



Name: Saad Rehman

Section: 6E

Student I'd: 21F-9640

Assignment: 03

Course: Web Programming

Topic: Js, JQuery

Department of Computer Science

Semester	Spring 2024
-----------------	--------------------

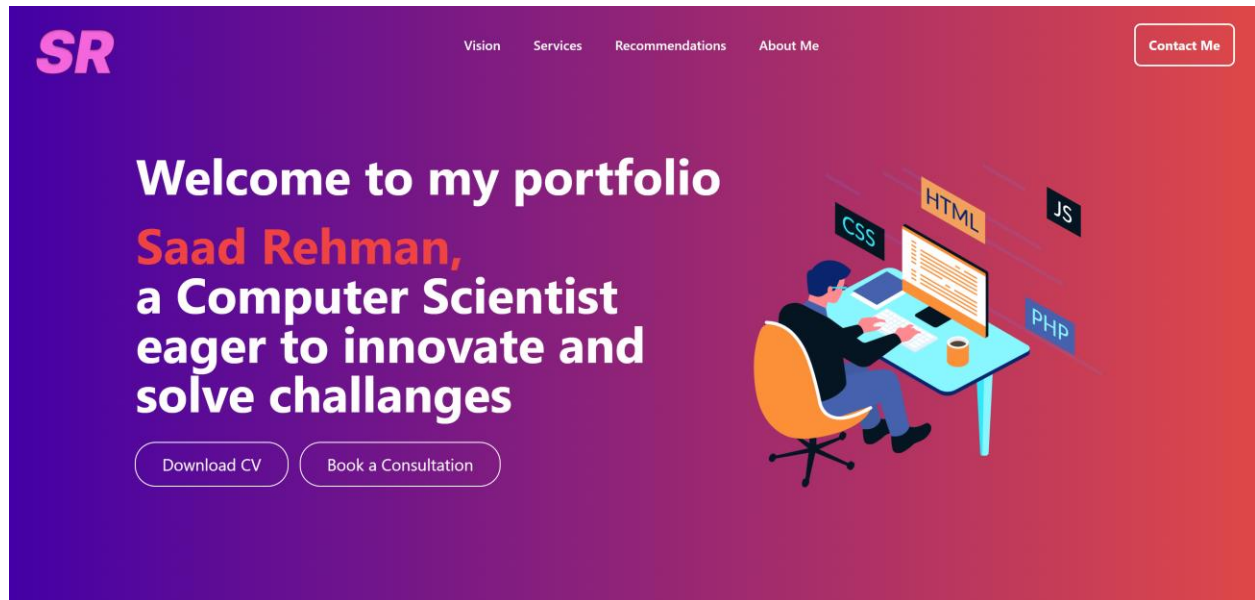
GitHub Link:

[Click here to open the link.](#)

Task 1

<https://saadrehman-portfolio.vercel.app/>

Code Repository



Task 2

Task2.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Task 2</title>
    <link
      href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&family=
Roboto:wght@400;500&display=swap"
      rel="stylesheet"
    />
    <link
      rel="stylesheet"
      href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.
css"
    />
```

```

    <link rel="stylesheet" href="Task2.css" />
</head>
<body onload="getLocation()">
  <div class="container">
    <h1 class="text-center mb-4">Weather App</h1>
    <div class="search-bar form-inline justify-content-center mb-4">
      <input
        type="text"
        id="search-city"
        class="form-control mr-2"
        placeholder="Enter City..."
      />
      <button class="btn btn-primary" onclick="WeatherDetails()">
        Search
      </button>
    </div>
    <div id="weather-info"></div>
    <div id="hourly-forecast"></div>
    <div id="daily-forecast"></div>
  </div>

  <script src="Task2.js"></script>

  <footer class="text-center mt-4">
    <p>Developed by Saad Rehman</p>
  </footer>
</body>
</html>

```

Task2.css

```

body {
  background: linear-gradient(to right, rgba(0, 0, 0, 0.6), rgba(0, 0, 0, 0.4)), url('background.jpeg') no-repeat center center fixed;
  background-size: cover;
  font-family: 'Roboto', sans-serif;
  color: #fff;
}

h1, h2, h3, h4, h5, h6 {
  font-family: 'Poppins', sans-serif;
  margin: 0.5em 0;
  color: #f1f1f1;
}

```

