Project Report: Stopwatch Web Application

1. Introduction

A stopwatch application is a practical and essential tool for time tracking and management. It allows users to measure time intervals accurately, making it useful for various tasks such as workouts, productivity tracking, and experiments. This project focuses on creating a responsive and visually engaging stopwatch web application using HTML, CSS, and JavaScript.

2. Project Description

The stopwatch application includes features such as start, stop, reset, and lap recording. Users can interact with the stopwatch through intuitive buttons. The timer is displayed in hours, minutes, seconds, and milliseconds. The design includes a visually appealing interface with gradient elements such as circles and half-circle shapes for aesthetics.

3. Methodology

The application was developed using:

- HTML to structure the user interface (buttons, display screen, lap records)
- CSS to style the elements and add visual gradients and effects
- JavaScript to implement the functionality of time tracking and user interactions

Key functions implemented:

- startTimer(): Starts the timer
- pauseTimer(): Pauses the timer
- resetTimer(): Resets the timer
- lapTime(): Captures the current time without stopping the timer

4. Source Code

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

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

  <title>Stopwatch</title>

  <style>

    body {

      font-family: Arial, sans-serif;

      background: linear-gradient(to right, #83a4d4, #b6fbff);

      display: flex;

      justify-content: center;

      align-items: center;

      height: 100vh;

      flex-direction: column;

    }

    .stopwatch {

      background: white;

      padding: 30px;

      border-radius: 20px;

      box-shadow: 0 0 20px rgba(0,0,0,0.2);

      text-align: center;

    }

    .display {

      font-size: 2em;
      margin-bottom: 20px;
    }
    .buttons button {
      padding: 10px 20px;
      margin: 5px;
      border: none;
      border-radius: 5px;
      background-color: #007bff;
      color: white;
      cursor: pointer;
    }
    .laps {
      margin-top: 15px;
      text-align: left;
    }
  </style>
</head>
<body>
  <div class="stopwatch">
    <div class="display" id="display">00:00:00.000</div>
    <div class="buttons">
      <button onclick="startTimer()">Start</button>
      <button onclick="pauseTimer()">Pause</button>
      <button onclick="resetTimer()">Reset</button>
      <button onclick="lapTime()">Lap</button>
    </div>
  <div class="laps" id="laps"></div>

</div>


<script>

  let startTime = 0;

  let elapsedTime = 0;

  let timerInterval;


  function timeToString(time) {

    let diffInHrs = time / 3600000;

    let hh = Math.floor(diffInHrs);


    let diffInMin = (diffInHrs - hh) * 60;

    let mm = Math.floor(diffInMin);


    let diffInSec = (diffInMin - mm) * 60;

    let ss = Math.floor(diffInSec);


    let diffInMs = (diffInSec - ss) * 1000;

    let ms = Math.floor(diffInMs);


    let formattedHH = hh.toString().padStart(2, "0");

    let formattedMM = mm.toString().padStart(2, "0");

    let formattedSS = ss.toString().padStart(2, "0");

    let formattedMS = ms.toString().padStart(3, "0");
  return `${formattedHH}:${formattedMM}:${formattedSS}.${formattedMS}`;
}

function startTimer() {
  startTime = Date.now() - elapsedTime;
  timerInterval = setInterval(function printTime() {
    elapsedTime = Date.now() - startTime;
    document.getElementById("display").innerHTML = timeToString(elapsedTime);
  }, 10);
}

function pauseTimer() {
  clearInterval(timerInterval);
}

function resetTimer() {
  clearInterval(timerInterval);
  document.getElementById("display").innerHTML = "00:00:00.000";
  document.getElementById("laps").innerHTML = "";
  elapsedTime = 0;
}

function lapTime() {
  const lap = document.createElement("div");
  lap.textContent = timeToString(elapsedTime);
  document.getElementById("laps").appendChild(lap);
```

```
    }
  </script>
</body>
</html>
```

5. Output

The application displays:

- A timer showing 00:00:00.000

- Four buttons: Start, Pause, Reset, Lap

- Lap times appearing below the stopwatch when Lap is clicked

- Visually appealing gradient background and styled interface with smooth interaction

6. Conclusion

The stopwatch web application successfully demonstrates how HTML, CSS, and JavaScript can be combined to build an interactive time management tool. The project improves understanding of DOM manipulation, event handling, and dynamic content rendering. The addition of gradient shapes and a responsive layout makes the application user-friendly and modern.

This project can be further enhanced by adding features such as:

- Exporting lap times

- Dark/light theme toggle

- Sound alerts