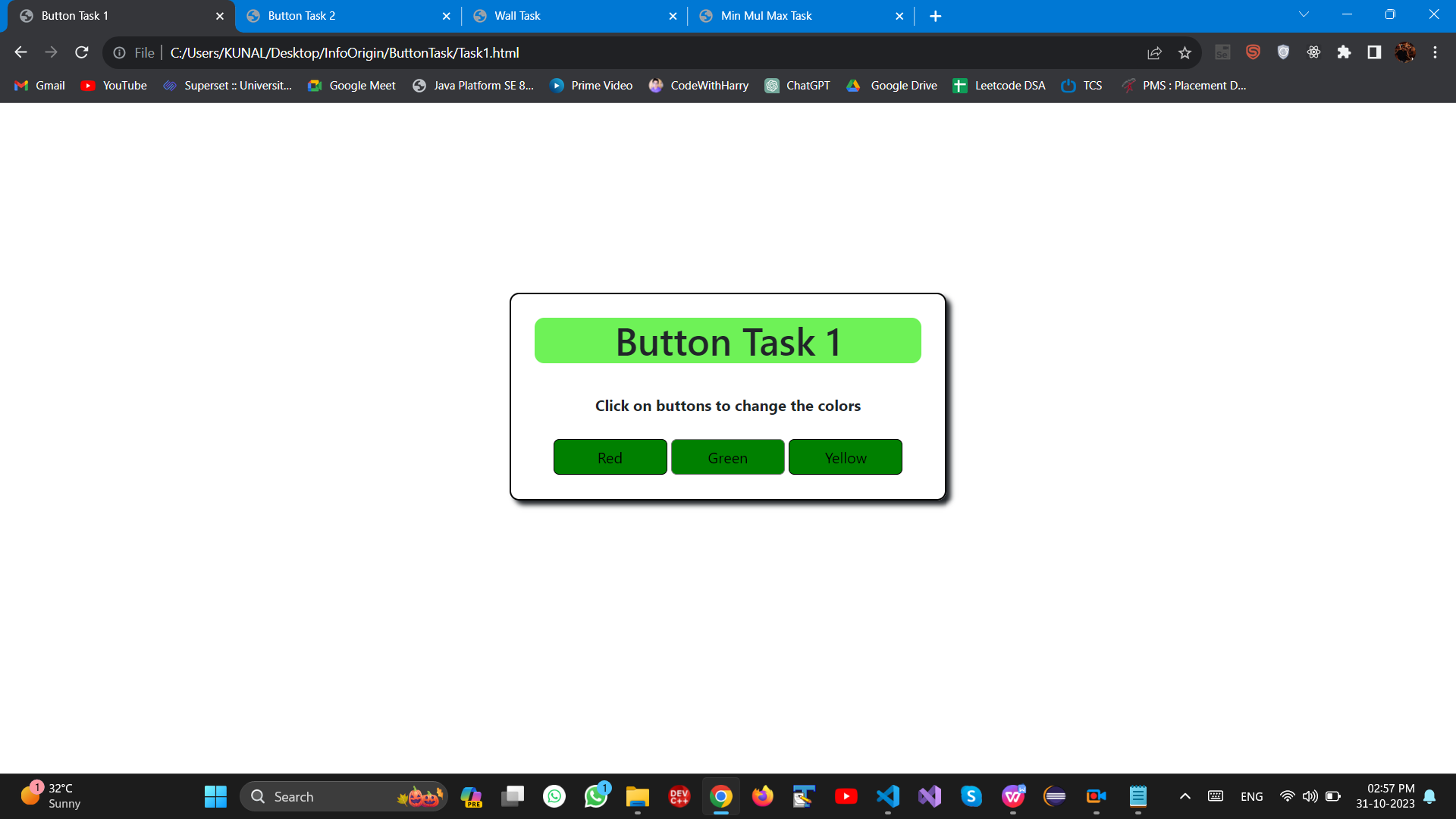
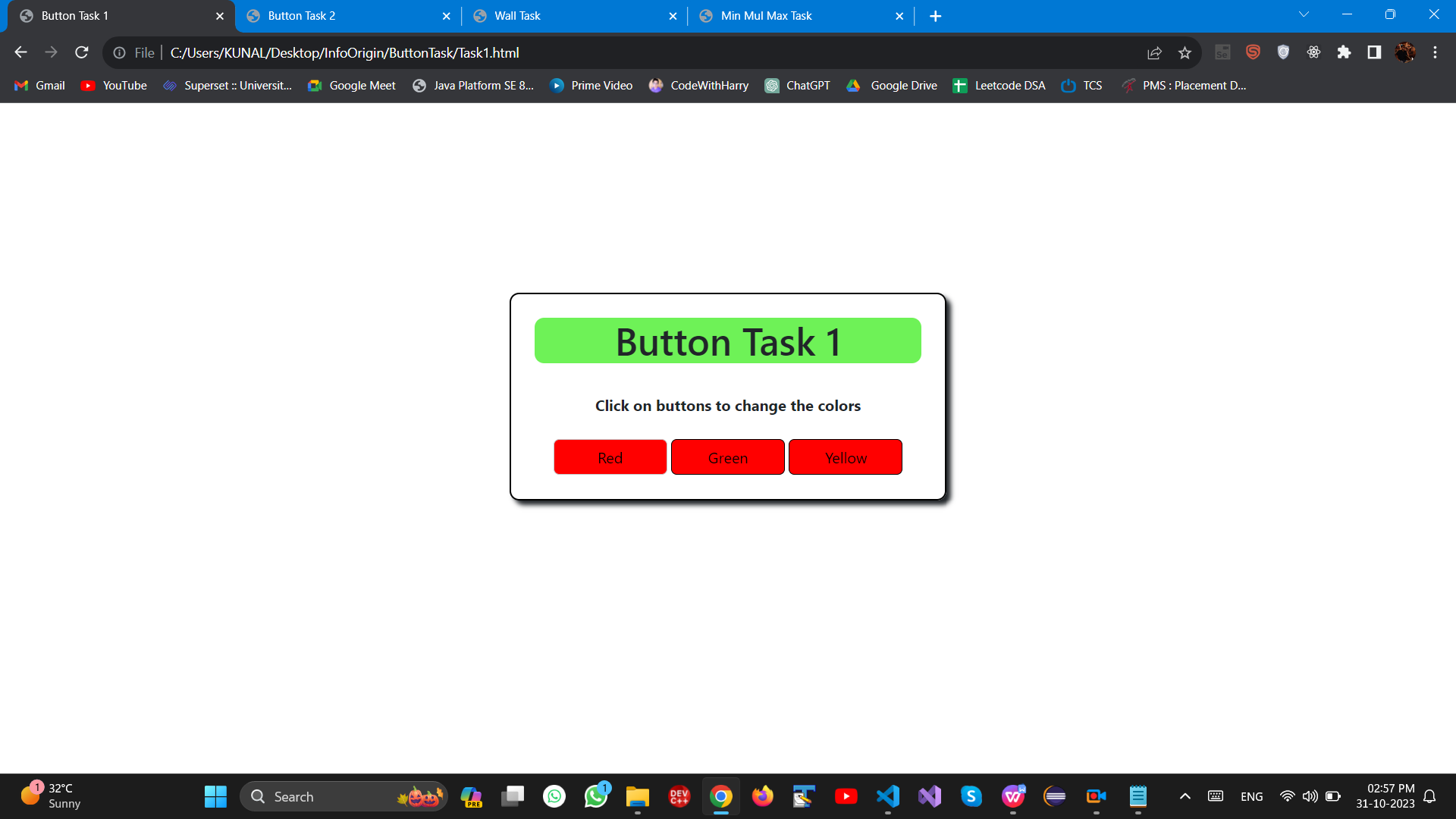
}