```
.search-bar input, .search-bar button {
  border: none;
  margin: 5px;
  border-radius: 30px;
  padding: 10px 15px;
}

.search-bar button {
  background-color: #007bff;
  color: white;
  padding: 10px 20px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.search-bar button:hover {
  background-color: #0056b3;
}

.container, .forecast-item {
  background-color: rgba(255, 255, 255, 0.2);
  backdrop-filter: blur(8px);
  border-radius: 15px;
  padding: 20px;
}

#weather-info, #hourly-forecast, #daily-forecast {
  text-align: center;
  display: flex;
  flex-direction: column;
  align-items: center;
}

.forecast-item {
  border: 1px solid #ffffff50;
  margin: 10px;
  padding: 15px;
  border-radius: 10px;
  transition: all 0.3s ease-in-out;
}

.forecast-item:hover {
  transform: scale(1.05);
  border-color: #ffffff;
}
```

```

}

#hourly-forecast > h4, #daily-forecast > h4 {
  color: #ade8f4;
  font-weight: 600;
}

.footer {
  text-align: center;
  font-size: 0.9rem;
  margin-top: 20px;
}

.current-weather img {
  width: 100px;
  height: auto;
  margin-top: 10px;
}

body {
  background-image: url('image.jpg');
  background-size: cover;
}

```

Task2.js

```

function debounce(func, wait) {
  let timeout;
  return function executedFunction(...args) {
    const later = () => {
      clearTimeout(timeout);
      func(...args);
    };
    clearTimeout(timeout);
    timeout = setTimeout(later, wait);
  };
}

function WeatherDetails() {
  const city = document.getElementById("search-city").value;
  if (city.length > 3) {
    const apiKey = "69fa95c453084c6862ae54f6a212636c";

```

```

        fetchCurrentWeather(city, apiKey);
        fetchForecast(city, apiKey);

document.getElementById("weather-info").classList.add("visible");

    }
}

const debouncedWeatherDetails = debounce(WeatherDetails, 800);

document.getElementById("search-city").addEventListener("input",
debouncedWeatherDetails);

function fetchCurrentWeather(city, apiKey) {
    const url =
`https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=${apiKey}&units=
metric`;

    const httpRequest = new XMLHttpRequest();
    httpRequest.onreadystatechange = function () {
        if (this.readyState === 4 && this.status === 200) {
            const data = JSON.parse(this.responseText);
            const iconUrl =
`http://openweathermap.org/img/wn/${data.weather[0].icon}.png`; // Constructing
the URL for the icon
            document.getElementById("weather-info").innerHTML = `
                <div class="current-weather mb-4 text-center">
                    <h1>${data.name}</h1>
                     <!-- Weather icon -->
                    <h2>${data.main.temp}°C</h2>
                    <p>Description: ${data.weather[0].description}</p>
                    <p>Humidity: ${data.main.humidity}%</p>
                    <p>Wind Speed: ${data.wind.speed} m/s</p>
                </div>
            `;
        }
    };
    httpRequest.open("GET", url, true);
    httpRequest.send();
}

function fetchForecast(city, apiKey) {

```

```

    const url =
`https://api.openweathermap.org/data/2.5/forecast?q=${city}&appid=${apiKey}&units
=metric`;
    const httpRequest = new XMLHttpRequest();
    httpRequest.onreadystatechange = function () {
        if (this.readyState === 4 && this.status === 200) {
            const data = JSON.parse(this.responseText);
            const hourlyHTML = data.list
                .slice(0, 5)
                .map(
                    (forecast) => `
                        <div class="forecast-item">
                            <p>${new Date(forecast.dt_txt).toLocaleTimeString([], {
                                hour: "2-digit",
                                minute: "2-digit",
                            }): ${forecast.main.temp}°C, ${
                                forecast.weather[0].description
                            }</p>
                        </div>
                    `
                )
                .join("");

            document.getElementById(
                "hourly-forecast"
            ).innerHTML = `<h4>Next Today</h4>${hourlyHTML}`;

            const dailyHTML = data.list
                .filter((_, index) => index % 8 === 0)
                .slice(0, 5)
                .map(
                    (forecast) => `
                        <div class="forecast-item">
                            <p>${new Date(forecast.dt_txt).toLocaleDateString(): ${
                                forecast.main.temp
                            }°C, ${forecast.weather[0].description}</p>
                        </div>
                    `
                )
                .join("");

            document.getElementById(
                "daily-forecast"
            ).innerHTML = `<h4>Next 5 Days</h4>${dailyHTML}`;
        }
    }

```

