INTERNSHIP REPORT

WEEK 4 DAY 3

Submitted to:	**Ali Hyder**	Submission Date:	**16th July, 2025**
Internship Domain:	**Front Development**	Internship Name:	**ProSensia**
Student Name:	**Yasal Qamar**	Roll No.	**S25031**

JavaScript Loops and Arrays

Objective

To understand the use of different types of loops in JavaScript and how arrays are used to store and manage collections of data efficiently. Today's focus was on mastering control flow with loops and applying array methods for dynamic and interactive front-end functionalities.

Topics Covered

1. JavaScript Loops

Loops are used to run a block of code multiple times.

a. **for loop**
b. **while loop**
c. **do-while loop**

2. JavaScript Arrays

Arrays are used to store multiple values in a single variable.

Common Array Methods:

- **push() – Adds item to the end**
- **pop() – Removes last item**
- **shift() – Removes first item**
- **unshift() – Adds item to beginning**
- **length – Returns total elements**

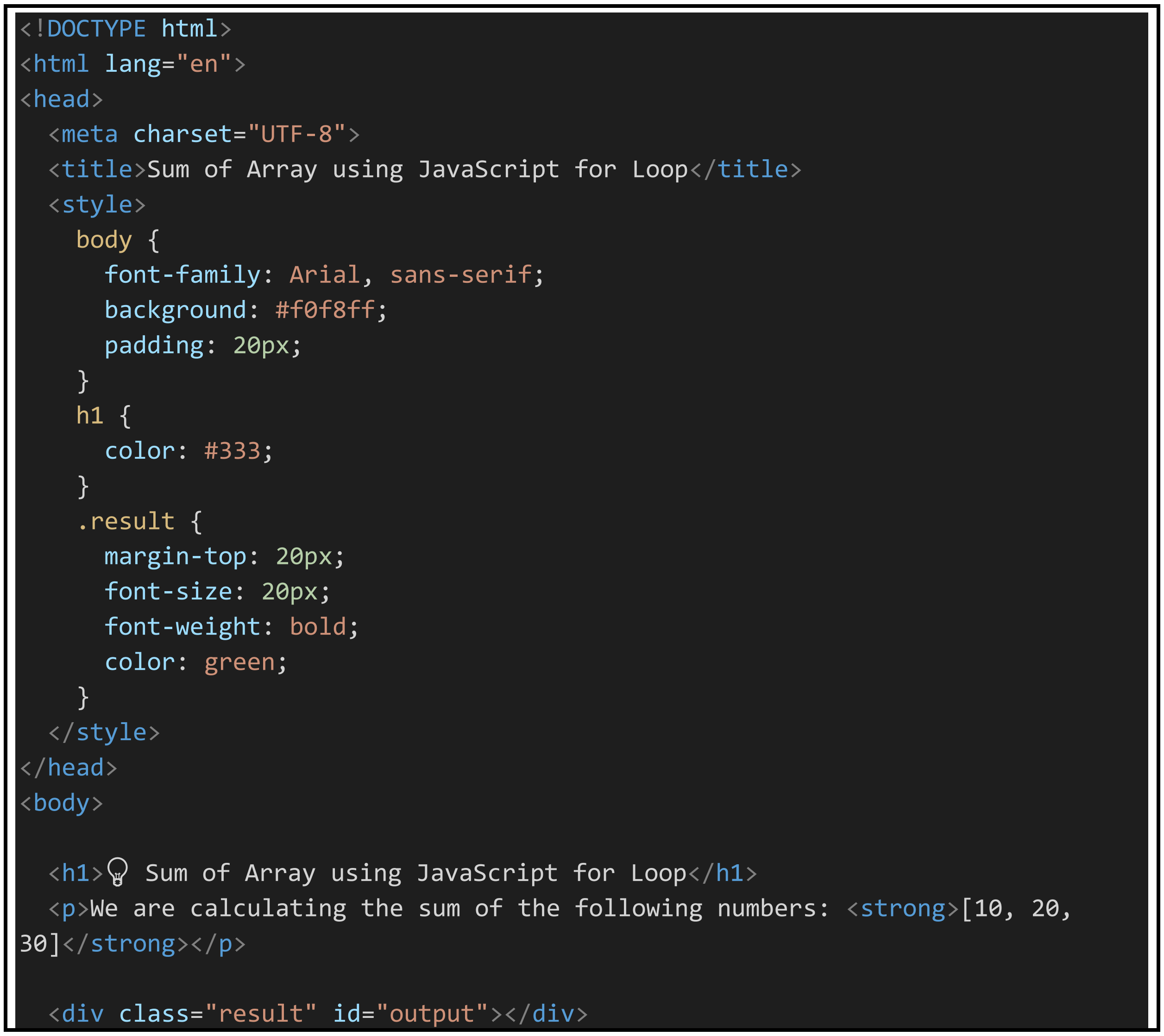
- **forEach() – Iterates through array**