}

</script>

</body>

</html>



***--------------------------------------------------------------------------------------------------------------------------------------------------------------------------***

***--------------------------------------------------------------------------------------------------------------------------------------------------------------------------***

**Task -2 (Button task)**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet"

integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL"

crossorigin="anonymous"></script>

<title>Button Task 2</title>

<link rel="stylesheet" href="../Style.css">

</head>

<body>

<div class="container text-center">

<div id="head">

<h1>Button Task 2</h1>

</div><br>

<div>

<button class="btn" id="btn1" value="red" style="background-color: red;">Button 1</button>

<button class="btn" id="btn2" value="green" style="background-color: green;">Button 2</button>

<button class="btn" id="btn3" value="yellow" style="background-color: yellow;">Button 3</button><br><br>

<button class="btn" onclick="Shuffle()" style="background-color: darkgray;">Rotate</button><br>

<strong>Click on rotate button to shuffle the colors</strong>

</div>

</div>

<script>

function Shuffle() {

const btn1 = document.getElementById('btn1');

const btn2 = document.getElementById('btn2');

const btn3 = document.getElementById('btn3');

// Store the current background colors

const color1 = btn1.style.backgroundColor;

const color2 = btn2.style.backgroundColor;

const color3 = btn3.style.backgroundColor;

// Rotate the colors in a clockwise manner

btn1.style.backgroundColor = color3;