```

    };
    httpRequest.open("GET", url, true);
    httpRequest.send();
  }

  async function fetchWeather(lat, lon) {
    const apiKey = "dd39413e316a2ab84efa29d9fb1f8bd1";
    const url =
      `https://api.openweathermap.org/data/2.5/weather?lat=${lat}&lon=${lon}&appid=${ap
        iKey}&units=metric`;

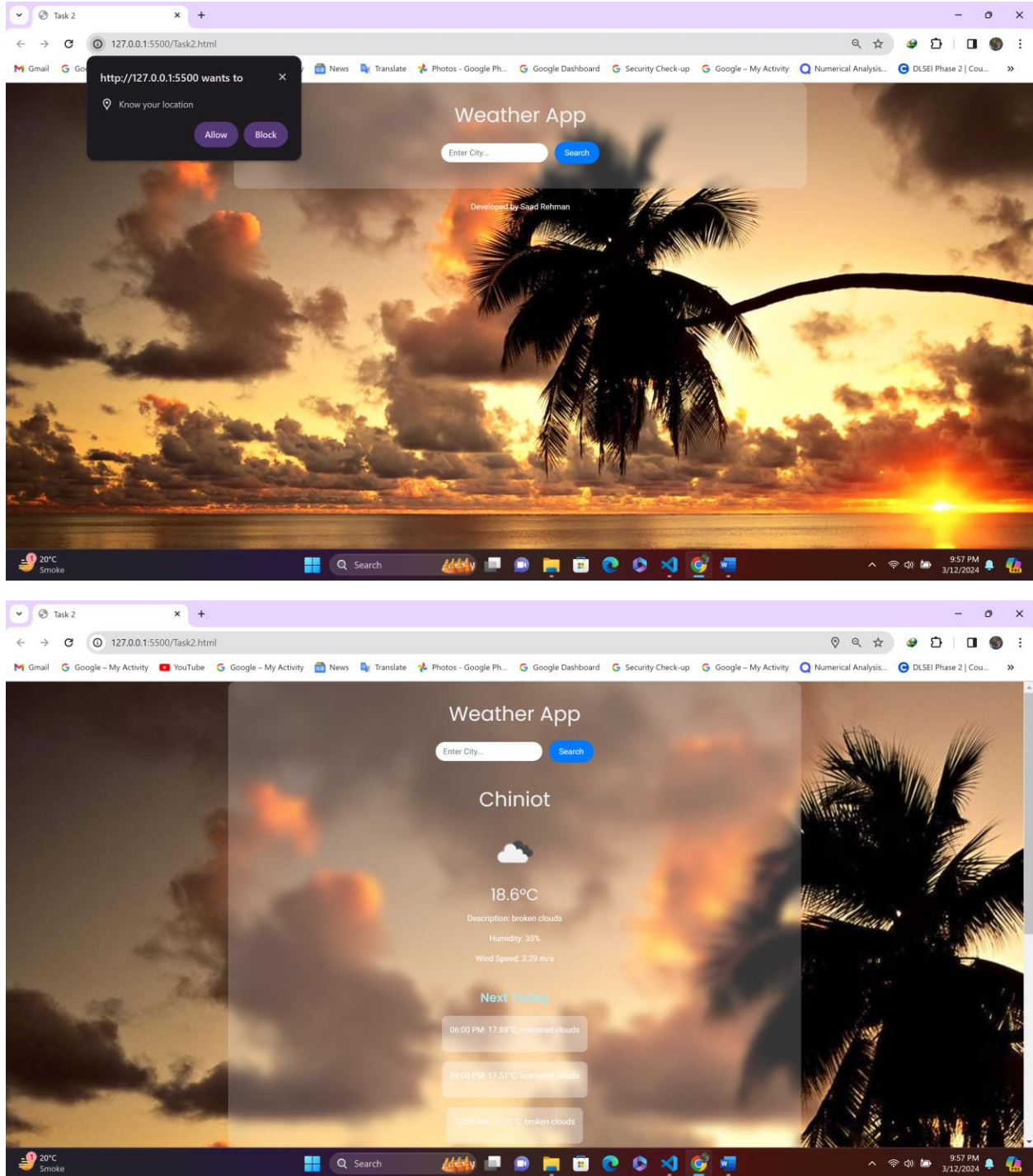
    const response = await fetch(url);
    const data = await response.json();
    fetchCurrentWeather(data.name, apiKey);
    fetchForecast(data.name, apiKey);
  }