Learning Outcome

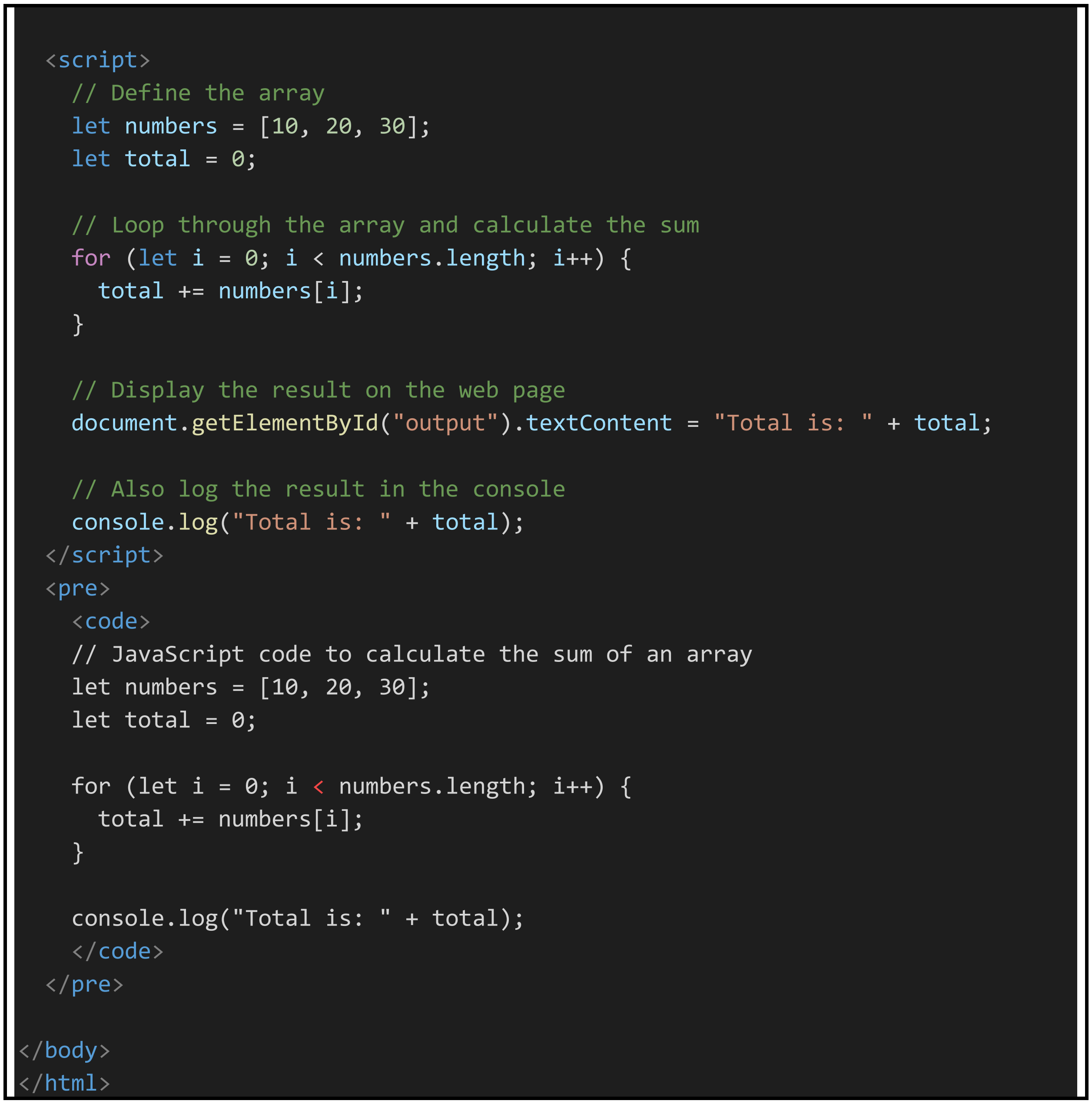
- Gained clear understanding of different loop types and their ideal use cases.
- Learned how to manipulate arrays using built-in methods.
- Practiced using loops with arrays to process and display data dynamically on a webpage.

1. FOR LOOP:

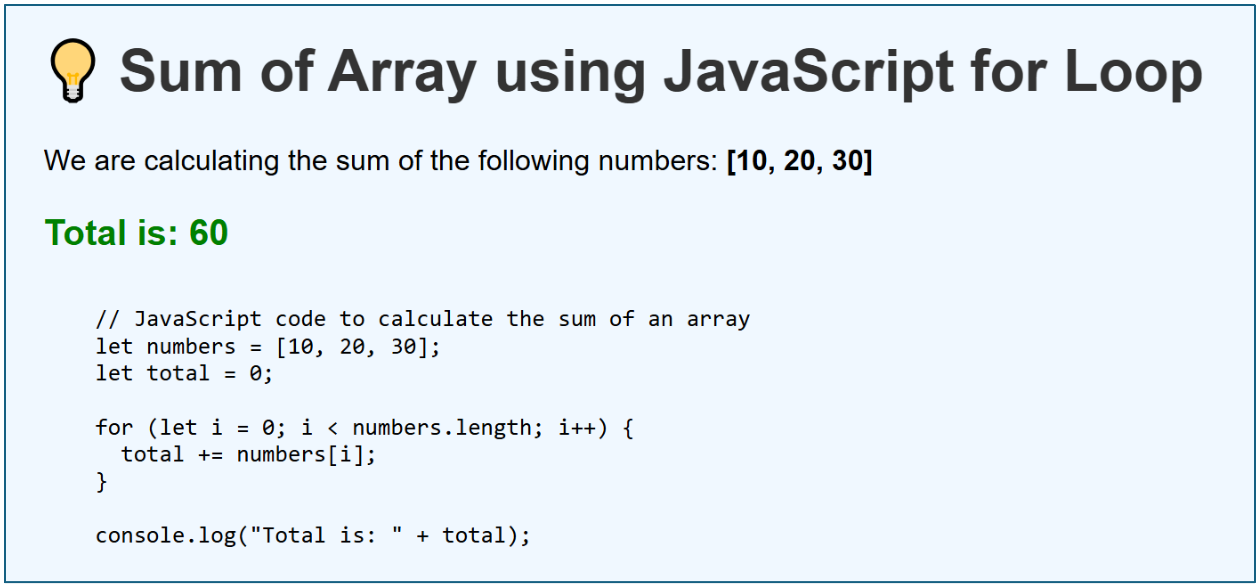
CODING



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Sum of Array using JavaScript for Loop</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: #f0f8ff;
      padding: 20px;
    }
    h1 {
      color: #333;
    }
    .result {
      margin-top: 20px;
      font-size: 20px;
      font-weight: bold;
      color: green;
    }
  </style>
</head>
<body>

  <h1>💡 Sum of Array using JavaScript for Loop</h1>
  <p>We are calculating the sum of the following numbers: <strong>[10, 20,
30]</strong></p>

  <div class="result" id="output"></div>
```



```
  <script>
    // Define the array
    let numbers = [10, 20, 30];
    let total = 0;

    // Loop through the array and calculate the sum
    for (let i = 0; i < numbers.length; i++) {
      total += numbers[i];
    }

    // Display the result on the web page
    document.getElementById("output").textContent = "Total is: " + total;

