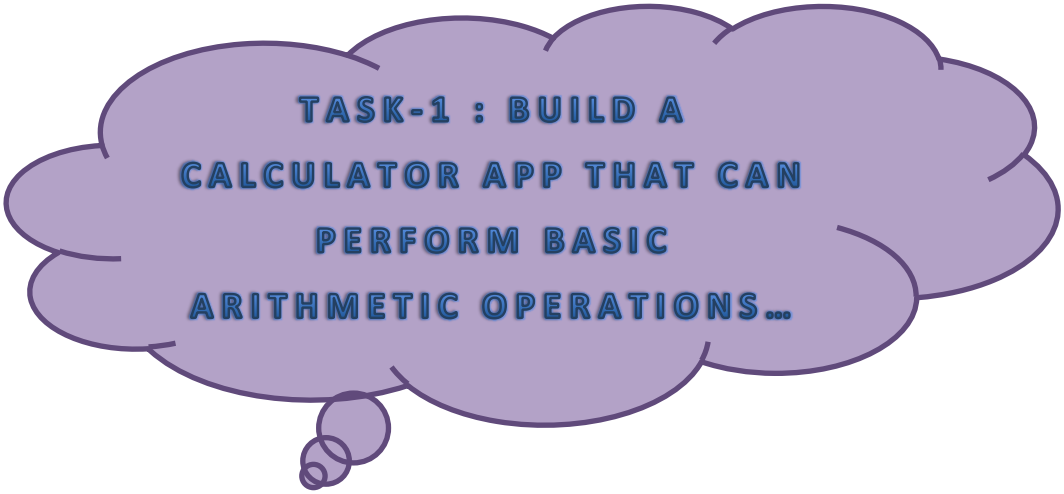



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Simple Calculator App</title>
  <style>
    body {
      background: #f4f4f4;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      font-family: 'Segoe UI', sans-serif;
    }

    .calculator {
      background: #ffffff;
      padding: 20px;
```

```
  border-radius: 20px;
  box-shadow: 0 8px 16px rgba(0,0,0,0.2);
  width: 320px;
}

#display {
  width: 100%;
  height: 60px;
  font-size: 24px;
  text-align: right;
  margin-bottom: 20px;
  padding: 10px;
  border-radius: 10px;
  border: 1px solid #ccc;
}

.buttons {
  display: grid;
  grid-template-columns: repeat(4, 1fr);
  gap: 10px;
}

button {
  padding: 20px;
  font-size: 18px;
```

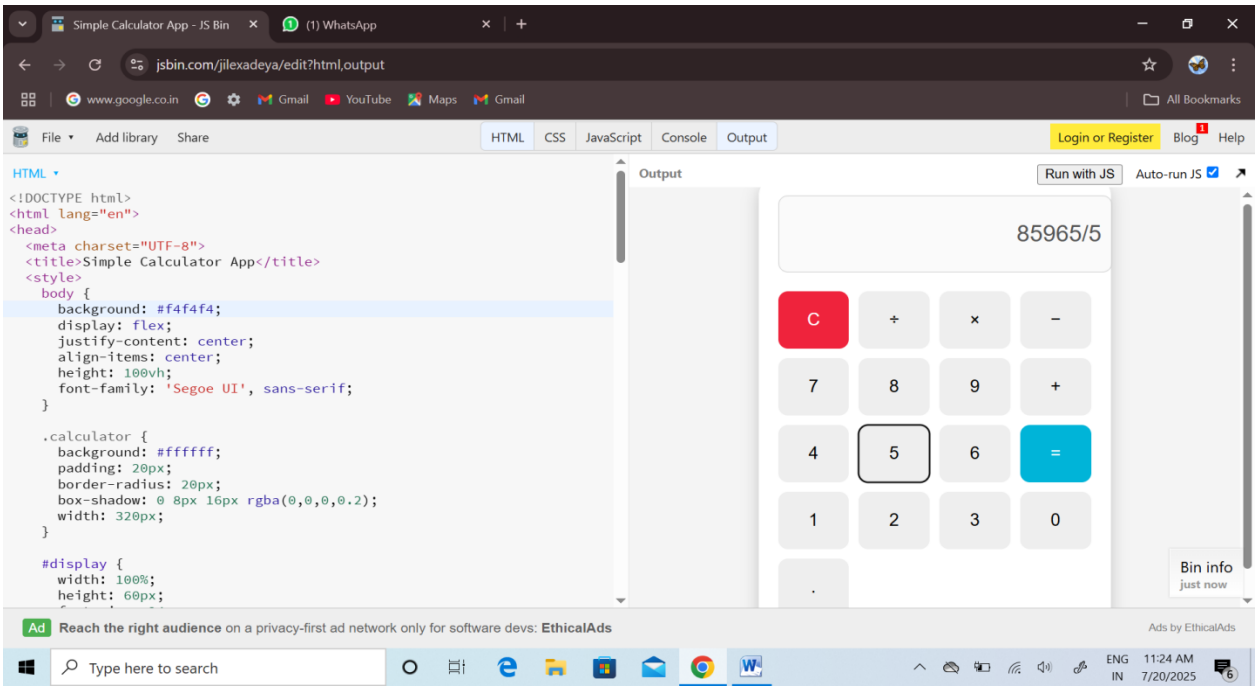
```
  border: none;
  border-radius: 10px;
  background: #eee;
  cursor: pointer;
  transition: background 0.2s;
}