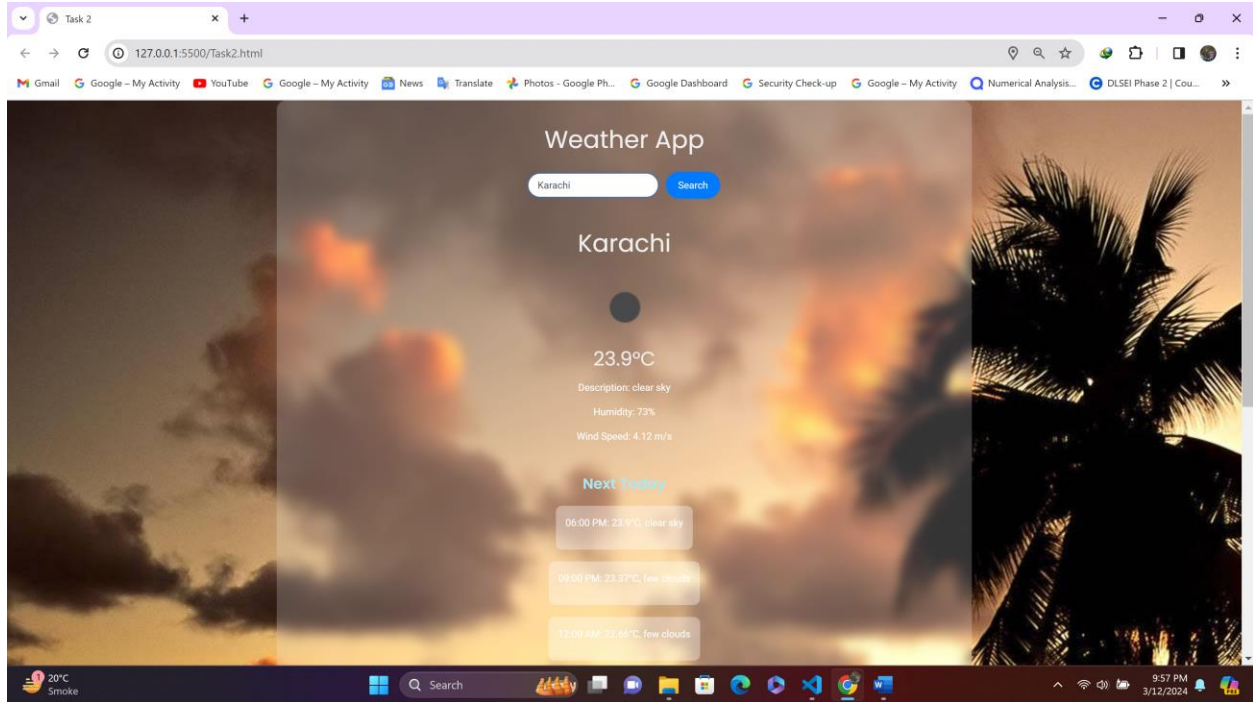
  function getLocation() {
    if (navigator.geolocation) {
      navigator.geolocation.getCurrentPosition(showPosition, null);
    }
  }