    // Also log the result in the console
    console.log("Total is: " + total);
  </script>
  <pre>
    <code>
    // JavaScript code to calculate the sum of an array
    let numbers = [10, 20, 30];
    let total = 0;

    for (let i = 0; i < numbers.length; i++) {
      total += numbers[i];
    }

    console.log("Total is: " + total);
    </code>
  </pre>

</body>
</html>
```



💡 Sum of Array using JavaScript for Loop

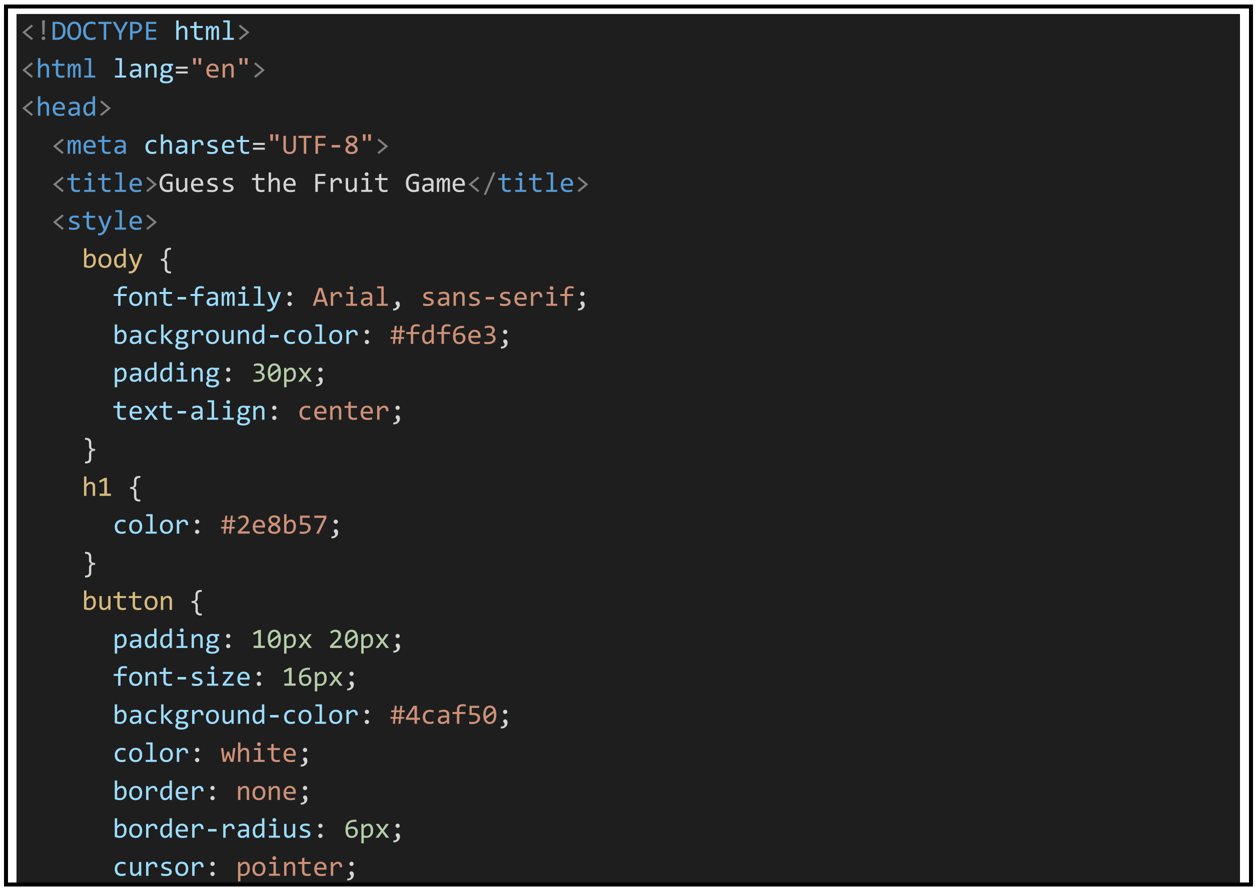
We are calculating the sum of the following numbers: **[10, 20, 30]**

Total is: 60

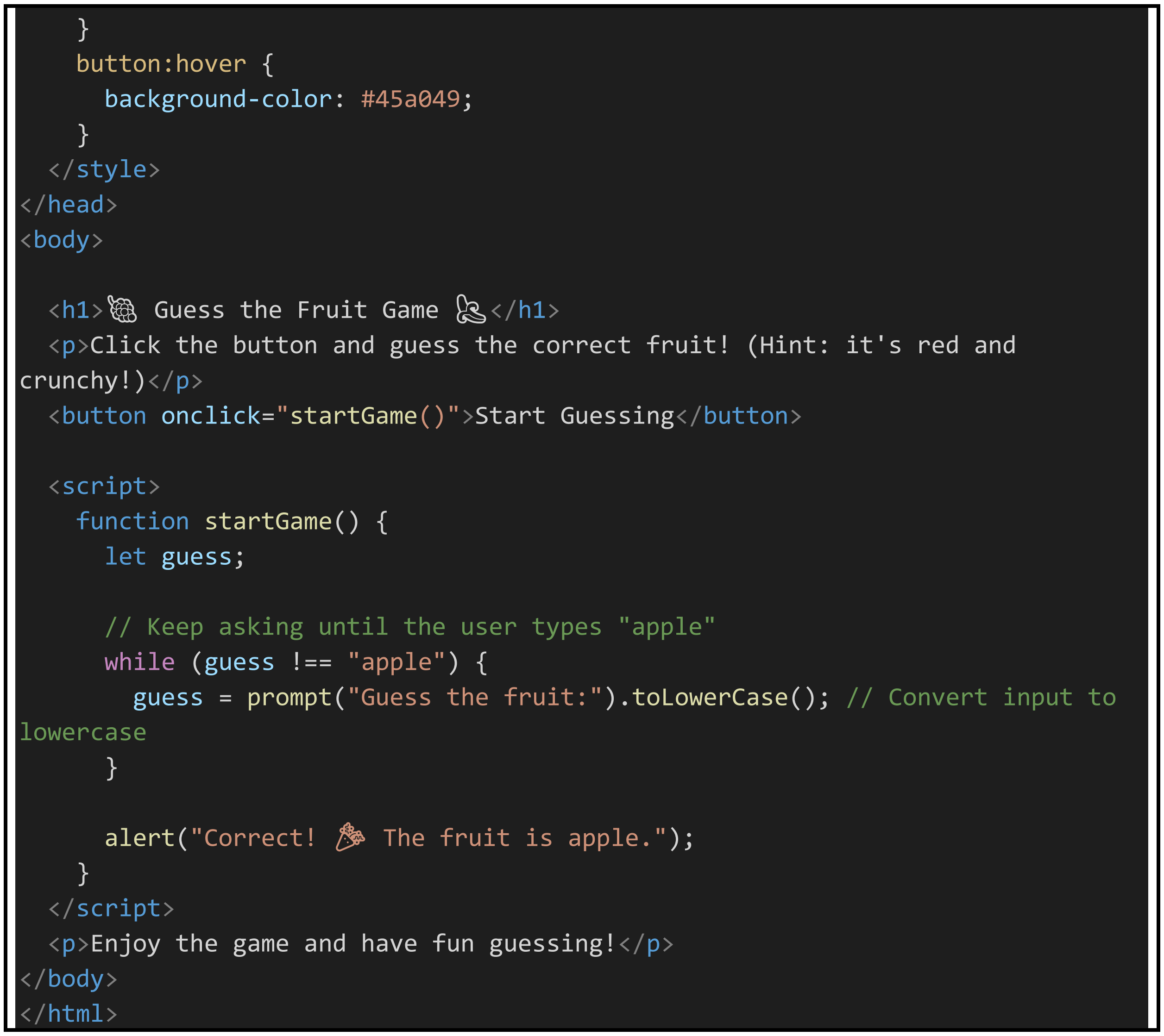
```
// JavaScript code to calculate the sum of an array
let numbers = [10, 20, 30];
let total = 0;

