**Phase 4: Development Part 2**

**Introduction:**

This document provides an in-depth overview of an Internet of Things (IoT) project, specifically an "Environmental Monitoring Dashboard." In the age of IoT, the ability to gather and visualize real-time data from various environmental sensors has become crucial for a wide range of applications, including smart agriculture, industrial automation, and climate control systems. This project aims to create a user-friendly web-based dashboard that allows users to monitor and interact with environmental data collected by IoT devices.

**Code:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Environmental Monitoring</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/css/all.min.css">

<link href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@300&family=Montserrat:wght@400&display=swap" rel="stylesheet">

<style>

body {

font-family: 'Open Sans', sans-serif;

margin: 0;

padding: 0;

background: url("https://i.ibb.co/BtWvH3M/hero-image.jpg" alt="hero-image" border="0") center center fixed;

background-size: cover;

color: #333;

}

header {

background-color: rgba(52, 152, 219, 0.8);

color: #ffffff;

text-align: center;

padding: 15px;}

header img {

max-width: 100%;

height: auto;

}

main {

margin: 20px;

text-align: center;

}

.data-display {

background-color: #ecf0f1;

border: 1px solid #bdc3c7;

border-radius: 5px;

padding: 20px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

transition: transform 0.2s;

}

.data-display:hover {

transform: scale(1.05);

}

.data-display h2 {

font-size: 24px;

margin-bottom: 20px;

}

.data-display p {

font-size: 18px;

margin: 10px 0;

}

.data-display span {

font-weight: bold;

}

.data-history {

background-color: #ecf0f1;

border: 1px solid #bdc3c7;

border-radius: 5px;

margin-top: 20px;

padding: 20px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

.data-history h2 {

font-size: 24px;

margin-bottom: 20px;

color: #3498db;

}

table {

width: 100%;

border-collapse: collapse;

margin-top: 10px;

}

table, th, td {

border: 1px solid #bdc3c7;

}

th, td {

padding: 10px;

text-align: center;

}

.gradient-button {

background: linear-gradient(to right, #3498db, #85C1E9);

color: #ffffff;

border: none;

padding: 10px 20px;

border-radius: 5px;

cursor: pointer;

font-size: 18px;

}

.button:hover {

transform: scale(1.1);

transition: transform 0.2s;

}

.data-update {

animation: pulse 2s infinite;

}

@keyframes pulse {

0% {

background-color: #ecf0f1;

}

50% {

background-color: #3498db;

}

100% {

background-color: #ecf0f1; }

}

</style>

</head>

<body>

<header>

<h1>Environmental Monitoring Dashboard</h1>

</header>

<main>

<section class="data-display">

<h2>Current Data</h2>

<p>Temperature: <span id="temperature">-- °C <i class="fas fa-thermometer-half"></i></span></p>

<p>Humidity: <span id="humidity">-- % <i class="fas fa-tint"></i></span></p>

<p>Location: <span id="location">--</span></p>

<p>Last Update: <span id="last-update">--</span></p>

</section>

<section class="data-history">

<h2>Data History</h2>

<table id="data-table">

<thead>

<tr>

<th>Timestamp</th>

<th>Temperature (°C)</th>

<th>Humidity (%)</th>

<th>Location</th>

</tr>

</thead>

<tbody id="data-rows">

<!-- Data rows will be added here dynamically -->

</tbody>

</table>

</section>

</main>

<button class="gradient-button">Refresh Data</button>

<script>

function updateData(data) {

document.getElementById("temperature").textContent = data.temperature + " °C";

document.getElementById("humidity").textContent = data.humidity + " %";

document.getElementById("location").textContent = data.location;

document.getElementById("last-update").textContent = data.timestamp; // Update data history table

const dataTable = document.getElementById("data-table");

const dataRows = document.getElementById("data-rows");

const newRow = dataTable.insertRow(1);

const timestampCell = newRow.insertCell(0);

const temperatureCell = newRow.insertCell(1);

const humidityCell = newRow.insertCell(2);

const locationCell = newRow.insertCell(3);

timestampCell.textContent = data.timestamp;

temperatureCell.textContent = data.temperature;

humidityCell.textContent = data.humidity;

locationCell.textContent = data.location;

}// Poll the server for data every 5 seconds

function fetchData() {

fetch('https://smartenviron.free.beeceptor.com/smartenviron/')

.then(response => response.json())

.then(data => {

updateData(data);

document.querySelector('.data-display').classList.add('data-update'); })

