**Phase 4 development part 2**

**Project name: smart water management**

**Introduction**:

This development part 2 is to develop the web which monitor the level of water in public tanks in public places. The web is developed using the html, CSS, and java script.

**Html:**

This HTML code defines the structure of a webpage for displaying water level data. It includes a header with a title and a canvas element for rendering a dynamic line chart. The required CSS and JavaScript files are linked.

HTML stands for Hyper Text Markup Language and is used to structure web content.

In this HTML file, we define the basic structure of a web page.

The <head> section includes meta information, such as character encoding and viewport settings.

Inside the <body>, we have the main content of the webpage.

The <header> element is used to create a header section with a title.

The <canvas> element is employed to create a space for rendering the line chart. It has an id attribute set to "water Level Chart," which is used to target it in JavaScript.

**Code for html:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<link rel="stylesheet" href="styles.css">

<title>Random Water Level Data</title>

</head>

<body>

<header>

<h1>Random Water Level Data</h1>

</header>

<main>

<div class="chart-container">

<canvas id="waterLevelChart"></canvas>

</div>

</main>

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

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

</body>

</html>

Css:

**CSS (styles.css):**

CSS stands for Cascading Style Sheets and is used for styling the appearance of web content.

In this CSS file, we apply styling to elements of the HTML page.

The body selector specifies styling for the entire webpage, including font family and text alignment.

The header selector sets styling for the header section, including background color and text color.

The h1 selector defines the appearance of the main title with a specific font size.

The chart-container class styles the chart container, controlling its maximum width and margins.

**JavaScript (script.js):**

JavaScript is a versatile programming language used to add interactivity and functionality to web pages.

In this JavaScript file, we utilize the Chart.js library to create a dynamic line chart.

We define several variables to manage the chart and data. timestamps and water Levels arrays hold the data for the chart.

The create Chart function initializes the Chart.js chart. It defines the type of chart (line chart), data labels, dataset properties, and chart options like time scaling for the X-axis.

The generate Random Data function generates random data points for the last 10 seconds. It calculates timestamps and water level values for each data point, adds them to the arrays, and updates the chart accordingly.

The generate Random Data function is called to initialize the chart with random data.

While this code uses random data for demonstration, in a real application, you would replace the data generation logic with data fetched from sensors or a server.

**Code for css:**

body {

font-family: Arial, sans-serif;

text-align: center;

background-color: #f0f0f0;

}

header {

background-color: #333;

color: white;

padding: 10px;

}

h1 {

font-size: 24px;

}

.chart-container {

max-width: 80%;

margin: 0 auto;

}

**Code for java script:**

document.addEventListener("DOMContentLoaded", function () {

const ctx = document.getElementById("waterLevelChart").getContext("2d");

let chart;

let timestamps = [];

let waterLevels = [];

function createChart() {

chart = new Chart(ctx, {

type: "line",

data: {

labels: timestamps,

datasets: [{

label: "Water Level (cm)",

data: waterLevels,

borderColor: "blue",

fill: false,

}],

},

options: {

scales: {

x: {

type: "time",

time: {

unit: "second",

displayFormats: {

second: "HH:mm:ss",

},

},

},

},

},

});

}

function generateRandomData() {

const currentTime = new Date();

for (let i = 0; i < 10; i++) {

const newTime = new Date(currentTime);

newTime.setSeconds(currentTime.getSeconds() - (10 - i)); // Generate data for the last 10 seconds

const waterLevel = Math.random() \* 100;

timestamps.push(newTime);

waterLevels.push(waterLevel);

}

createChart();

}

generateRandomData();

});

**Output:**