for (let i = 0; i < numbers.length; i++) {
  total += numbers[i];
}

console.log("Total is: " + total);
```

2. WHILE LOOP:

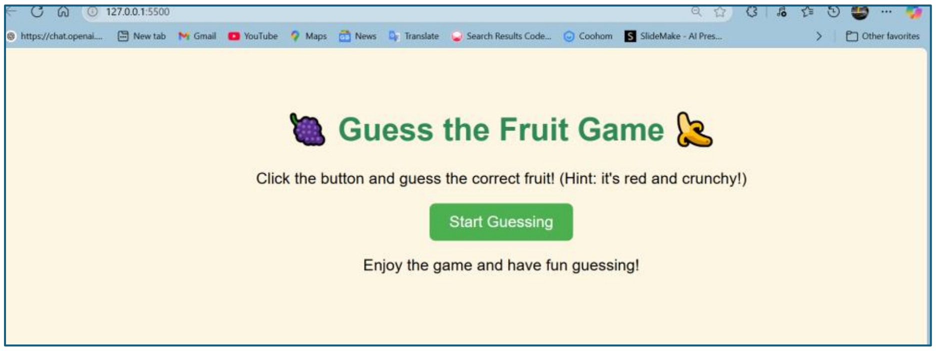
CODING



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Guess the Fruit Game</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #fdf6e3;
      padding: 30px;
      text-align: center;
    }
    h1 {
      color: #2e8b57;
    }
    button {
      padding: 10px 20px;
      font-size: 16px;
      background-color: #4caf50;
      color: white;
      border: none;
      border-radius: 6px;
      cursor: pointer;
```



```
    }
    button:hover {
      background-color: #45a049;
    }
  </style>
</head>
<body>

  <h1>🍇 Guess the Fruit Game 🍌</h1>
  <p>Click the button and guess the correct fruit! (Hint: it's red and 
crunchy!)</p>
  <button onclick="startGame()">Start Guessing</button>

  <script>
    function startGame() {
      let guess;

      // Keep asking until the user types "apple"
      while (guess !== "apple") {
        guess = prompt("Guess the fruit:").toLowerCase(); // Convert input to 
lowercase
      }

      alert("Correct! 🎉 The fruit is apple.");
    }
  </script>
  <p>Enjoy the game and have fun guessing!</p>
</body>
</html>
```



🍇 Guess the Fruit Game 🍌

Click the button and guess the correct fruit! (Hint: it's red and crunchy!)

Start Guessing

Enjoy the game and have fun guessing!

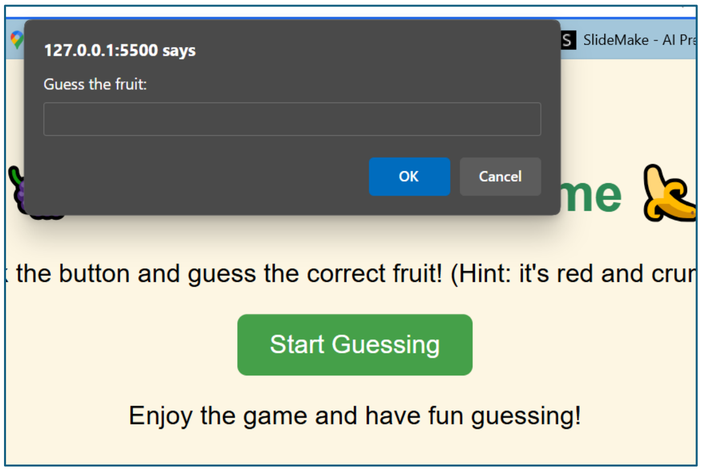
➢ **let guess;**

This declares a variable guess without giving it a value yet. We'll store the user's input here later.

➢ **while (guess !== "apple")**

This is a while loop that keeps repeating as long as the user's guess is not "apple".

"!==" means "not equal to"


127.0.0.1:5500 says

Guess the fruit:

the button and guess the correct fruit! (Hint: it's red and crun

Enjoy the game and have fun guessing!

guess = prompt("Guess the fruit:").toLowerCase();

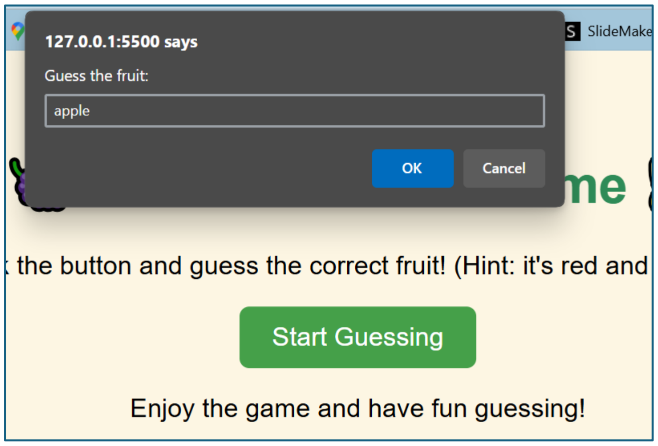
This shows a popup input box (prompt) asking the user to type a fruit.

- prompt(...) → lets the user type something.
- .toLowerCase() → makes sure it's in small letters so Apple, APPLE, or apple all match.

📌 Example:

If the user types APPLE, this line converts it to apple.

When the user finally types "apple", the loop ends.