  function showPosition(position) {
    var latitude = position.coords.latitude;
    var longitude = position.coords.longitude;
    fetchWeather(latitude, longitude);
  }

```

Output





Task 3

Task3.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Task 3</title>
  <link rel="stylesheet" href="Task3.css">
</head>
<body>
  <div class="container">
    <div id="welcome-screen" class="welcome-screen">
      <h1>Welcome to the Digital Clock App</h1>
      <p>Click the button below to view the current time.</p>
      <button id="view-time-btn" class="btn">View Time</button>
    </div>

    <div id="clock-screen" class="clock-screen" style="display: none;">
```

```

<div id="time-status" class="time-status">Current Running Time:</div>
<div class="clock">
  <div class="display">
    <div class="digit" id="hours">00</div>
    <div class="separator">:</div>
    <div class="digit" id="minutes">00</div>
    <div class="separator">:</div>
    <div class="digit" id="seconds">00</div>
  </div>
</div>
</div>
</div>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<script src="Task3.js"></script>
</body>
</html>

```

Task3.css

```

body {
  font-family: Arial, sans-serif;
  background-color: #222;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
  margin: 0;
}

.container {
  text-align: center;
}

.welcome-screen, .clock-screen {
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}

.welcome-screen h1, .welcome-screen p {
  color: #fff;
}

```

```
.btn {
  padding: 10px 20px;
  font-size: 1.2rem;
  cursor: pointer;
  background-color: #007bff;
  color: #fff;
  border: none;
  border-radius: 5px;
  transition: background-color 0.3s ease;
}

.btn:hover {
  background-color: #0056b3;
}

.clock {
  display: inline-block;
  background-color: #333;
  border-radius: 10px;
  padding: 20px;
  box-shadow: 0 0 20px rgba(0, 0, 0, 0.3);
  transform: rotateX(20deg); /* Apply 3D rotation */
}

.display {
  display: flex;
  align-items: center;
  justify-content: center;
}

.digit {
  font-size: 3rem;
  color: #fff;
  padding: 10px;
  border-radius: 5px;
  margin: 0 5px;
  background-color: #555;
}

.separator {
  font-size: 3rem;
  color: #fff;
  margin: 0 5px;
}
```

```
.time-status {
  font-size: 1.2rem;
  color: #fff;
  margin-bottom: 10px;
}
```