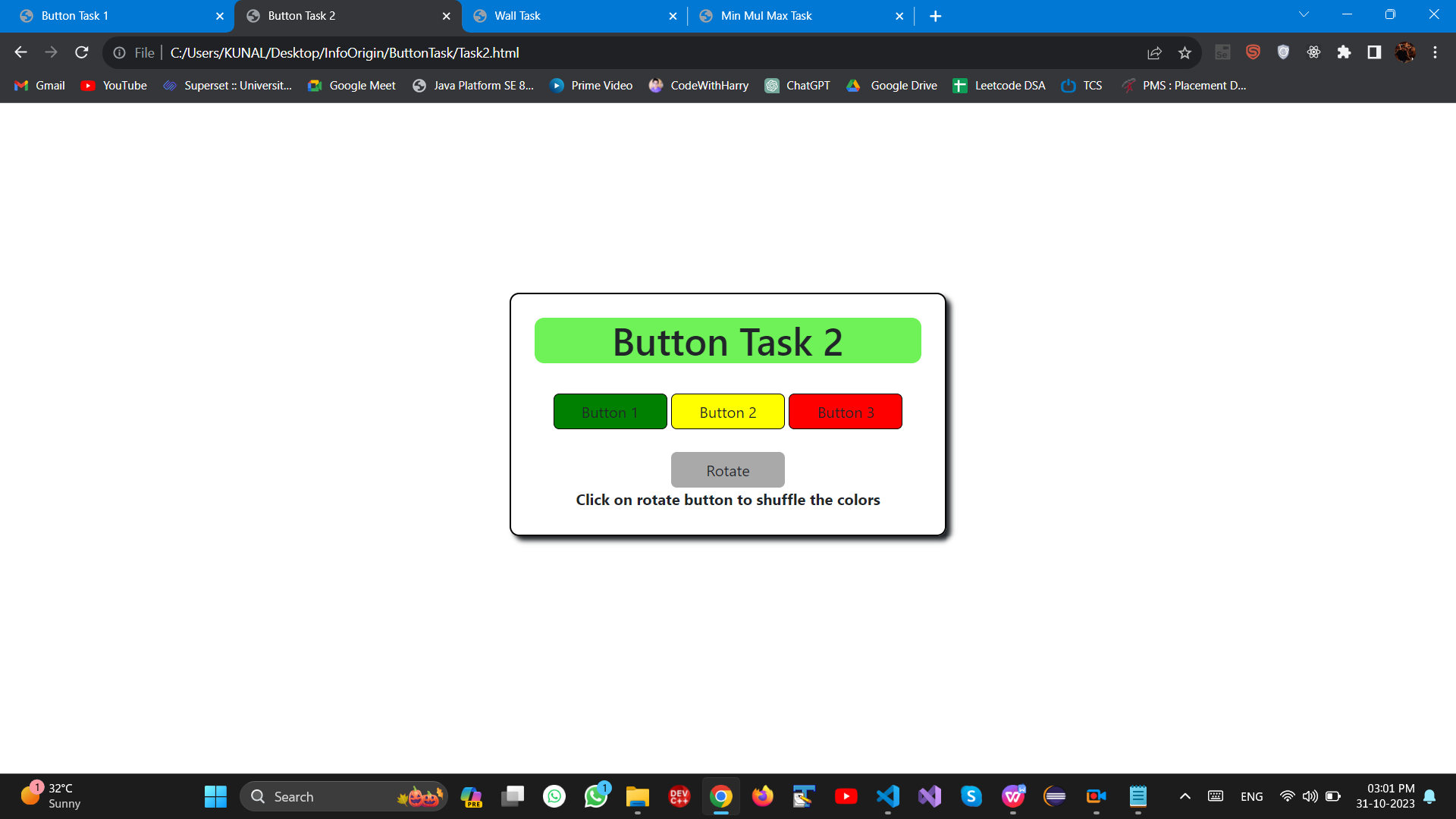
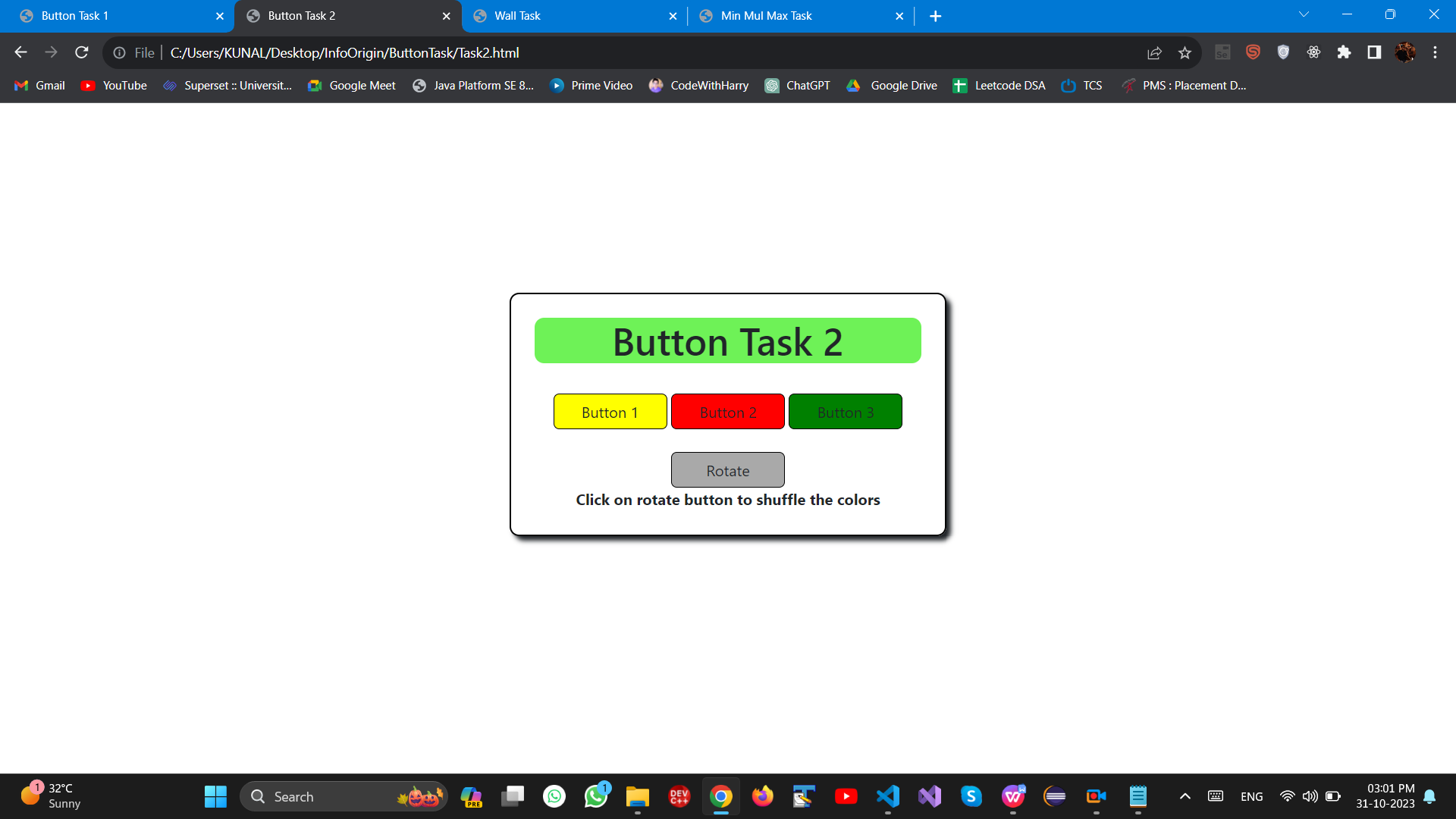
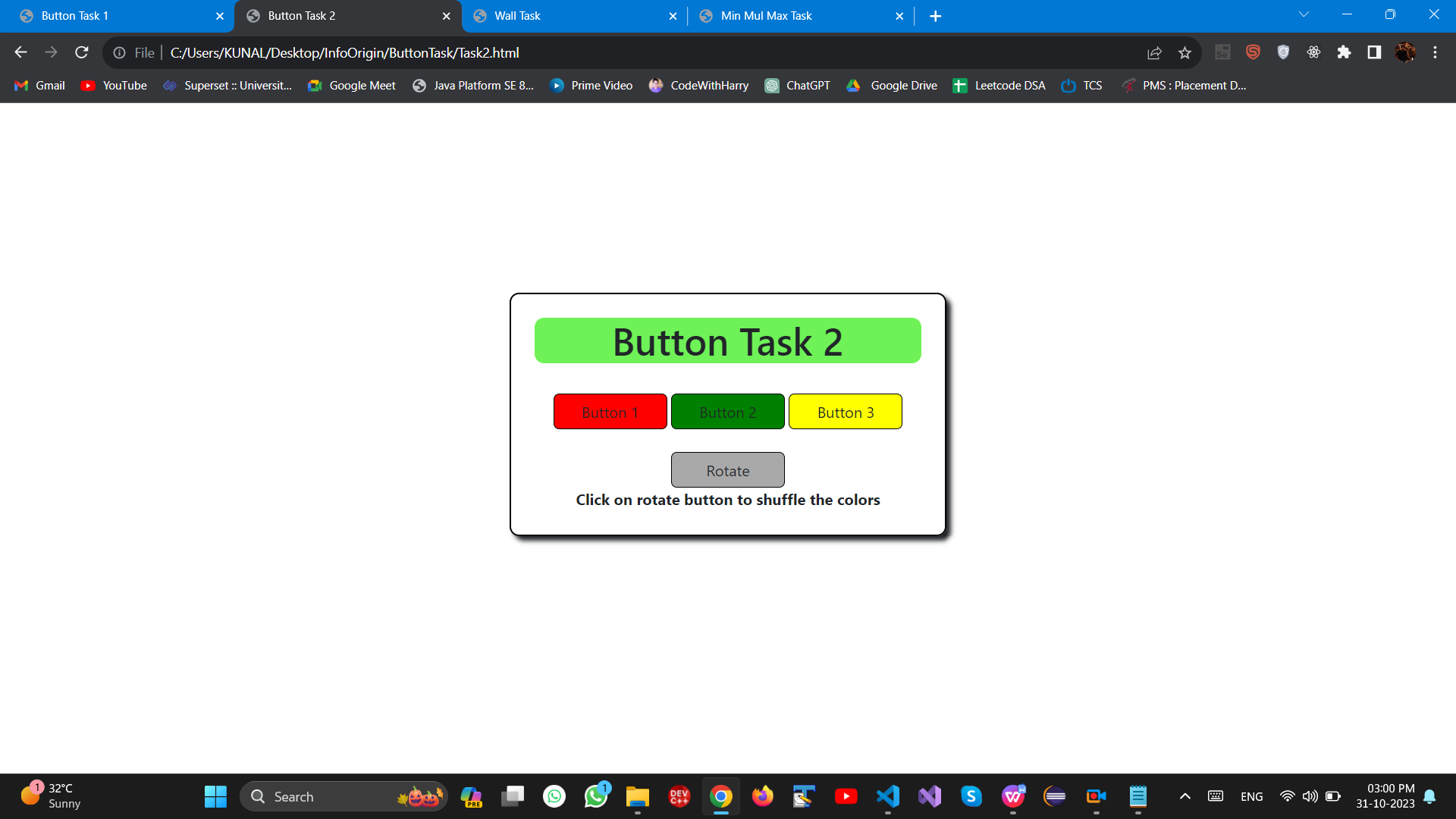
btn2.style.backgroundColor = color1;

btn3.style.backgroundColor = color2;