button:hover {
  background: #dcdcdc;
}

button.operator {
  background: #fca311;
  color: white;
}

button.operator:hover {
  background: #f77f00;
}

button.equal {
  background: #00b4d8;
  color: white;
}
```

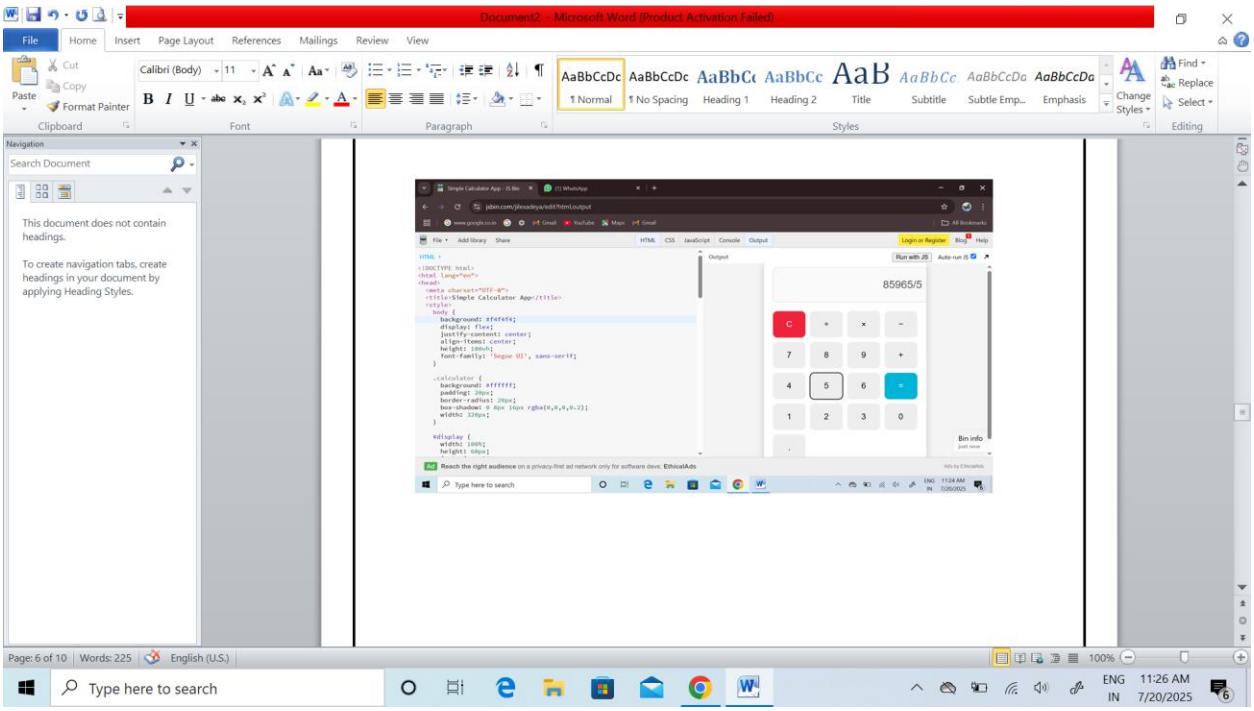
```
    button.equal:hover {
      background: #0077b6;
    }

    button.clear {
      background: #ef233c;
      color: white;
    }

    button.clear:hover {
      background: #d90429;
    }
  </style>
</head>
<body>

<div class="calculator">
  <input type="text" id="display" disabled placeholder="0" />
  <div class="buttons">
    <button class="clear" onclick="clearDisplay()">C</button>
    <button onclick="appendToDisplay('/')">÷</button>
    <button onclick="appendToDisplay('*')">×</button>
    <button onclick="appendToDisplay('-')">−</button>
    <button onclick="appendToDisplay('7')">7</button>
    <button onclick="appendToDisplay('8')">8</button>
```

```
    <button onclick="appendToDisplay('9')">9</button>
    <button onclick="appendToDisplay('+')">+</button>
    <button onclick="appendToDisplay('4')">4</button>
    <button onclick="appendToDisplay('5')">5</button>
    <button onclick="appendToDisplay('6')">6</button>
    <button class="equal" onclick="calculateResult()">=</button>
    <button onclick="appendToDisplay('1')">1</button>
    <button onclick="appendToDisplay('2')">2</button>
    <button onclick="appendToDisplay('3')">3</button>
    <button onclick="appendToDisplay('0')">0</button>
    <button onclick="appendToDisplay('.')">.</button>
  </div>
</div>

<script>
  function appendToDisplay(value) {
    document.getElementById('display').value += value;
  }

  function clearDisplay() {
    document.getElementById('display').value = '';
  }

  function calculateResult() {
    try {
```

```
    const result = eval(document.getElementById('display').value);
    document.getElementById('display').value = result;
  } catch {
    document.getElementById('display').value = 'Error';
  }
 }
</script>