Task3.js

```
$(document).ready(function() {
  $("#view-time-btn").click(function() {
    $("#welcome-screen").fadeOut(500); // Fade out the welcome screen
    $("#clock-screen").fadeIn(1000); // Fade in the clock screen

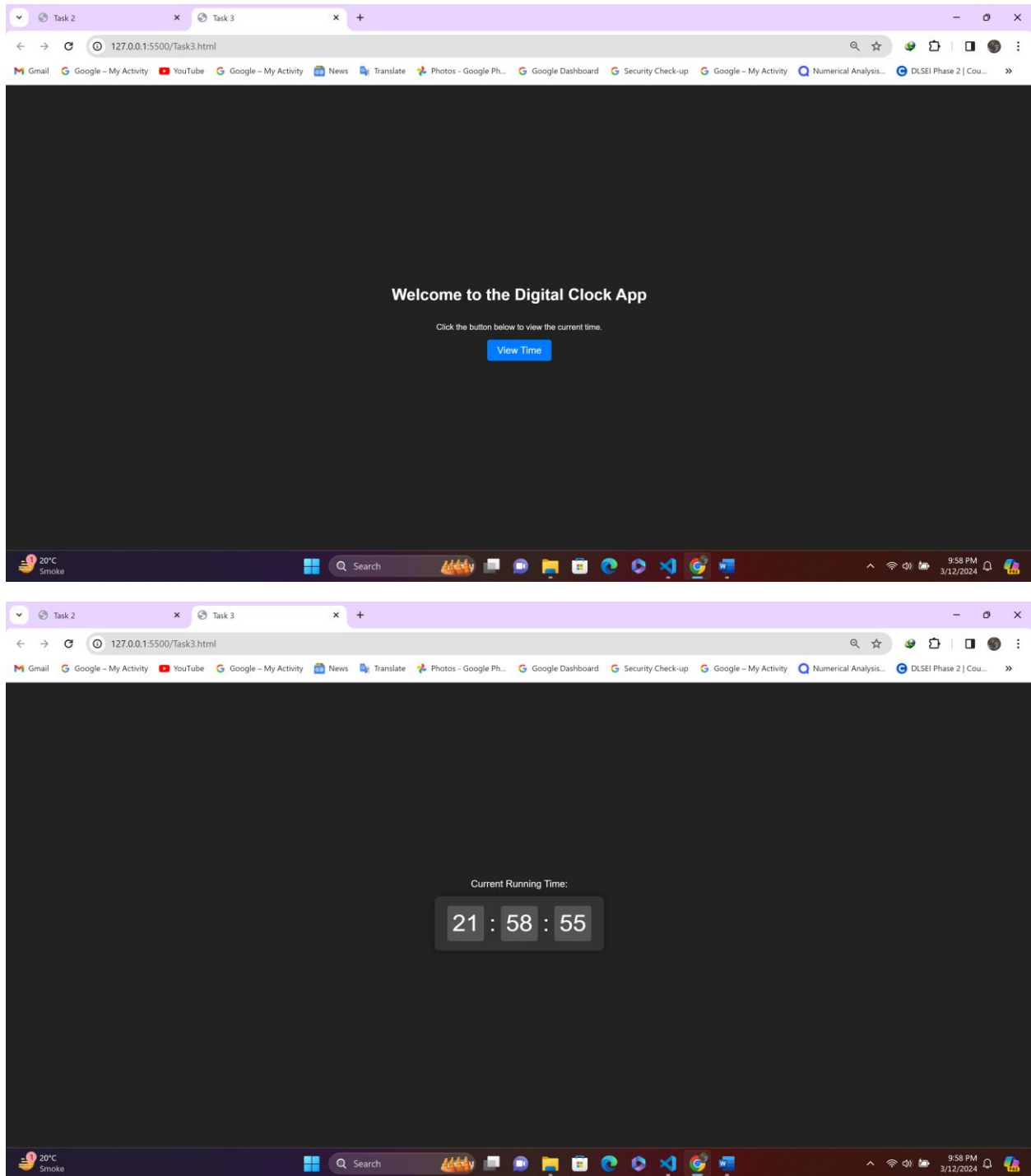
    updateTime(); // Update the time immediately
    setInterval(updateTime, 1000); // Update the time every second
  });

  function updateTime() {
    var now = new Date();
    var hours = formatTime(now.getHours());
    var minutes = formatTime(now.getMinutes());
    var seconds = formatTime(now.getSeconds());

    $("#hours").text(hours);
    $("#minutes").text(minutes);
    $("#seconds").text(seconds);
  }

  function formatTime(time) {
    return time < 10 ? "0" + time : time;
  }
});
```

Output



Task 4

Task4.html

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```

<title>Task 4</title>
<link rel="stylesheet" href="Task4.css">
</head>
<body>
  <div id="start-screen" class="container">
    <h1>Welcome to Rock Paper Scissors!</h1>
    <p>Enter your name and click start to begin playing.</p>
    <form id="start-form">
      <input type="text" id="player-name" class="input-field" placeholder="Enter
your name" autocomplete="off">
      <label for="player-name" class="input-label">Your Name</label>
      <button type="submit" class="btn">Start</button>
    </form>
  </div>

  <div id="game-screen" class="container" style="display: none;">
    <h1>Rock Paper Scissors</h1>
    <div class="options">
      
      
      
    </div>
    <div class="result-container">
      <div id="result" class="result"></div>
      <div id="score">Score: <span id="score-value">0</span></div>
    </div>
  </div>

  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
  <script src="Task4.js"></script>
</body>
</html>

```

Task4.css

```

body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
  background-color: #f3f3f3;
  perspective: 1000px;
}

.container {

```

```
max-width: 800px;
margin: 20px auto;
text-align: center;
padding: 50px;
}

h1 {
  margin-bottom: 20px;
}

p {
  margin-bottom: 30px;
}

.input-field {
  padding: 10px;
  font-size: 16px;
  margin-bottom: 10px;
  width: 100%;
  border: 2px solid #ddd;
  border-radius: 5px;
  transition: border-color 0.3s ease;
}