}

</script>

</body>

</html>

**--------------------------------------------------------------------------------------------------------------------------------------------------------------------------**

**--------------------------------------------------------------------------------------------------------------------------------------------------------------------------**

**Task-3 (Wall Task)**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet"

integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL"

crossorigin="anonymous"></script>

<title>Wall Task</title>

<link rel="stylesheet" href="../Style.css">

</head>

<body>

<div id="container1" class="container text-center">

<div id="head">

<h1>Wall Task</h1>

</div>

<form>

<div>

<label class="form-label">Number of walls</label>

<input type="text" class="form-control" id="textbox1">

</div>

<div>

<label class="form-label">Height of Walls (separated by '#')</label>

<input type="text" class="form-control" id="textbox2">

</div><br>

<button class="btn btn-primary" onclick="createGraph()">Create Graph</button>

</form>

</div>

<h3 id="left">Walls visible from person on Left<br><span id="lwall" class="badge bg-secondary">0</span></h3>

<div id="graphContainer" class="container text-center">

<div id="graph" class="barContainer"></div>

</div>

<h3 id="right">Walls visible from person on Right<br><span id="rwall" class="badge bg-secondary">0</span></h3>

<script>

function createGraph() {

var lcount = 1;

var rcount = 1;

event.preventDefault(); //Prevents page from reloading wen button is clicked

const walls = parseInt(document.getElementById("textbox1").value); // ---> Converting to integer...number of walls from user input

const heights = document.getElementById("textbox2").value.split("#").map(Number); // Seperating the values and storing in heights

// console.log(heights) // ---> We will get the collection

if (walls !== heights.length) {

alert("Number of walls and the number of heights should match.");

return;

}

// Declaring variables for comparison from left and right

var temp = heights[0]; // for left

var temp1 = heights[heights.length - 1]; // for right

for (let i = 0; i < heights.length; i++) {

if (heights[i] > temp) {

temp = heights[i]; // Reassigning the value to temp if the current bar height is greater for further comparison

lcount++;

//if (heights[i] == heights[i + 1]) {

// lcount--;

//}

}

}

document.getElementById("lwall").innerHTML = lcount; // It will update the no of walls visible from left , We are assigning value to inner html

for (let j = heights.length; j >= 0; j--) {

if (heights[j] > temp1) {

temp1 = heights[j];

rcount++;

//if (heights[j] == heights[j - 1]) {

// rcount--;

//}

}

}

document.getElementById("rwall").innerHTML = rcount; // It will update the no of walls visible from right , We are assigning value to inner html

const graphContainer = document.getElementById("graph");

graphContainer.innerHTML = "";

for (let i = 0; i < walls; i++) {

const height = heights[i] \* 30; // calculate height \*30 px to properly show on ui

// Create bars for each input from user

const bar = document.createElement("div"); //Creating html element div

bar.classList.add("bar"); // adding class property as "bar"

bar.style.height = height + "px"; // set height of bar using css

// Add a label to the bar

const labelValue = document.createElement("div"); //Creating html element div to display height of each bar

labelValue.classList.add("bar-label"); // adding class property as "bar-label"

labelValue.innerText = heights[i]; // Assigning height value to inner text of this div element

bar.appendChild(labelValue); //adding bar-label as child element inside bar

graphContainer.appendChild(bar); //adding bar(bar + bar-label) to graph container