</body>
</html>
```

OUTPUT SCREENSHOT





DESCRIPTION

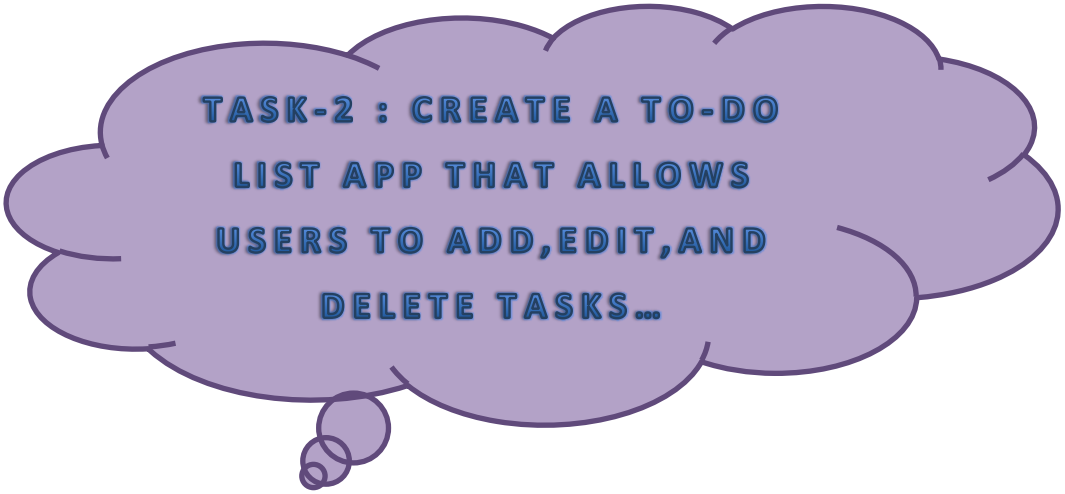
This is a simple calculator app developed using HTML, CSS, and JavaScript. It performs basic arithmetic operations such as addition, subtraction, multiplication, and division. The app has a clean interface and can be used in any modern web browser.

TECHNOLOGIES USED

HTML – For structuring the interface

CSS – For styling the calculator

JavaScript – For implementing calculation logic

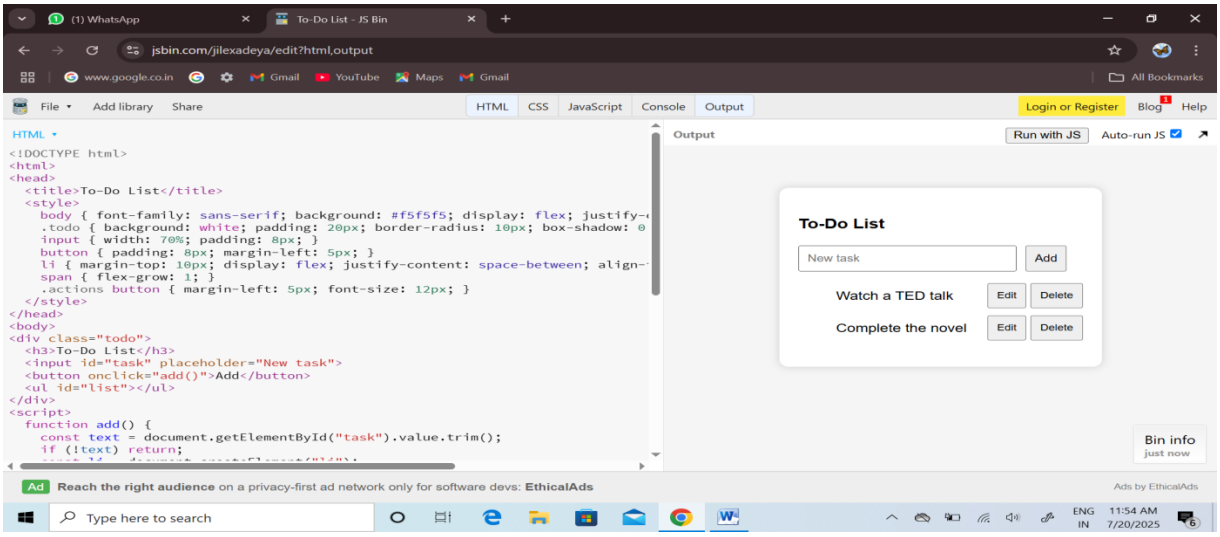



```
<!DOCTYPE html>

<html>

<head>

  <title>To-Do List</title>

  <style>

    body { font-family: sans-serif; background: #f5f5f5; display: flex; justify-content: center; padding: 50px; }

    .todo { background: white; padding: 20px; border-radius: 10px; box-shadow: 0 0 10px #ccc; width: 300px; }

    input { width: 70%; padding: 8px; }

    button { padding: 8px; margin-left: 5px; }

    li { margin-top: 10px; display: flex; justify-content: space-between; align-items: center; }

    span { flex-grow: 1; }

    .actions button { margin-left: 5px; font-size: 12px; }

  </style>

</head>

<body>

<div class="todo">
```

```
  <h3>To-Do List</h3>
  <input id="task" placeholder="New task">
  <button onclick="add()">Add</button>
  <ul id="list"></ul>
</div>
<script>
  function add() {
    const text = document.getElementById("task").value.trim();
    if (!text) return;
    const li = document.createElement("li");
    const span = document.createElement("span");
    span.textContent = text;

    const actions = document.createElement("div");
    actions.className = "actions";

    const edit = document.createElement("button");
    edit.textContent = "Edit";
    edit.onclick = () => {
      const newText = prompt("Edit task", span.textContent);
      if (newText) span.textContent = newText.trim();
    };