.input-field:focus {
  outline: none;
  border-color: #007bff;
}

.input-label {
  display: none;
}

.btn {
  padding: 10px 20px;
  font-size: 16px;
  cursor: pointer;
  background-color: #007bff;
  color: #fff;
  border: none;
  border-radius: 5px;
  transition: background-color 0.3s ease;
}

.btn:hover {
```



```

    background-color: #0056b3;
}

.options {
    margin-bottom: 20px;
}

.choice {
    width: 150px;
    height: 150px;
    margin: 0 10px;
    cursor: pointer;
    border-radius: 50%;
    transition: all 0.3s ease;
    transform-style: preserve-3d;
}

.choice:hover {
    transform: scale(1.1) rotateY(10deg);
}

.result-container {
    margin-top: 30px;
}

.result {
    font-size: 24px;
    font-weight: bold;
    margin-bottom: 20px;
}

#score {
    font-size: 18px;
    font-weight: bold;
    color: #333;
}

#score-value {
    color: #007bff;
}

```

Task4.js

```
$(document).ready(function() {
```

```

$("#start-form").submit(function(event) {
    event.preventDefault();
    var playerName = $("#player-name").val().trim();
    if (playerName !== "") {
        $("#start-screen").hide();
        $("#game-screen").show();
        $("#game-screen h1").text("Hello, " + playerName + "! Let's play Rock
Paper Scissors!");
    }
});

var choices = ["rock", "paper", "scissors"];
var score = 0;

$(".choice").click(function() {
    var userChoice = $(this).attr("id");
    var computerChoice = choices[Math.floor(Math.random() * choices.length)];

    var result = getResult(userChoice, computerChoice);
    updateScore(result);

    $("#result").text("You chose " + userChoice + ". Computer chose " +
computerChoice + ". " + result);
});