}

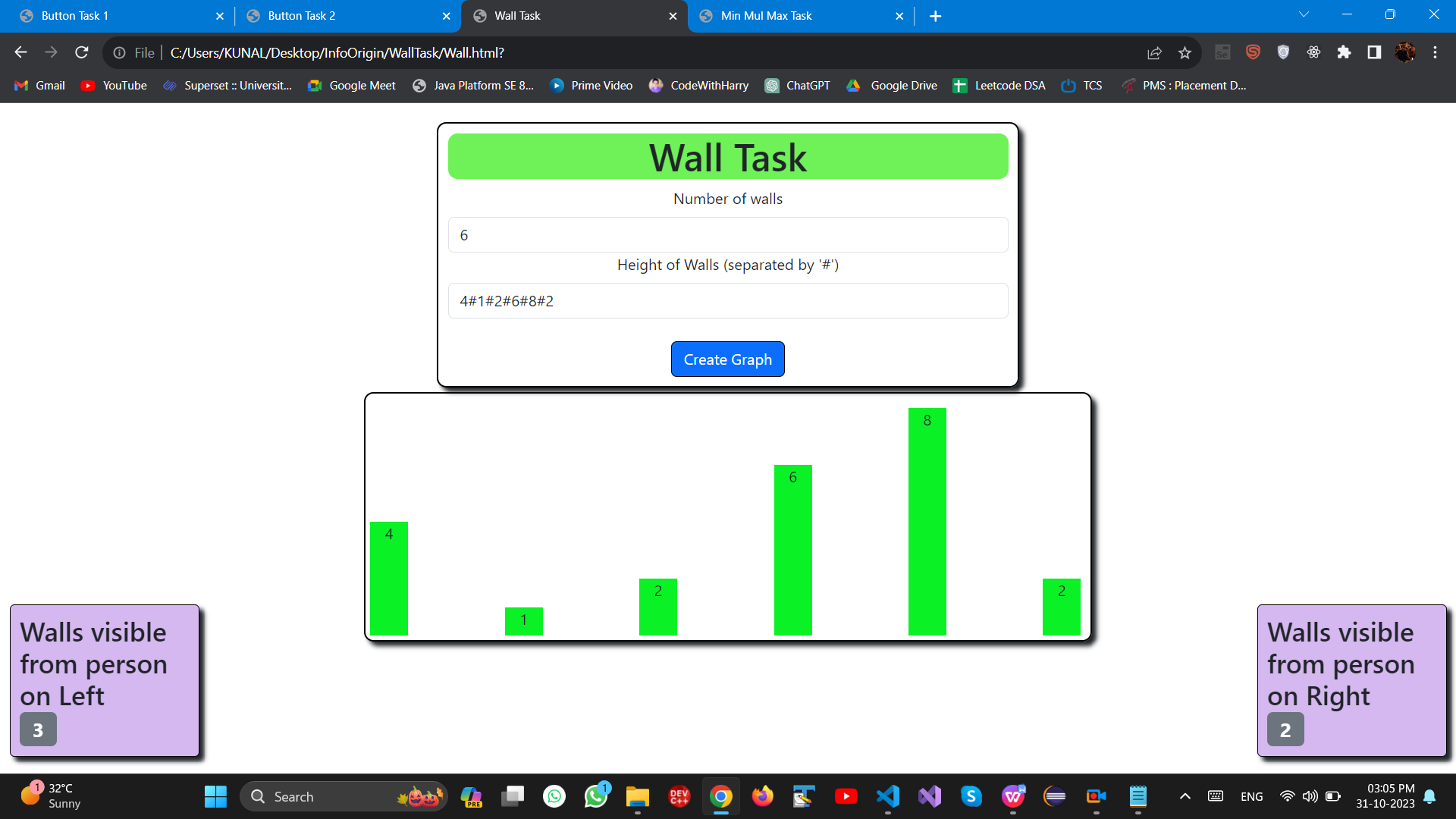
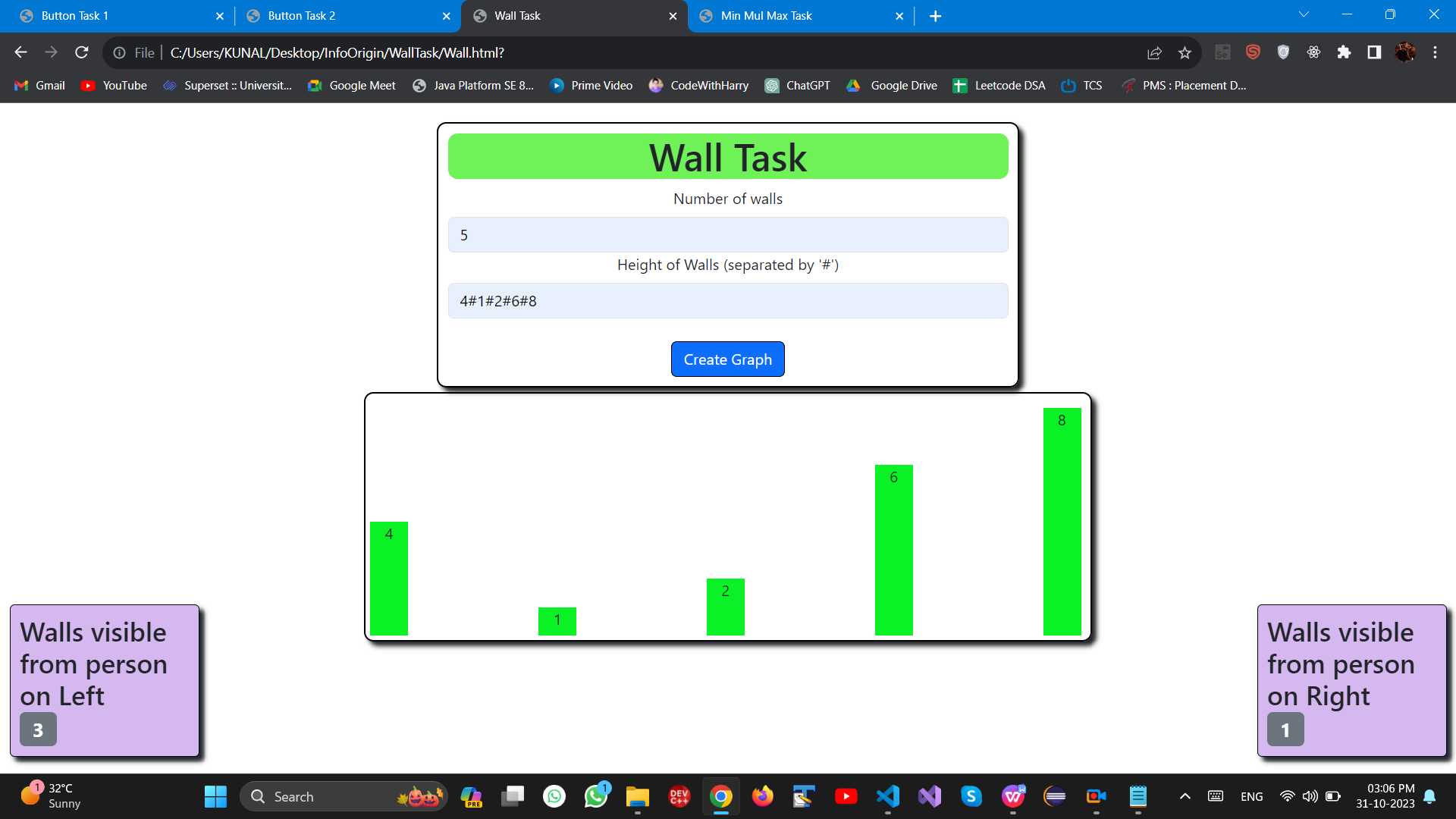
document.getElementById("graphContainer").style.display = "block"; //style of the "graphContainer" element to "block," makes graph on the web page

}

</script>

</body>

</html>

**--------------------------------------------------------------------------------------------------------------------------------------------------------------------------**

**--------------------------------------------------------------------------------------------------------------------------------------------------------------------------**

**Task-4 (Min Max Mul)**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet"

integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL"

crossorigin="anonymous"></script>

<title>Min Mul Max Task</title>

<link rel="stylesheet" href="../Style.css">

</head>

<body>

<div id="container1" class="container text-center">

<div id="head">

<h1>Min Max Mul Task</h1>

</div>

<form>

<div>

<label class="form-label">Enter Min Value</label>

<input type="text" class="form-control" id="min">

</div>

<div>

<label class="form-label">Enter Max Value</label>

<input type="text" class="form-control" id="max">

</div>

<div>

<label class="form-label">Enter Mul Value</label>

<input type="text" class="form-control" id="mul">

</div><br>

<!-- Creating this div to display error -->

<div>

<span id="error"></span>

</div><br>

<button class="btn btn-primary" onclick="validate()">Validate</button>

</form>

</div>

<script>

function validate() {

event.preventDefault(); //Prevents page from reloading wen button is clicked

const min\_val = parseFloat(document.getElementById("min").value);

const max\_val = parseFloat(document.getElementById("max").value);

const mul\_val = parseFloat(document.getElementById("mul").value);

if (isNaN(min\_val) != false && isNaN(mul\_val) != false && isNaN(max\_val) != false) {

//console.log("Not a Number")

document.getElementById("min").style.backgroundColor = "red";

document.getElementById("max").style.backgroundColor = "red";

document.getElementById("mul").style.backgroundColor = "red";

document.getElementById("error").innerHTML = "Input Error..! Enter Numbers only";

return;

}

if (min\_val % mul\_val == 0 && max\_val % mul\_val == 0 && mul\_val <= min\_val && min\_val <= max\_val && mul\_val < max\_val) {

//console.log("All condition are true")

document.getElementById("min").style.backgroundColor = "green";

document.getElementById("max").style.backgroundColor = "green";

document.getElementById("mul").style.backgroundColor = "green";

document.getElementById("error").innerHTML = "All Condition are met.";

}

else {

if ((min\_val % mul\_val != 0 || max\_val % mul\_val != 0) && mul\_val != 0) { // --->Condition A