    const del = document.createElement("button");
    del.textContent = "Delete";
```

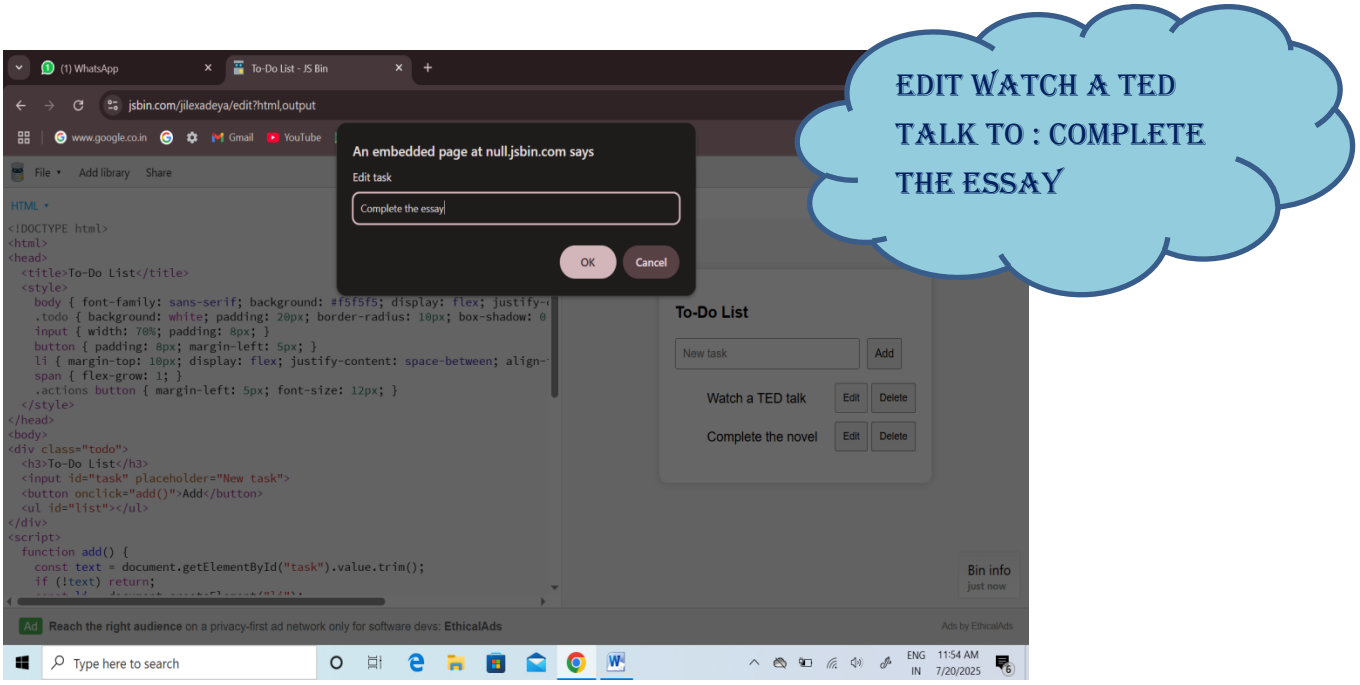
```
    del.onclick = () => li.remove();