.catch(error => console.error(error));}

document.querySelector('.gradient-button').addEventListener('click', function () {

fetchData(); });

fetchData(); // Initial fetch

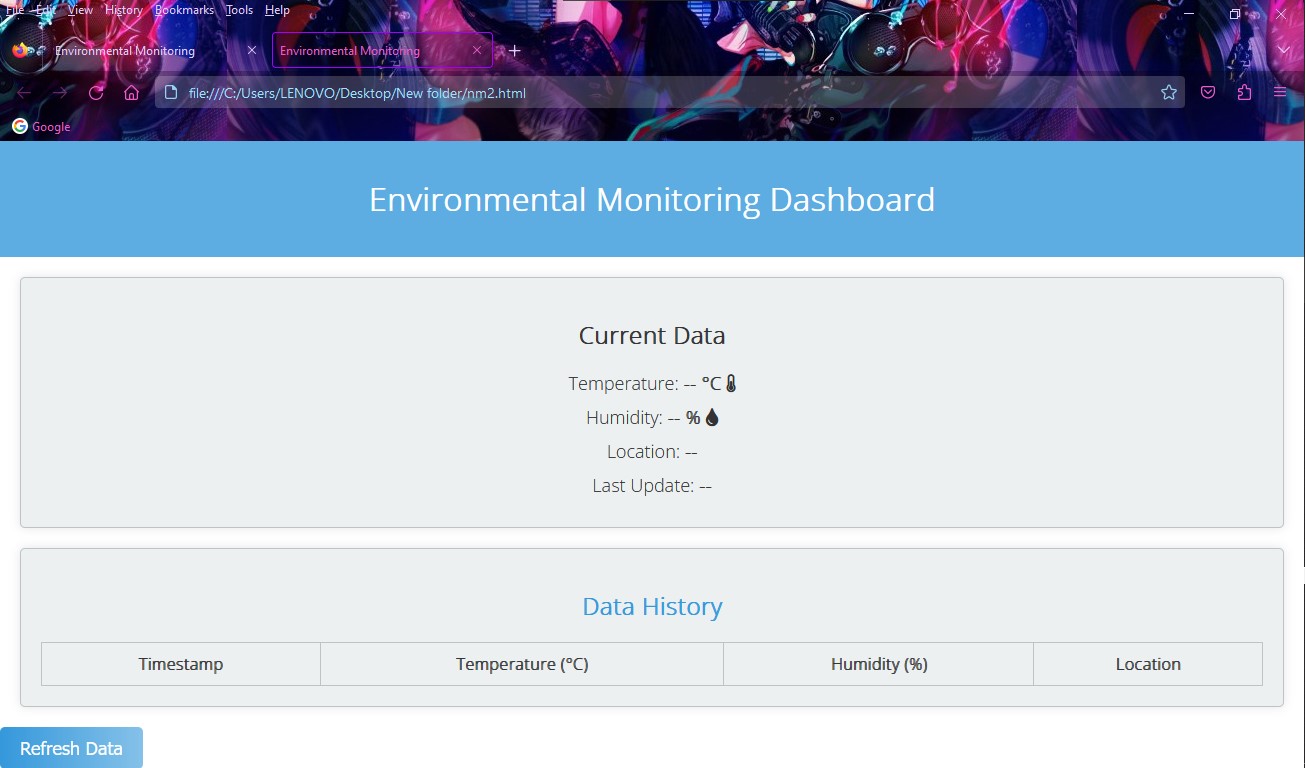
setInterval(fetchData, 5000); // Poll every 5 seconds

</script>

</body>

</html>

**Output:**



**Title:**

The HTML document is for an "Environmental Monitoring" dashboard.

**Metadata and External Resources:**

The <meta> tags specify character encoding and viewport settings. These settings are essential for ensuring the web page displays correctly on various devices, which is important for IoT devices that may have different screen sizes and resolutions.

External resources, like Font Awesome icons and Google Fonts, are included for enhanced visual elements. In an IoT context, these resources can be used to make the dashboard more user-friendly or informative.

**Page Styling:**

The CSS styles define the visual aspects of the page. In IoT, it's crucial to have an appealing and user-friendly interface, especially if the dashboard is used for monitoring IoT device data.

**Header:**

The header section provides a clear title for the IoT dashboard. This title informs users that the dashboard is for environmental monitoring, which is important for context.

**Current Data Display:**

This section is dedicated to displaying real-time data from IoT sensors. In an IoT project, this could represent data like temperature, humidity, or other sensor readings. The use of icons (thermometer and water drop) enhances the user's understanding of the data.

**Data History:**

The data history section is where historical data is displayed in a tabular format. This is important in IoT because it allows users to track trends and patterns over time, which can be critical for decision-making or identifying issues.

**Refresh Data Button:**

The "Refresh Data" button allows users to manually request updated data. In an IoT context, this can be useful for ensuring that users have the most recent information from their IoT devices.

**JavaScript:**

The JavaScript code within the script tag is responsible for fetching data from a server and updating the page with the latest data. In IoT, this script could interact with IoT devices, gather sensor data, and display it in real-time.

**Conclusion:**

In an IoT context, this HTML document serves as a user interface for monitoring and interacting with IoT devices. It provides real-time and historical data visualization, which is crucial for effective IoT system management and decision-making. The refresh button and JavaScript functionality ensure that the dashboard is always up-to-date with the latest data from IoT devices.