127.0.0.1:5500 says

Guess the fruit:

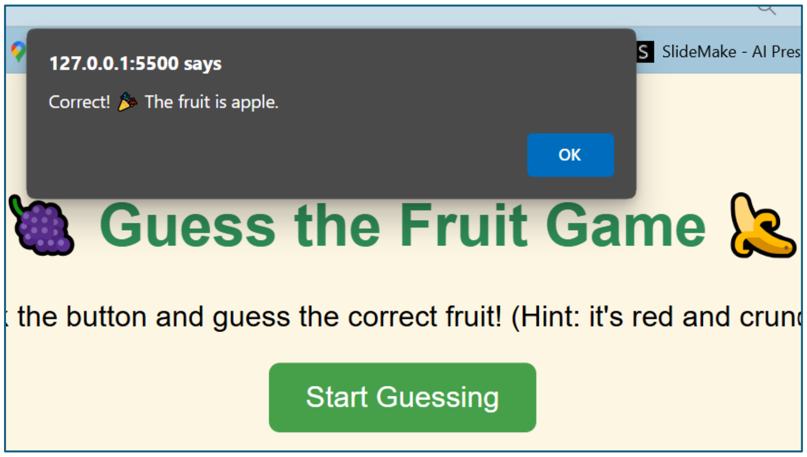
apple

the button and guess the correct fruit! (Hint: it's red and

Enjoy the game and have fun guessing!

alert("Correct! 🎉 The fruit is apple.");

Once the correct answer is given, it shows a success message using alert().



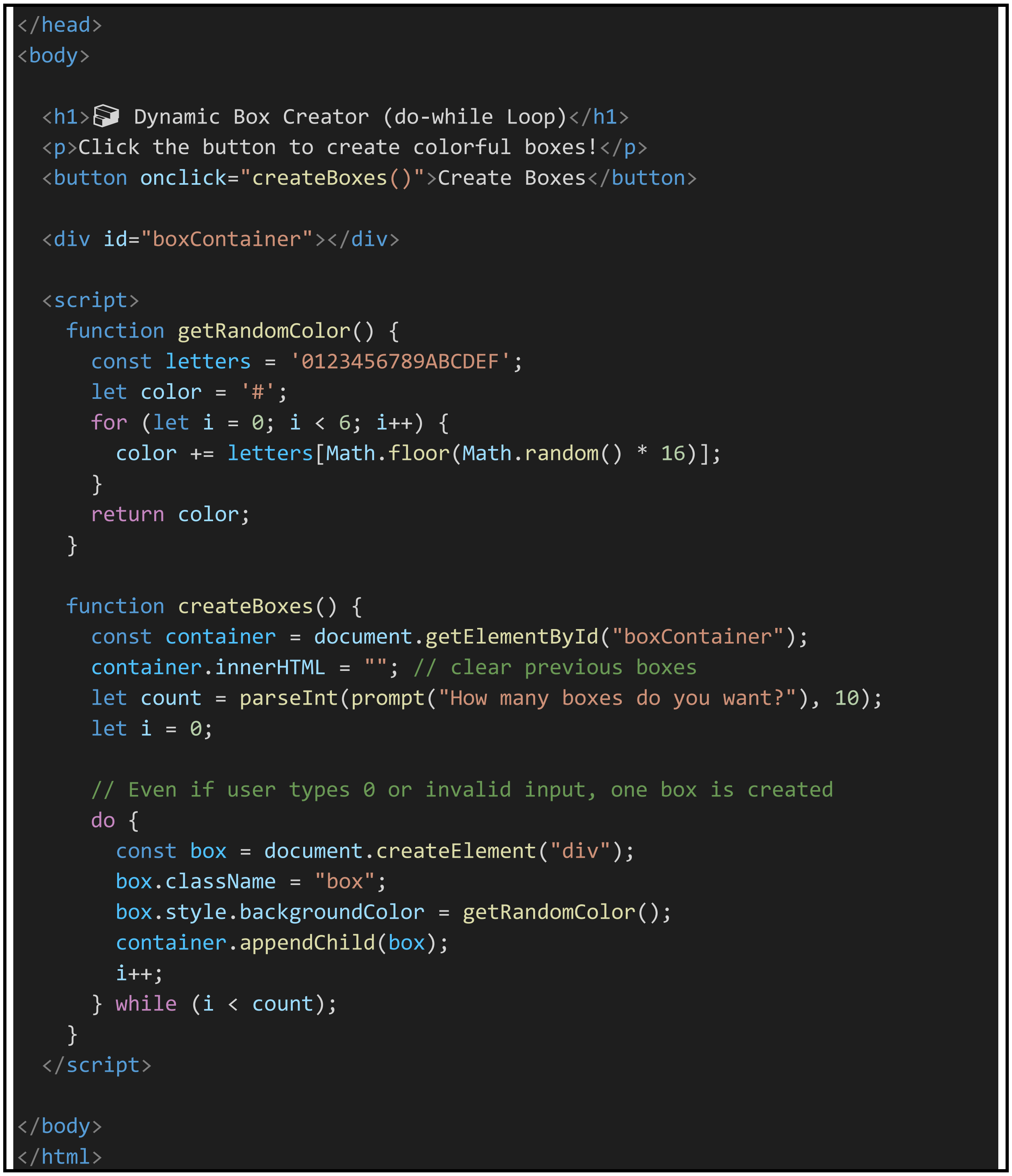
127.0.0.1:5500 says

Correct! 🎉 The fruit is apple.