    actions.append(edit, del);
    li.append(span, actions);
    document.getElementById("list").appendChild(li);
    document.getElementById("task").value = '';
  }
</script>
</body>
</html>
```

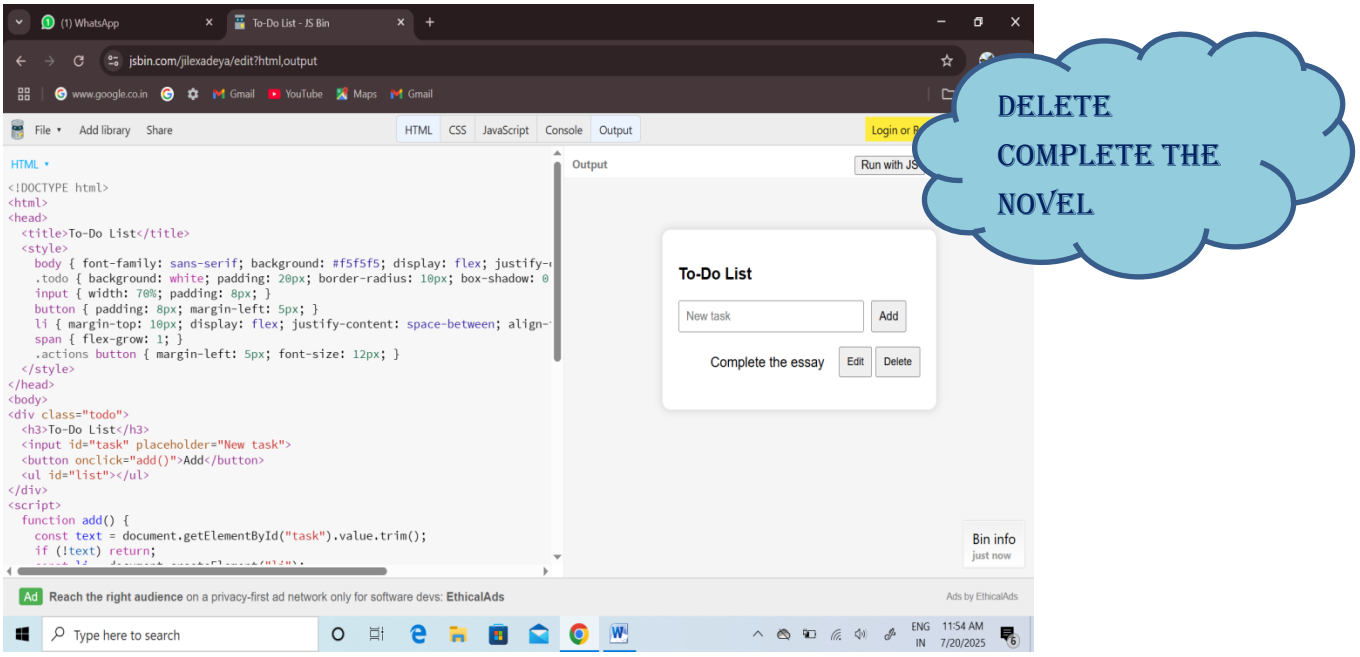
OUTPUT SCREENSHOT



ADD 2 TO-DO LIST NAMELY:

1. WATCH A TED TALK
2. COMPLETE THE NOVEL





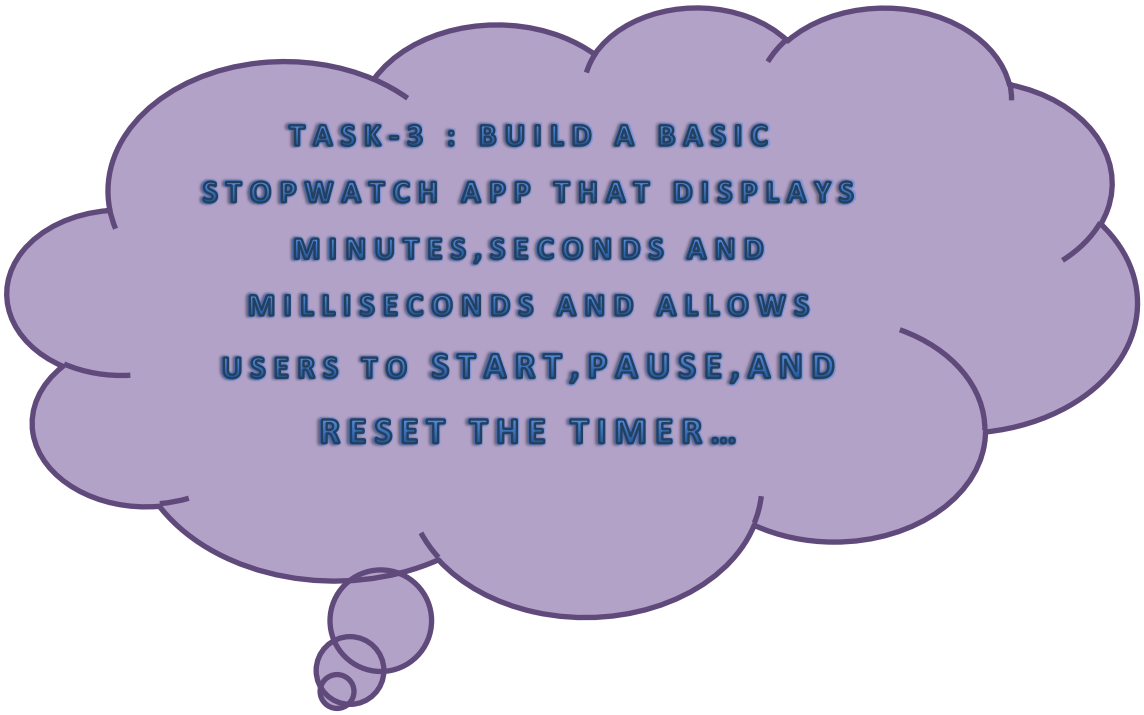
This To-Do List app is built using HTML, CSS, and JavaScript.

Users can add tasks using the input field and "Add" button.

Each task has "Edit" and "Delete" options for updates and removal.

The design is simple and user-friendly, ideal for daily task tracking.

DESCRIPTION

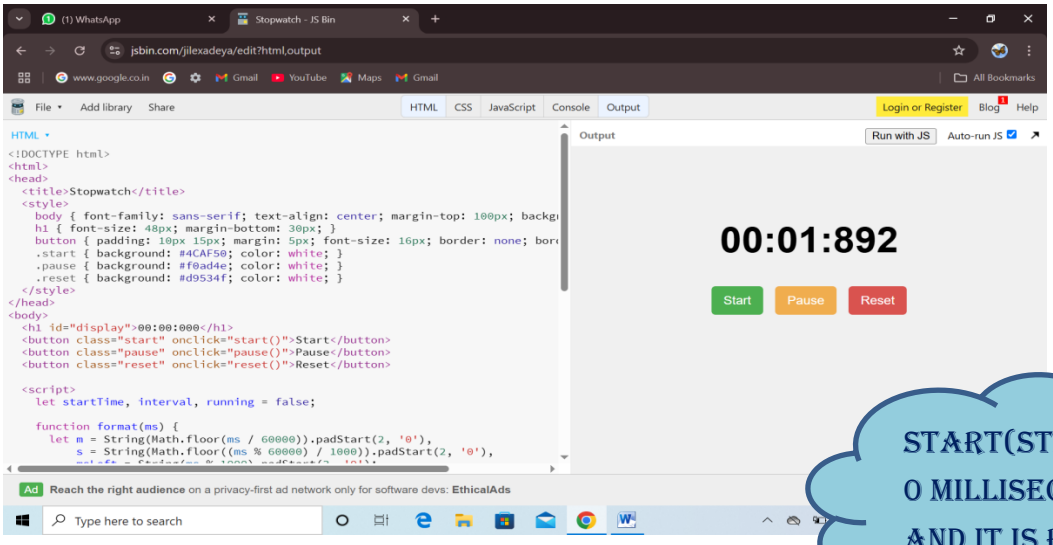



```
<!DOCTYPE html>

<html>

<head>

  <title>Stopwatch</title>

  <style>

    body { font-family: sans-serif; text-align: center; margin-top: 100px; background: #f0f0f0; }

    h1 { font-size: 48px; margin-bottom: 30px; }

    button { padding: 10px 15px; margin: 5px; font-size: 16px; border: none; border-radius: 5px; }