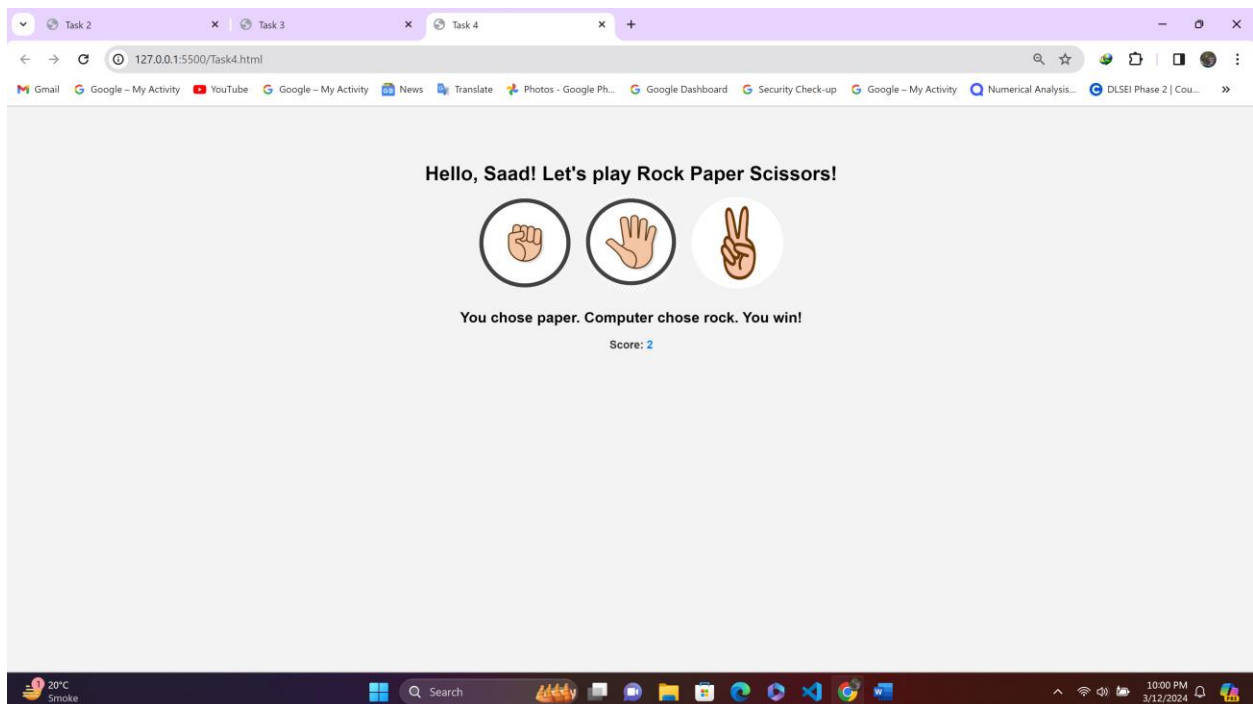
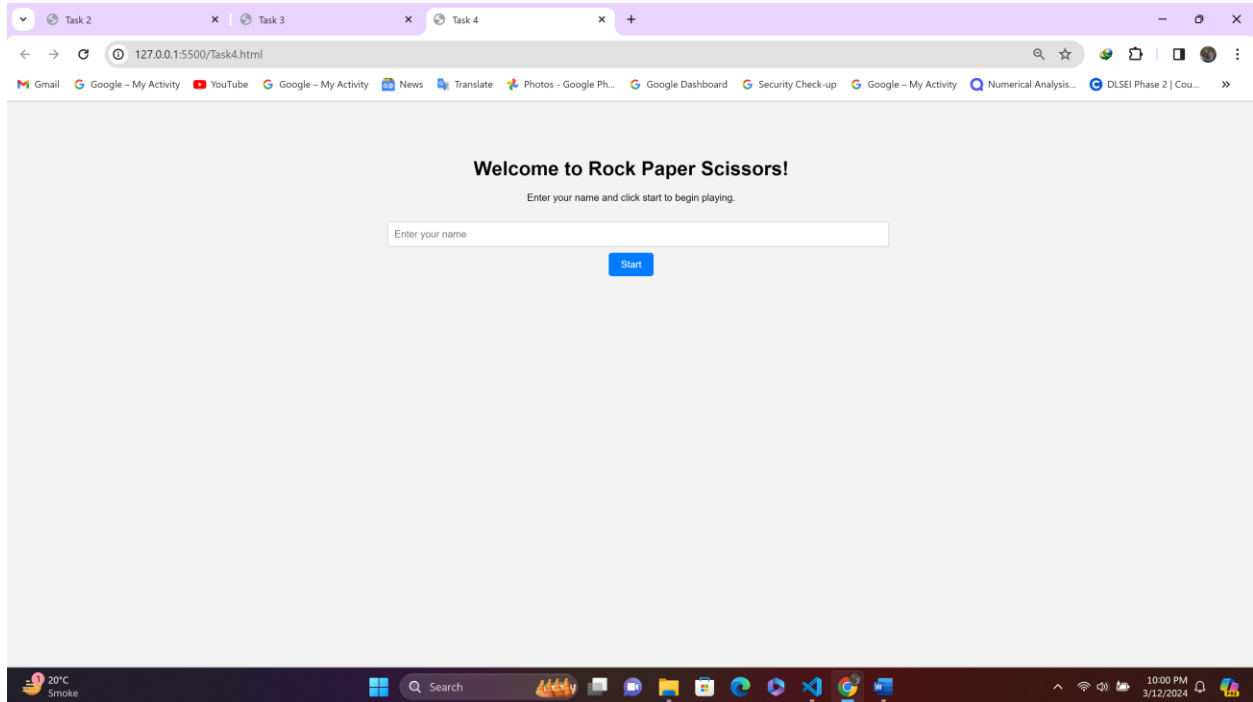
function getResult(user, computer) {
    if (user === computer) {
        return "It's a tie!";
    } else if ((user === "rock" && computer === "scissors") ||
        (user === "paper" && computer === "rock") ||
        (user === "scissors" && computer === "paper")) {
        return "You win!";
    } else {
        return "Computer wins!";
    }
}

function updateScore(result) {
    if (result === "You win!") {
        score++;
    } else if (result === "Computer wins!") {
        score--;
    }
    $("#score-value").text(score);
}

```

```
});
```

Output



THE END