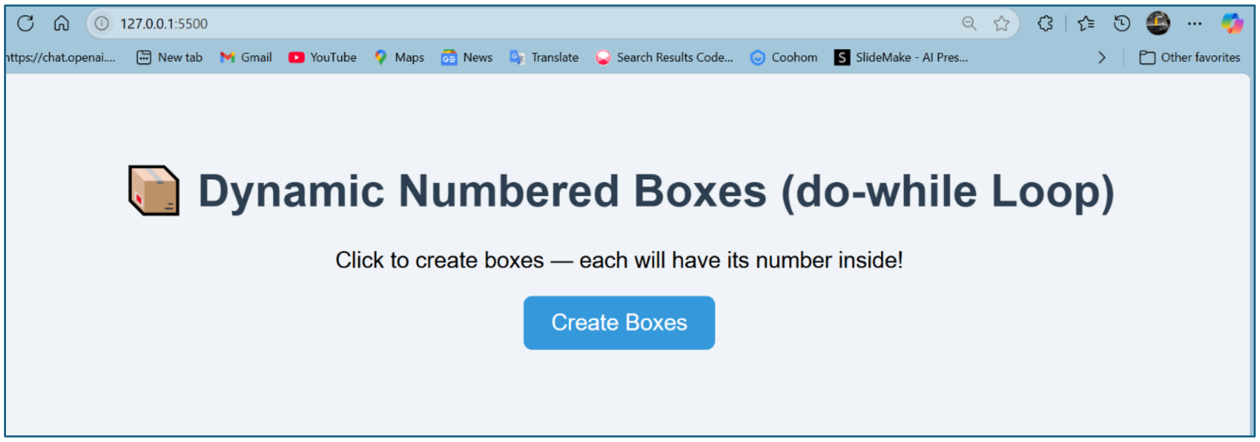
OK
SlideMake - AI Pres
🍇 Guess the Fruit Game 🍌
the button and guess the correct fruit! (Hint: it's red and crun
Start Guessing

3. DO-WHILE LOOP:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Dynamic Boxes with do-while</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f4f4f4;
      padding: 30px;
      text-align: center;
    }
    h1 {
      color: #333;
    }
    .box {
      width: 100px;
      height: 100px;
      margin: 10px;
      display: inline-block;
      border-radius: 8px;
      box-shadow: 0 0 5px rgba(0,0,0,0.2);
    }
  </style>
```



```html
</head>
<body>

  <h1>📦 Dynamic Box Creator (do-while Loop)</h1>
  <p>Click the button to create colorful boxes!</p>
  <button onclick="createBoxes()">Create Boxes</button>

  <div id="boxContainer"></div>

  <script>
    function getRandomColor() {
      const letters = '0123456789ABCDEF';
      let color = '#';
      for (let i = 0; i < 6; i++) {
        color += letters[Math.floor(Math.random() * 16)];
      }
      return color;
    }

    function createBoxes() {
      const container = document.getElementById("boxContainer");
      container.innerHTML = ""; // clear previous boxes
      let count = parseInt(prompt("How many boxes do you want?"), 10);
      let i = 0;

      // Even if user types 0 or invalid input, one box is created
      do {
        const box = document.createElement("div");
        box.className = "box";
        box.style.backgroundColor = getRandomColor();
        container.appendChild(box);
        i++;
      } while (i < count);
    }
  </script>