    .start { background: #4CAF50; color: white; }

    .pause { background: #f0ad4e; color: white; }

    .reset { background: #d9534f; color: white; }

  </style>

</head>

<body>
```

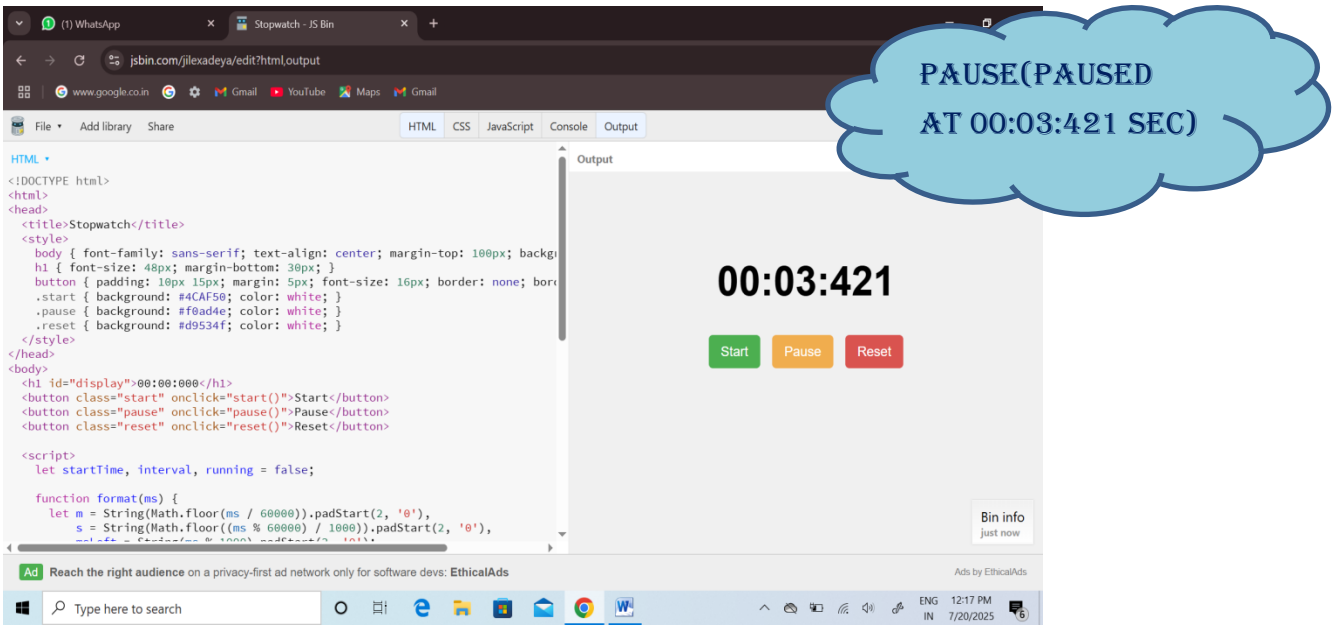
```
<h1 id="display">00:00:000</h1>
<button class="start" onclick="start()">Start</button>
<button class="pause" onclick="pause()">Pause</button>
<button class="reset" onclick="reset()">Reset</button>

<script>
  let startTime, interval, running = false;

  function format(ms) {
    let m = String(Math.floor(ms / 60000)).padStart(2, '0'),
        s = String(Math.floor((ms % 60000) / 1000)).padStart(2, '0'),
        msLeft = String(ms % 1000).padStart(3, '0');
    return `${m}:${s}:${msLeft}`;
  }

  function start() {
    if (running) return;
    running = true;
    startTime = Date.now() - (window.elapsed || 0);
    interval = setInterval(() => {
      window.elapsed = Date.now() - startTime;
      document.getElementById("display").textContent = format(window.elapsed);
    }, 10);
  }
```

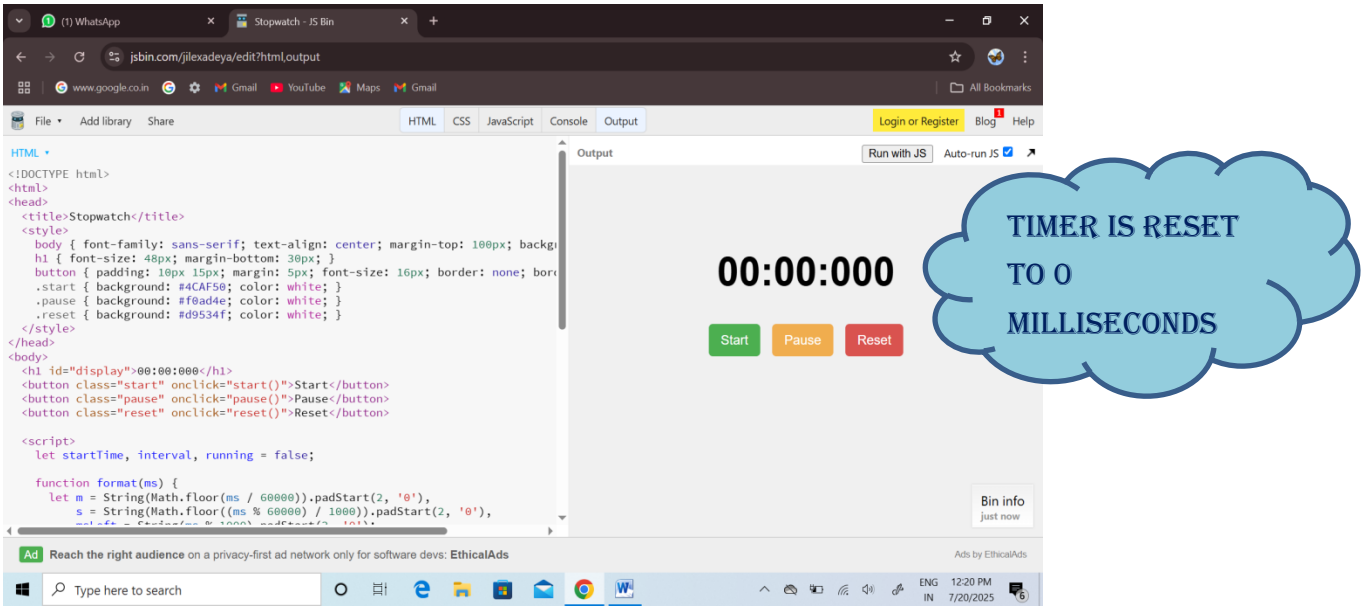
```
    function pause() {
      clearInterval(interval);
      running = false;
    }