//console.log("inside else")

if (min\_val % mul\_val !== 0) {

//console.log("min not factor of mul")

document.getElementById("min").style.backgroundColor = "red";

document.getElementById("error").innerHTML = "Min not Divisible by Mul";

}

if (max\_val % mul\_val != 0) {

//console.log("max not factor of mul")

document.getElementById("max").style.backgroundColor = "red";

document.getElementById("error").innerHTML = "Max not Divisible by Mul";

}

}

else if (min\_val > max\_val) { // --->Condition B

//console.log("Max is smaller than min")

document.getElementById("min").style.backgroundColor = "red";

document.getElementById("error").innerHTML = "Min is greater than Max";

}

else if (mul\_val > max\_val) { // --->Condition C

//console.log("Mul greater than max")

document.getElementById("mul").style.backgroundColor = "red";

document.getElementById("error").innerHTML = "Mul is greater than Max";

}

else if (mul\_val > min\_val) { // --->Condition D

//console.log("Mul greater than min")

document.getElementById("mul").style.backgroundColor = "red";

document.getElementById("error").innerHTML = "Mul is greater than Min";

}

else if (min\_val % mul\_val != 0 && max\_val % mul\_val != 0 && mul\_val == 0) { // --->Condition if mul is 0 and min/max are not factors

//console.log("Mul 0")

document.getElementById("mul").style.backgroundColor = "red";

document.getElementById("error").innerHTML = "Mul is 0 so Min/Max are not Factors";