</body>
</html>
```


127.0.0.1:5500
https://chat.openai... New tab Gmail YouTube Maps News Translate Search Results Code... Coohom SlideMake - AI Pres... Other favorites
Dynamic Numbered Boxes (do-while Loop)
Click to create boxes — each will have its number inside!
Create Boxes

Step 1: let i = 0;

We start counting from i = 0.

Step 2: do { ... }

This block **runs first**, no matter what the user entered.

Inside the block:

const box = document.createElement("div");

Creates a new <div> element in JavaScript (a box).

box.className = "box";

Adds the CSS class box for styling (width, height, shadow, etc).

box.style.backgroundColor = getRandomColor();

👉 Sets a random background color using the getRandomColor() function.javascript

container.appendChild(box);

👉 Adds the box to the webpage inside <div id="boxContainer">.

i++;

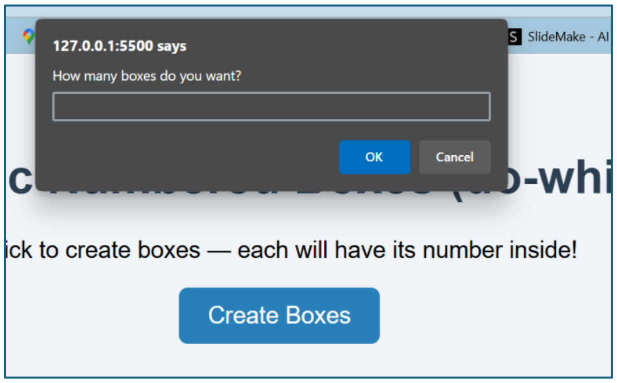
👉 Increases the counter i by 1.

Step 3: while (i < count);

After running once, it checks:

Is i still less than count (number user entered)?

- If **yes**, repeat the loop.
- If **no**, stop the loop.

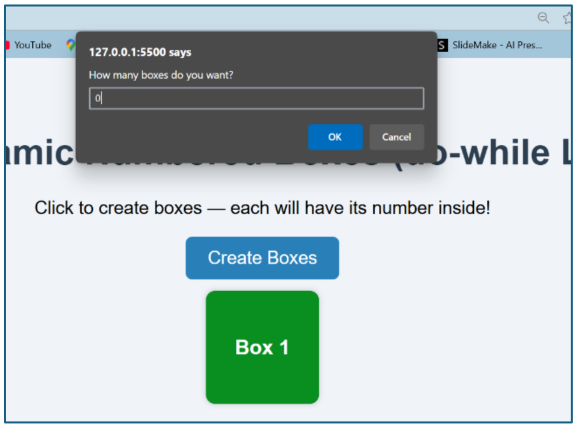

SlideMake - AI
127.0.0.1:5500 says
How many boxes do you want?
OK
Cancel
c
-whi
ick to create boxes — each will have its number inside!
Create Boxes

Why Use do-while Here?

Imagine the user enters **0 boxes**.

- while loop won't run at all (0 < 0 is false).
- But do-while still runs **once**, so **one box is shown**.

📌 That's the special behavior: do-while **always runs at least one time.**

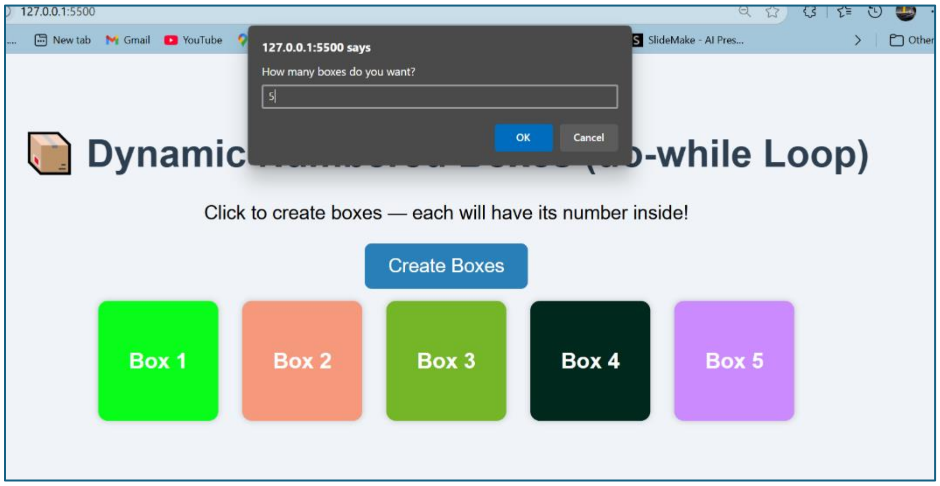


box.textContent = "Box " + (i + 1);

➢ This line adds the box number inside each box.

Uses line-height: 100px; in CSS

- ➢ To vertically center the number.



127.0.0.1:5500 says

How many boxes do you want?

5

Dynamic ... -while Loop)

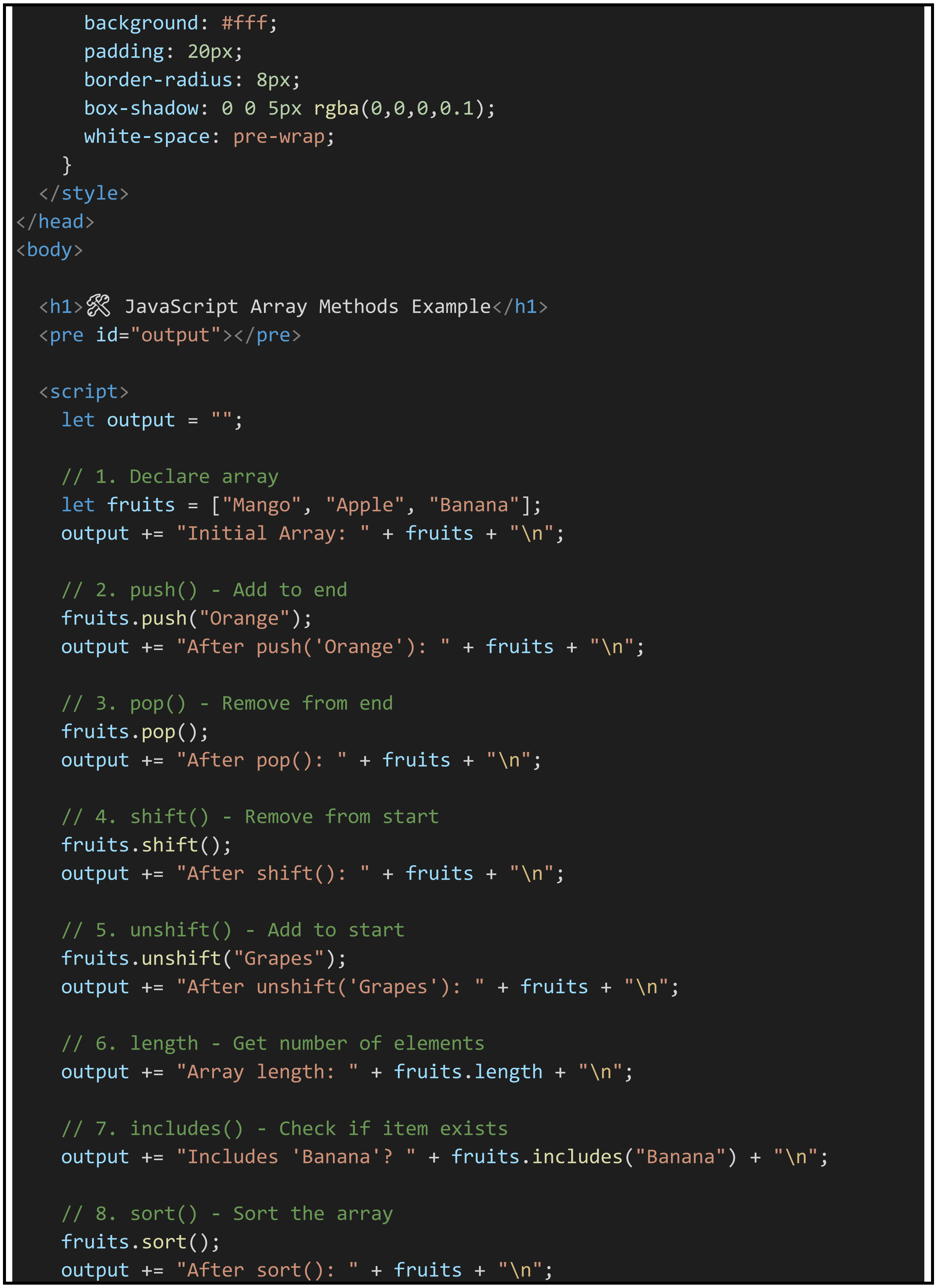
Click to create boxes — each will have its number inside!

- ➢ If the user enters 5, you'll see:
  - **Box 1**
  - **Box 2**
  - **Box 3**
  - **Box 4**
  - **Box 5**

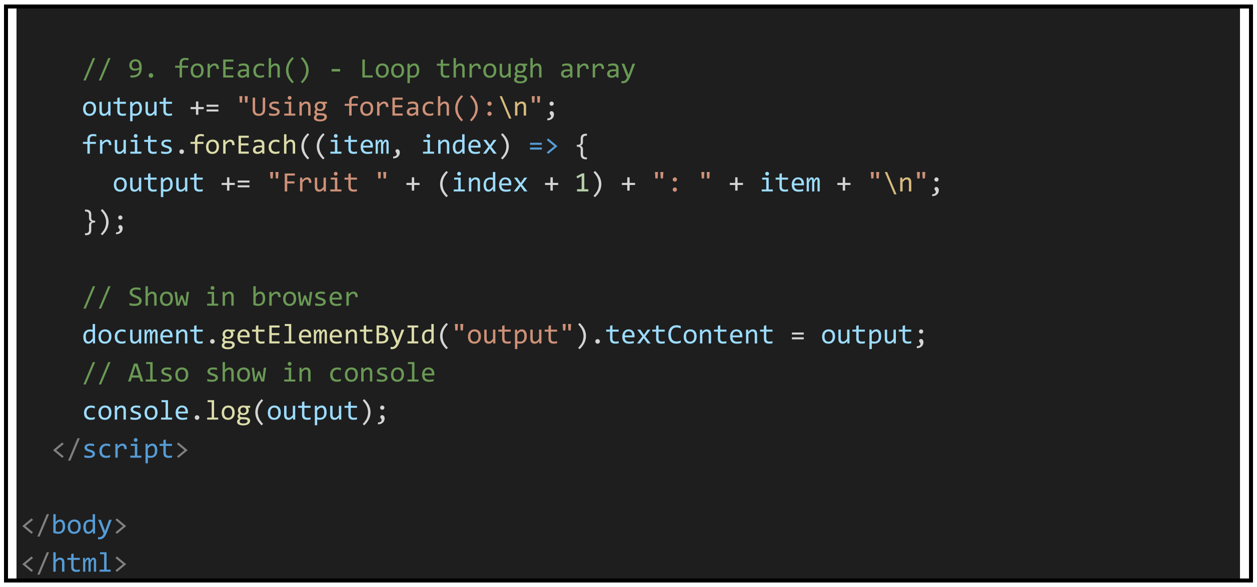
All with different colors and numbers inside.

4. ARRAYS JAVA:

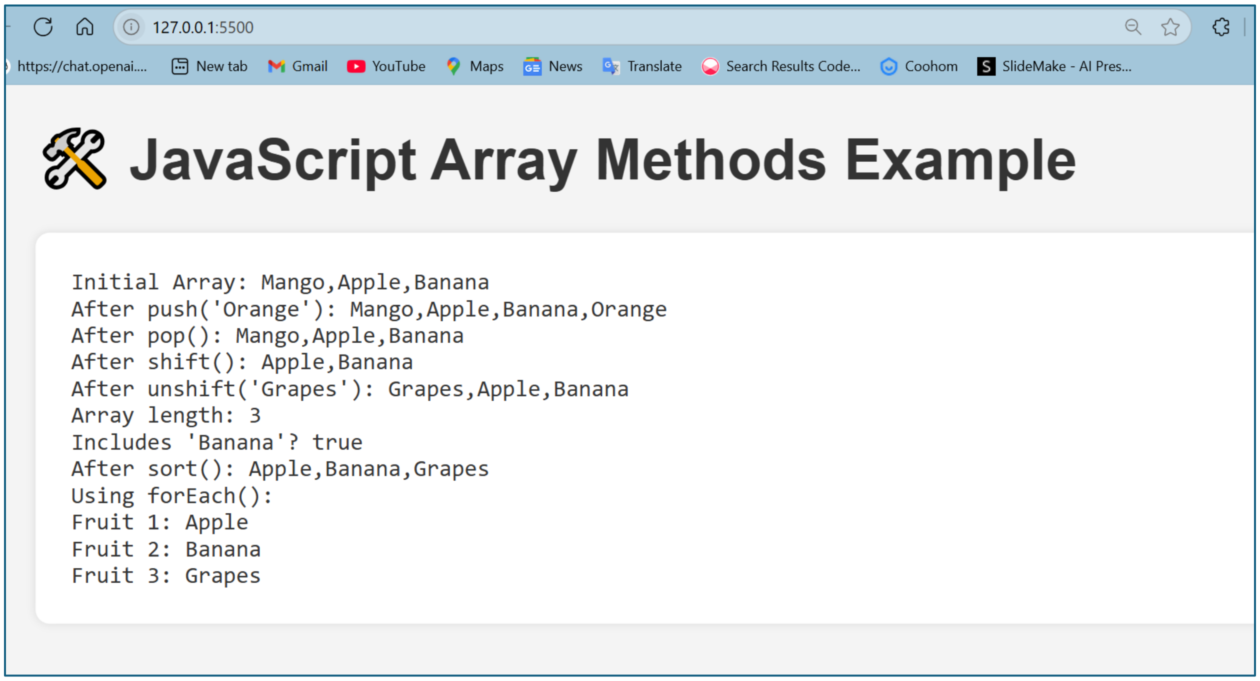
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Array Methods Demo</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: #f4f4f4;
      padding: 30px;
      color: #333;
    }
    pre {
```



```html
      background: #fff;