    function reset() {
      clearInterval(interval);
      running = false;
      window.elapsed = 0;
      document.getElementById("display").textContent = "00:00:000";
    }
  </script>
</body>
</html>
```

OUTPUT SCREENSHOT



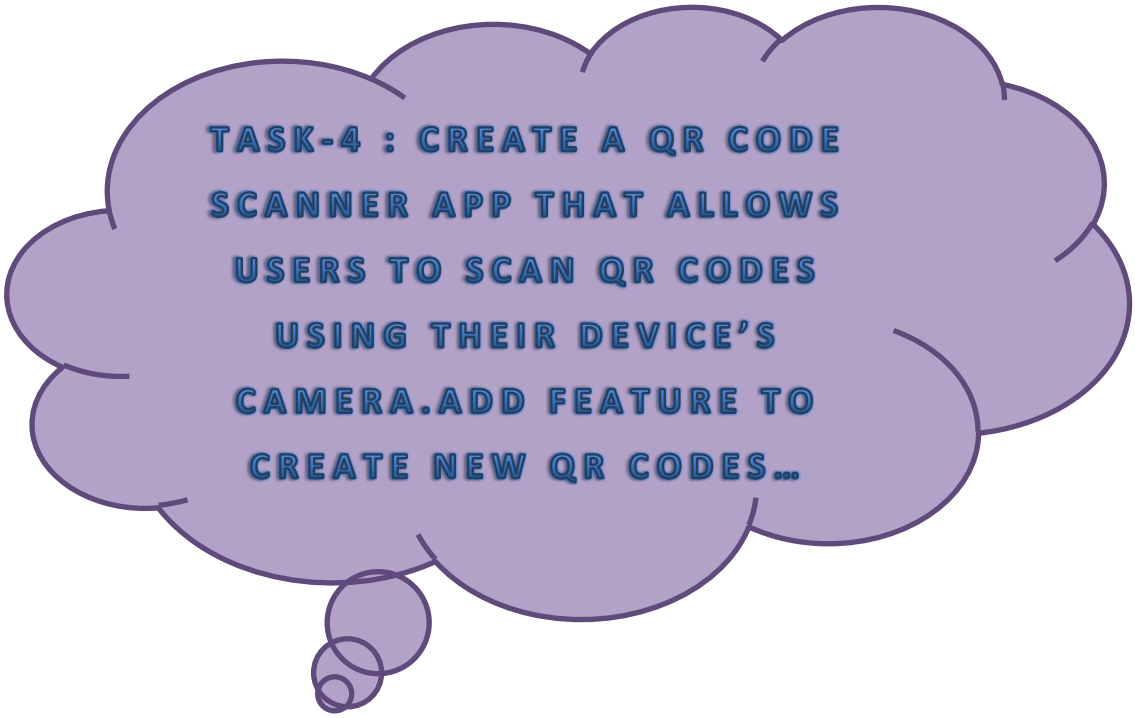
START(STARTED AT 0 MILLISECONDS AND IT IS RUNNING)





DESCRIPTION

This is a simple stopwatch web app built using HTML, CSS, and JavaScript. It displays time in minutes, seconds, and milliseconds (MM:SS:MS) format. Users can start, pause, and reset the timer using buttons. The time updates every 10 milliseconds for accurate tracking.




```
<!DOCTYPE html>
<html>
<head>
  <title>QR Scanner & Generator</title>
  <script src="https://unpkg.com/html5-qrcode" type="text/javascript"></script>
  <script src="https://cdn.jsdelivr.net/npm/qrcode/build/qrcode.min.js"></script>
  <style>
    body { font-family: sans-serif; text-align: center; background: #f4f4f4; padding: 20px; }
    h1 { margin-bottom: 20px; }
    #reader, #image-reader { width: 300px; margin: auto; }
    #qr-output, #scanner-result { margin-top: 20px; word-break: break-word; }
    input, button, textarea {
      padding: 10px; margin: 5px; font-size: 16px;
      width: 90%; max-width: 400px;
    }
```

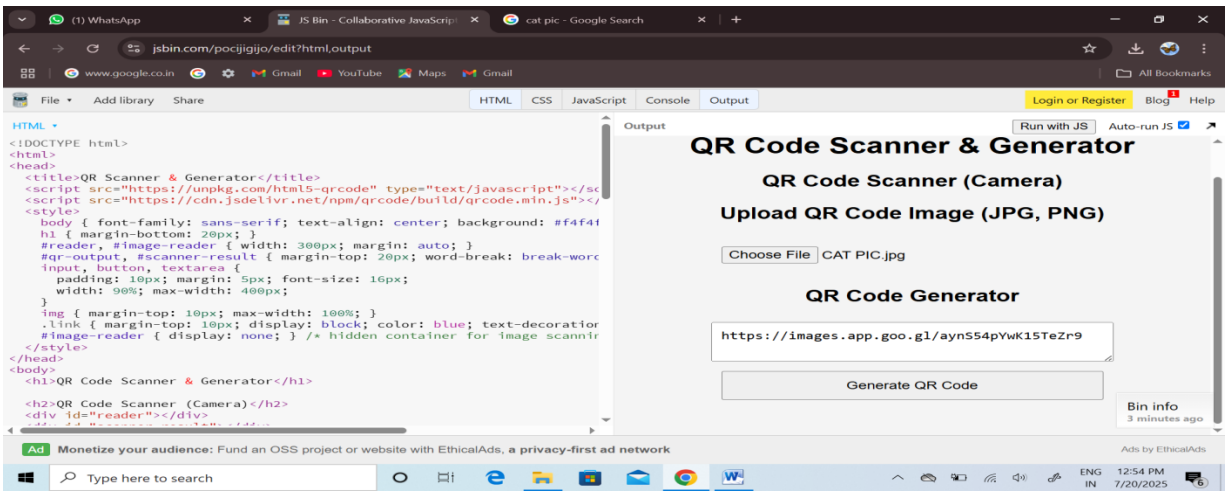
```
    img { margin-top: 10px; max-width: 100%; }
    .link { margin-top: 10px; display: block; color: blue; text-decoration: underline; }
    #image-reader { display: none; } /* hidden container for image scanning */
  </style>
</head>
<body>
  <h1>QR Code Scanner & Generator</h1>