}

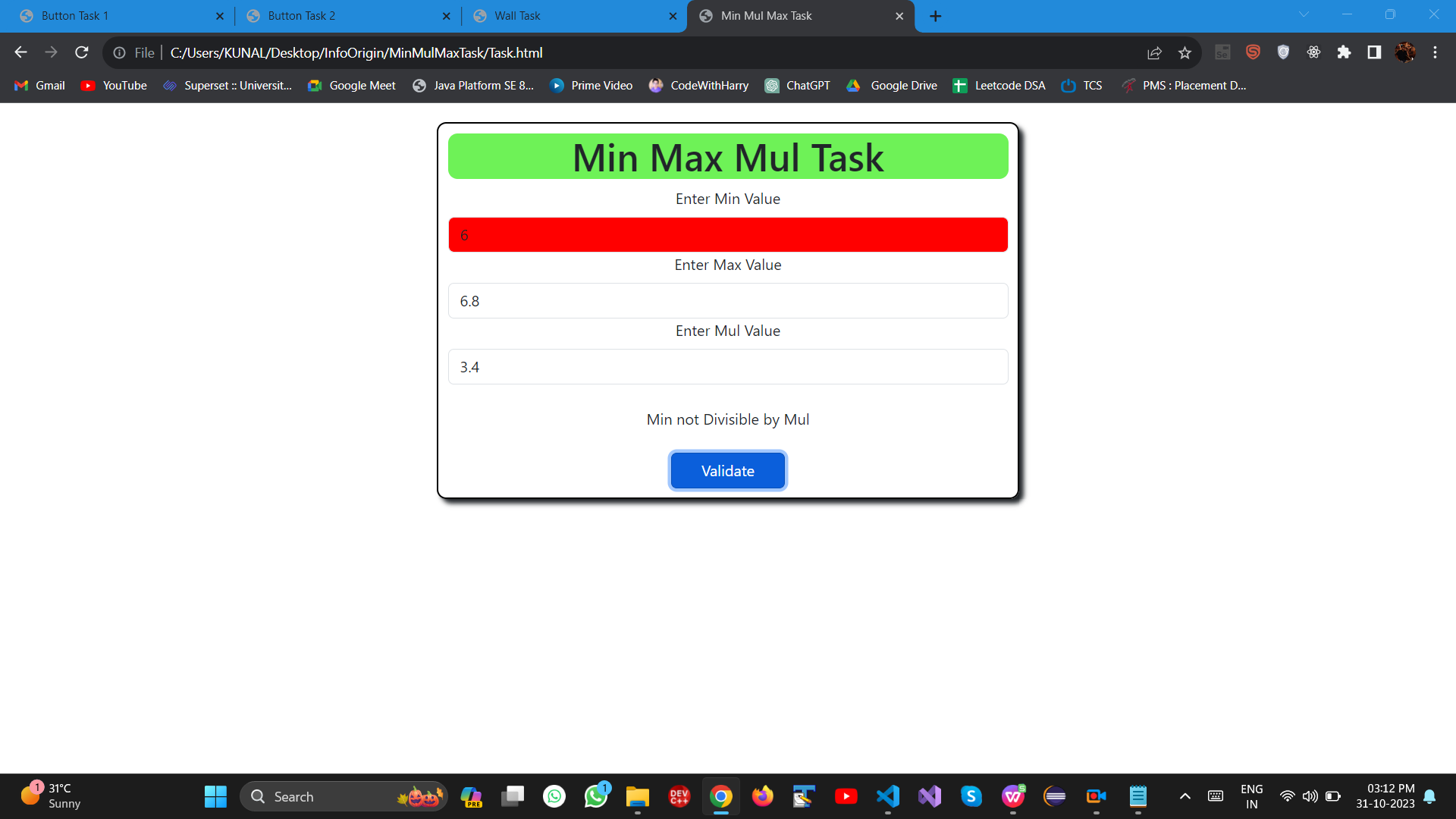
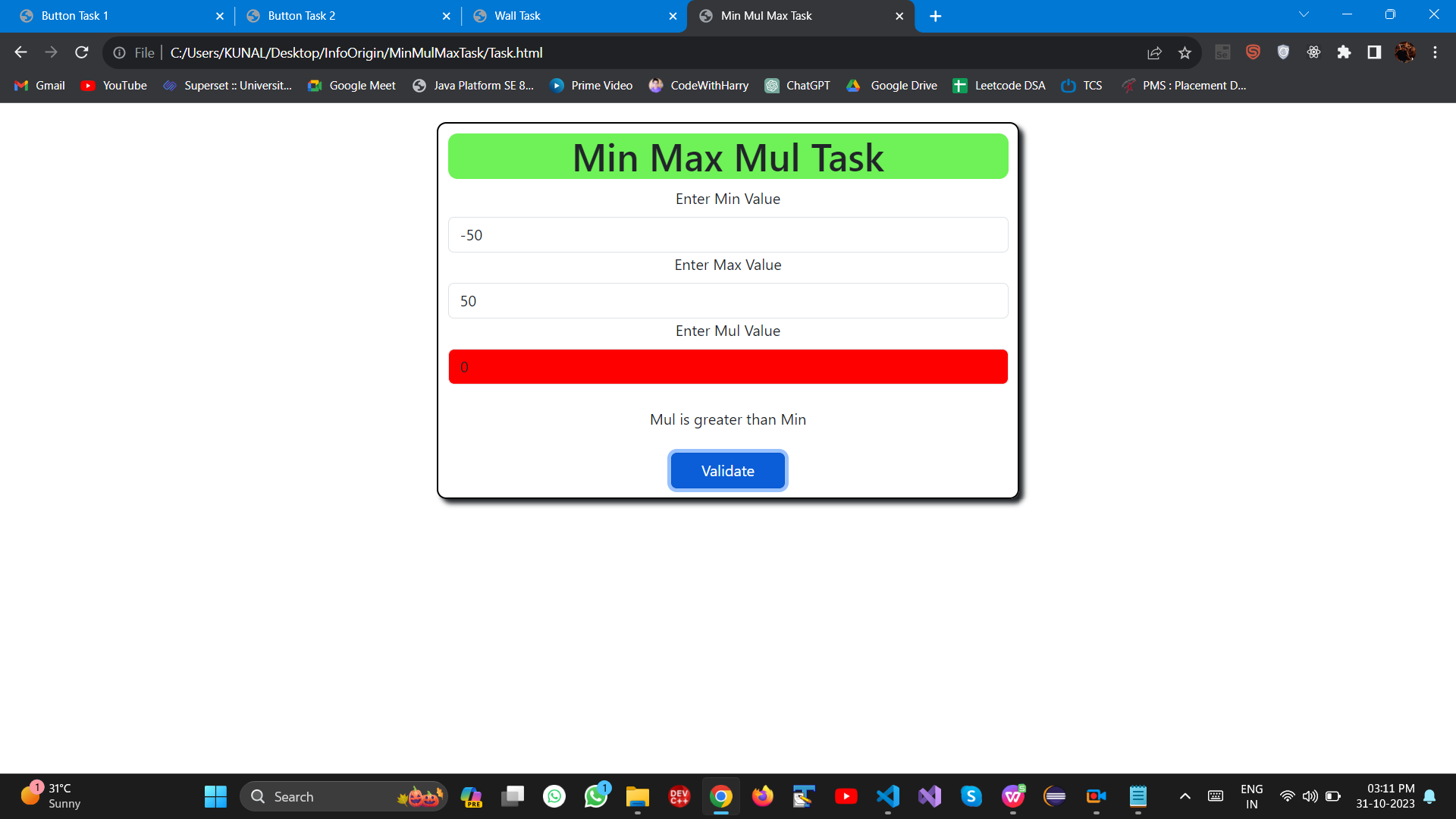
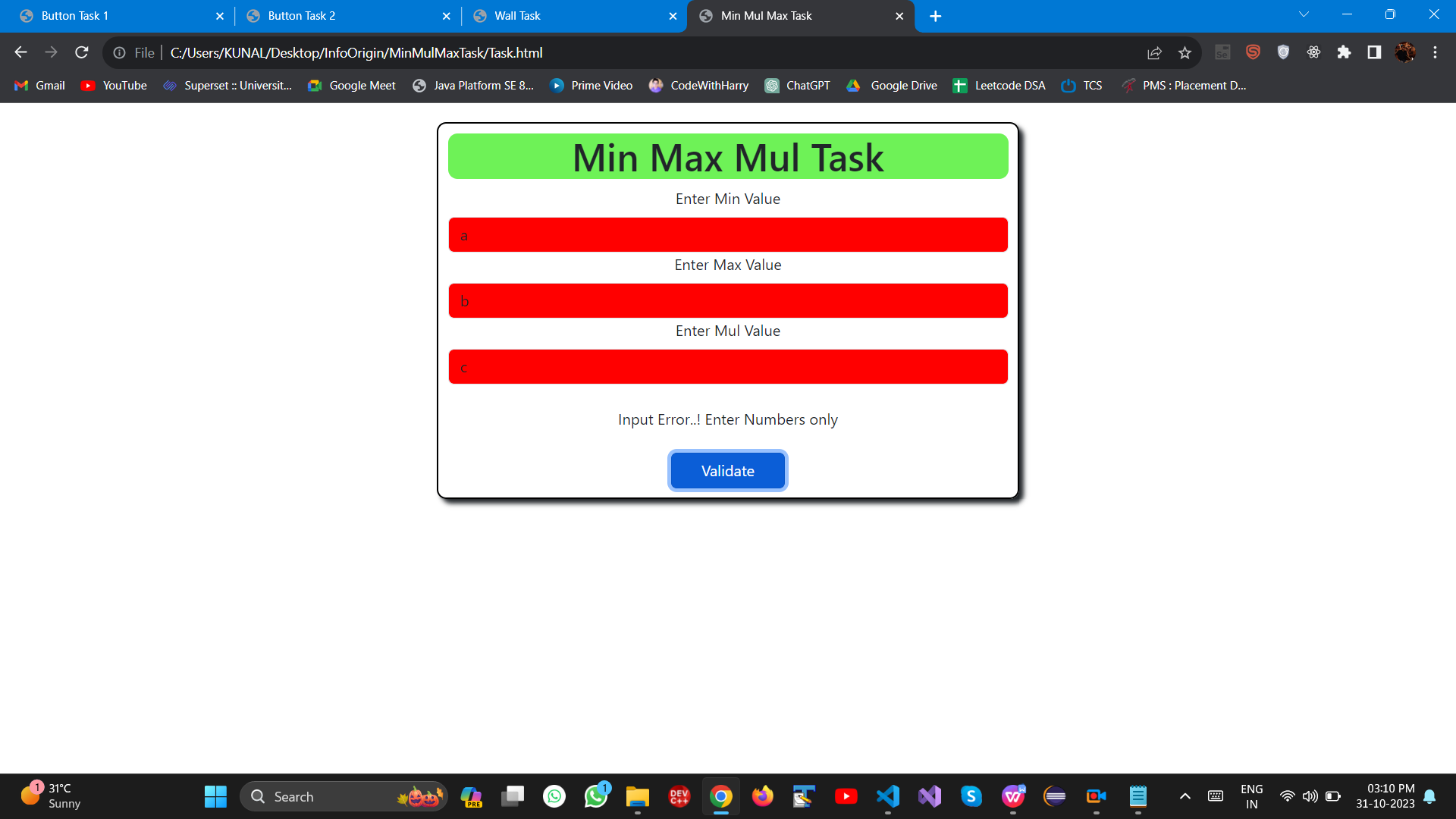
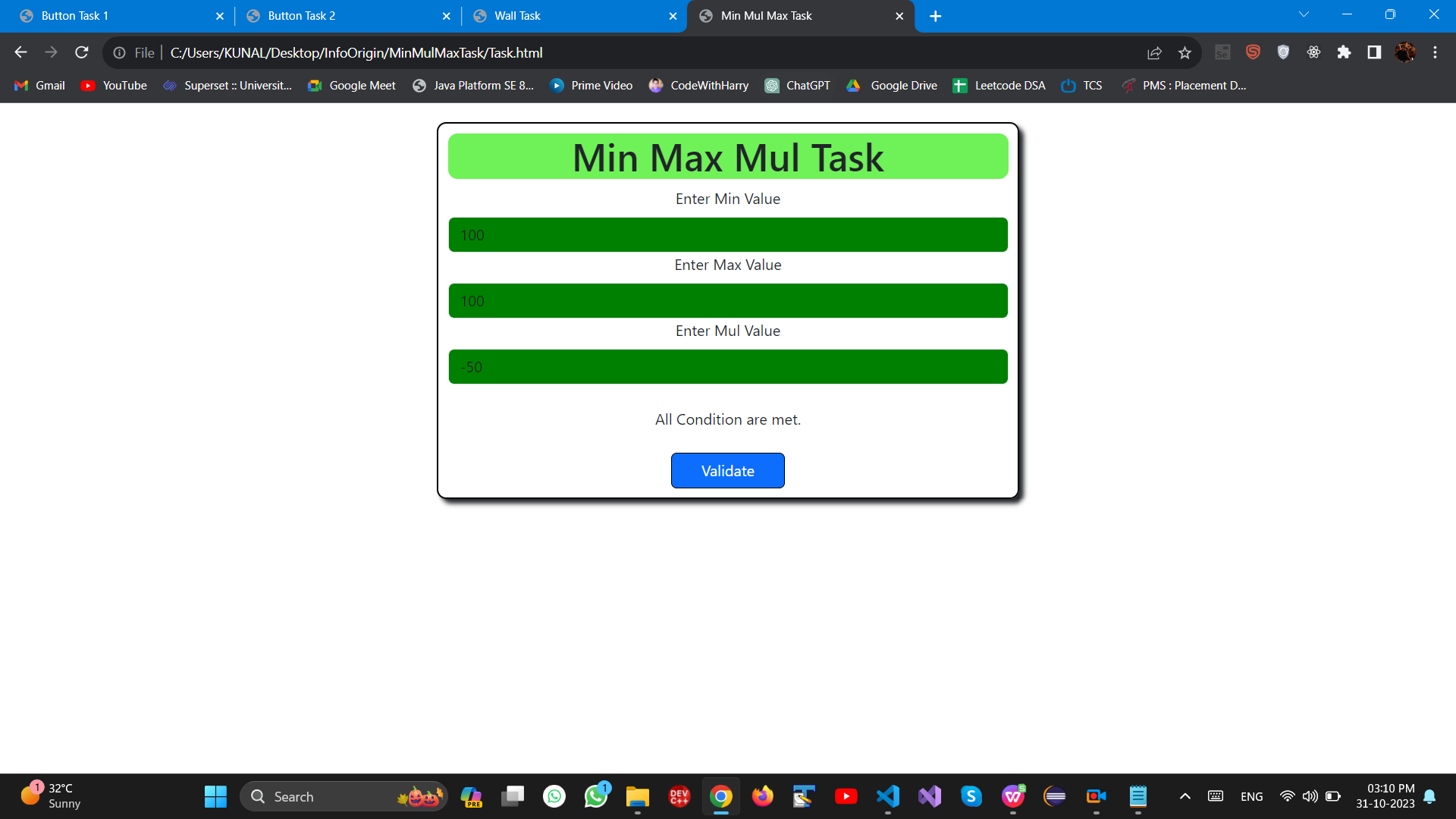
}