      padding: 20px;
      border-radius: 8px;
      box-shadow: 0 0 5px rgba(0,0,0,0.1);
      white-space: pre-wrap;
    }
  </style>
</head>
<body>

  <h1>🛠️ JavaScript Array Methods Example</h1>
  <pre id="output"></pre>

  <script>
    let output = "";

    // 1. Declare array
    let fruits = ["Mango", "Apple", "Banana"];
    output += "Initial Array: " + fruits + "\n";

    // 2. push() - Add to end
    fruits.push("Orange");
    output += "After push('Orange'): " + fruits + "\n";

    // 3. pop() - Remove from end
    fruits.pop();
    output += "After pop(): " + fruits + "\n";

    // 4. shift() - Remove from start
    fruits.shift();
    output += "After shift(): " + fruits + "\n";

    // 5. unshift() - Add to start
    fruits.unshift("Grapes");
    output += "After unshift('Grapes'): " + fruits + "\n";

    // 6. length - Get number of elements
    output += "Array length: " + fruits.length + "\n";

    // 7. includes() - Check if item exists
    output += "Includes 'Banana'? " + fruits.includes("Banana") + "\n";

    // 8. sort() - Sort the array
    fruits.sort();
    output += "After sort(): " + fruits + "\n";
```



```
    // 9. forEach() - Loop through array
    output += "Using forEach():\n";
    fruits.forEach((item, index) => {
      output += "Fruit " + (index + 1) + ": " + item + "\n";
    });

    // Show in browser
    document.getElementById("output").textContent = output;
    // Also show in console
    console.log(output);
  </script>

</body>
</html>
```



🛠 JavaScript Array Methods Example

```
Initial Array: Mango,Apple,Banana
After push('Orange'): Mango,Apple,Banana,Orange
After pop(): Mango,Apple,Banana
After shift(): Apple,Banana
After unshift('Grapes'): Grapes,Apple,Banana
Array length: 3
Includes 'Banana'? true
After sort(): Apple,Banana,Grapes
Using forEach():
Fruit 1: Apple
Fruit 2: Banana
Fruit 3: Grapes
```

Display the Output

document.getElementById("output").textContent = output;

- Shows everything inside the <pre> element in the browser.

console.log(output);

- Also prints it in the **browser console** (for developers)

Conclusion:

This session was highly productive in building my confidence in JavaScript fundamentals. By combining HTML, CSS, and JavaScript, I developed interactive web pages that clearly demonstrated variable behavior and data type usage an essential step in frontend development.