  <h2>QR Code Scanner (Camera)</h2>
  <div id="reader"></div>
  <div id="scanner-result"></div>

  <h2>Upload QR Code Image (JPG, PNG)</h2>
  <input type="file" accept="image/*" onchange="scanImageFile(this)">
  <div id="image-reader"></div>

  <h2>QR Code Generator</h2>
  <textarea id="text" placeholder="Enter text or URL"></textarea><br>
  <button onclick="generateQRCode()">Generate QR Code</button>
  <div id="qr-output"></div>

  <script>
    // Camera-based scanner
    const scanner = new Html5Qrcode("reader");
    scanner.start(
```

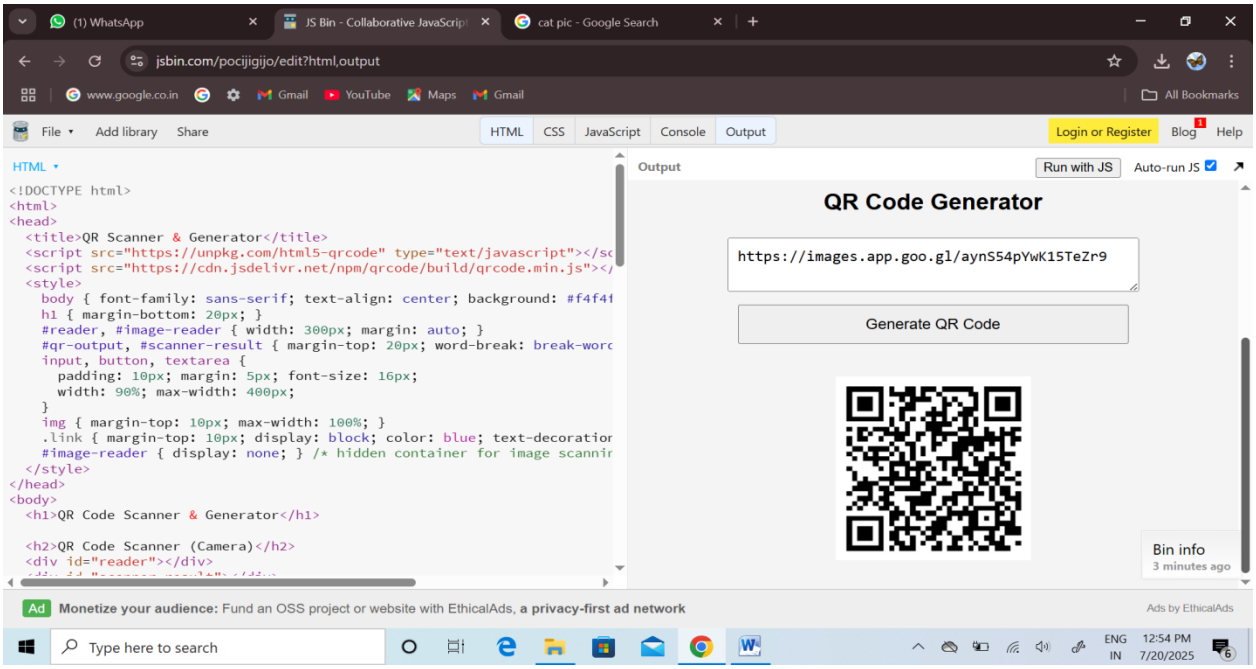
```
      { facingMode: "environment" },
      { fps: 10, qrbox: 250 },
      (decodedText) => {
        displayScannedText(decodedText);
        scanner.stop(); // Stop after successful scan
      },
      (error) => { /* ignore continuous scan errors */ }
    );

    // Display result with link if valid
    function displayScannedText(text) {
      const resultDiv = document.getElementById('scanner-result');
      resultDiv.innerHTML = `<b>Scanned Result:</b> ${text}`;
      if (text.startsWith("http://") || text.startsWith("https://")) {
        resultDiv.innerHTML += `<br><a class="link" href="${text}" target="_blank">Open
Link</a>`;
      }
    }

    // Upload and scan QR image (JPG/PNG)
    function scanImageFile(input) {
      if (!input.files.length) return;
      const file = input.files[0];
      const imageScanner = new Html5Qrcode("image-reader");
```

```
      scanner.stop().then(() => {
        // Safe to scan image now
        imageScanner.scanFile(file, true)
          .then(decodedText => {
            displayScannedText(decodedText);
            imageScanner.clear();
          })
          .catch(err => {
            document.getElementById('scanner-result').innerHTML = "Could not detect QR code in the image.";
          });
      }).catch(() => {
        // Even if stop fails, try to scan image anyway
        imageScanner.scanFile(file, true)
          .then(decodedText => {
            displayScannedText(decodedText);
            imageScanner.clear();
          })
          .catch(err => {
            document.getElementById('scanner-result').innerHTML = "Could not detect QR code in the image.";
          });
      });
    }

    // Generate QR from text/URL
```

```
    function generateQRCode() {
      const text = document.getElementById("text").value.trim();
      const output = document.getElementById("qr-output");
      if (!text) {
        output.innerHTML = "Please enter text to generate QR code.";
        return;
      }
      output.innerHTML = "";
      QRCode.toDataURL(text, { width: 200, margin: 2 }, (err, url) => {
        if (err) return output.innerHTML = "Error generating QR Code.";
        output.innerHTML = `<img src="${url}" alt="QR Code">`;
      });
    }
  </script>
</body>
</html>
```

OUTPUT SCREENSHOT





DESCRIPTION

This web application allows users to scan QR codes using their device's camera or by uploading QR code images (in JPG, PNG, etc. formats). It also provides a feature to generate new QR codes from user input (text or URLs). Built using HTML, CSS, and JavaScript, the app leverages open-source libraries like html5-qrcode for scanning and qrcode.js for generation.