}

</script>

</body>

</html>



**TESTING RESULTS:-**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Min** | **Max** | **Mul** | **Test Conditions** | Result | | |
|
|
|
|  |  |  |  | **Testing 1** | **Testing 2** | **Testing 3** |
| 2.5 | 5 | 2.5 | All Conditions are true | pass | pass | pass |
| -2 | 4.1 | 2 | A - false (Max is not divisible by Mul) | pass | pass | pass |
| -2 | 4 | -2 | All Conditions are true | pass | pass | pass |
| -100 | 100 | 1 | A - true, B - true , C -true, D- false(Mul is not less than min) | pass | pass | pass |
| -50 | 50 | 0 | A - false (Mul is not a factor of Max/Min) | fail | pass | pass |
| 100 | 100 | -50 | All Conditions are true | pass | pass | pass |
| 6 | 6.8 | 3.4 | 3.4 | pass | pass | pass |
| 6 | 12 | 6 | All Conditions are true | pass | pass | pass |
| a | b | c | Incorrect Input | pass | pass | pass |
| 100 | 50 | 25 | A - true, B - false (Min is not less than Max) | pass | pass | pass |
| 50 | 125 | 25 | All Conditions are true | pass | pass | pass |
| 50 | 101 | 25 | A - false(Max is not divisible by Mul) | pass | pass | pass |
| 25 | 50 | 0 | A - false (Mul is not a factor of Max/Min) | pass | pass | pass |

**Style.css --> CSS of all the tasks**

**/\* CSS For Button task 1 \*/**

.btn {

border-color: black;

width: 120px;

}

#head {

background-color: rgb(110, 242, 87);

border-radius: 10px;

top: 100px;

align-self: center;

}

.container {

text-align: center !important;

position: relative;

top: 200px;

border: 2px solid black;

padding: 25px;

border-radius: 10px;

width: 30%;

box-shadow: 5px 5px 5px;

}

**/\* CSS for Button task 2 \*/**

.btn {

border-color: black;

width: 120px;

}

#head {

background-color: rgb(110, 242, 87);

border-radius: 10px;

top: 100px;

align-self: center;

}

.container {

text-align: center !important;

position: relative;

top: 200px;

border: 2px solid black;

padding: 25px;

border-radius: 10px;

width: 30%;

box-shadow: 5px 5px 5px;

}

**/\* CSS for Wall Task \*/**

#head {

background-color: rgb(110, 242, 87);

border-radius: 10px;

top: 100px;

align-self: center;

}

#container1 {

text-align: center;

position: relative;

top: 20px;

border: 2px solid black;

padding: 10px;

border-radius: 10px;

width: 40%;

box-shadow: 5px 5px 5px;

}

#graphContainer {

text-align: center;

display: none;

position: relative;

width: 50%;

top: 25px;

padding: 5px;

border: 2px solid black;

box-shadow: 5px 5px 5px;

}

.barContainer {

display: flex;

align-items: flex-end;

justify-content: space-between;

margin-top: 10px;

}

.bar {

width: 40px;

/\* Width of each bar \*/

background-color: rgb(10, 241, 37);

margin-right: 5px;

/\* Space between bars \*/

}

#left {

width: 200px;

background-color: rgb(213, 184, 240);

border: 1px solid black;

box-shadow: 5px 5px 5px;

border-radius: 5px;

padding: 10px;

position: fixed;

bottom: 10px;

left: 10px;

}

#right {

width: 200px;

background-color: rgb(213, 184, 240);

border: 1px solid black;

box-shadow: 5px 5px 5px;

border-radius: 5px;

padding: 10px;

position: fixed;

bottom: 10px;

right: 10px;

}