ส่วนที่ 1 จะเป็นส่วนของโครงสร้างพื้นฐานของ HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Soil Moisture Chart</title>
  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
  <h1>Soil Moisture Levels</h1>
  <canvas id="moistureChart" width="400" height="200"></canvas>
```

คำอธิบาย:

<!DOCTYPE html>: บอกเบราว์เซอร์ว่าไฟล์นี้ใช้ HTML5

<html lang="en">: ระบุว่าเนื้อหาในหน้าเว็บใช้ภาษาอังกฤษ (ช่วยเรื่องการเข้าถึงและ SEO)

<head>:

<meta charset="UTF-8">: ระบุชุดรหัสตัวอักษรเป็น UTF-8 เพื่อรองรับตัวอักษรทุกภาษา

<meta name="viewport" content="width=device-width, initial-scale=1.0">: ทำให้หน้าเว็บแสดงผลได้ดีในอุปกรณ์มือถือ (Responsive Design)

<title>Soil Moisture Chart</title>: ชื่อของหน้าเว็บที่จะแสดงบนแท็บของเบราว์เซอร์

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>: โหลดไลบรารี Chart.js จาก CDN เพื่อสร้างกราฟในหน้าเว็บ

<body>:

<h1>Soil Moisture Levels</h1>: ส่วนหัวของหน้าเว็บที่แสดงชื่อหัวข้อ

<canvas id="moistureChart" width="400" height="200"></canvas>: องค์ประกอบ <canvas> ใช้สำหรับวาดกราฟ โดยกำหนด id="moistureChart" เพื่ออ้างถึงใน JavaScript

```
<script>
  // ฟังก์ชันเพื่อดึงข้อมูลความชื้นจากเซิร์ฟเวอร์
  async function fetchData() {
    try {
      const response = await fetch('http://192.168.130.191:3000/data'); // เปลี่ยน URL ให้ตรงกับพอร์ตใหม่
      const data = await response.json();

      const moistureLevels = data.map(item => item.moisture);
      const timestamps = data.map(item => item.timestamp);

      const ctx = document.getElementById('moistureChart').getContext('2d');
      new Chart(ctx, {
        type: 'line',
        data: {
          labels: timestamps,
          datasets: [{
            label: 'Soil Moisture (%)',
            data: moistureLevels,
            borderColor: 'rgba(75, 192, 192, 1)',
            backgroundColor: 'rgba(75, 192, 192, 0.2)',
            borderWidth: 2,
            fill: true,
          }]
        },
        options: {
          responsive: true,
          plugins: {
            legend: {
              position: 'top'
            },
            tooltip: {
              callbacks: {
                label: function(tooltipItem) {
                  return `Moisture: ${tooltipItem.raw}%`;
                }
              }
            }
          },
          scales: {
            x: {
              title: {
                display: true,
                text: 'Timestamp'
              }
            },
            y: {
              title: {
                display: true,
                text: 'Moisture (%)'
              },
              min: 0,
              max: 100,
              ticks: {
                stepSize: 10
              }
            }
          }
        }
      });
    } catch (error) {
      console.error('Error fetching data:', error);
    }
  }

  // เรียกใช้ฟังก์ชันเมื่อโหลดหน้าเว็บ
  fetchData();
</script>
```

2.1

2.2

3

ส่วนที่ 2 จะเป็นส่วนของ JavaScript (การดึงข้อมูลและสร้างกราฟ)

คำอธิบายแต่ละส่วน

1. async function fetchData()

เป็นฟังก์ชันแบบ Asynchronous (ทำงานแบบไม่บล็อก) ที่ใช้สำหรับดึงข้อมูลจากเซิร์ฟเวอร์

fetch('http://192.168.130.191:3000/data'):

ใช้คำสั่ง fetch เพื่อร้องขอข้อมูลจาก URL ที่ระบุ

URL นี้ชี้ไปยังเซิร์ฟเวอร์ของคุณ (เช่น ESP32) ที่ส่งค่าความชื้นดินออกมาในรูปแบบ JSON

await response.json():

แปลงข้อมูลที่ได้จากเซิร์ฟเวอร์ให้อยู่ในรูปแบบ JSON (ใช้ await เพื่อรอการโหลดข้อมูล)

2.const moistureLevels และ const timestamps

data.map(item => item.moisture):

ใช้ map เพื่อดึงค่า moisture (ความชื้น) ออกมาจากแต่ละออบเจ็กต์ในข้อมูลที่ได้จาก

เซิร์ฟเวอร์

data.map(item => item.timestamp):

ใช้ map เพื่อดึงค่าเวลา (timestamp) ในลักษณะเดียวกัน ผลลัพธ์ของ

timestamps คือ ["2024-12-25 12:00:00", "2024-12-25 12:05:00"]

ส่วนที่ 3 จะเป็นส่วนของการสร้างกราฟด้วย Chart.js

const ctx:

อ้างถึง <canvas> ที่มี id="moistureChart"

ใช้ getContext('2d') เพื่อบอกว่าจะวาดกราฟแบบ 2D

new Chart(ctx, { ... }):

สร้างกราฟใหม่ใน <canvas> โดยใช้ข้อมูลที่เรากำหนด

ส่วนที่ 4 จะเป็นส่วนของข้อมูลในกราฟ

```
data: {
  labels: timestamps,
  datasets: [{
    label: 'Soil Moisture (%)',
    data: moistureLevels,
    borderColor: 'rgba(75, 192, 192, 1)',
    backgroundColor: 'rgba(75, 192, 192, 0.2)',
    borderWidth: 2,
    fill: true,
  }]
},
```

labels: timestamps:

กำหนดแกน X ของกราฟให้ใช้ timestamps (เวลา)

datasets:

ข้อมูลที่แสดงในกราฟ

label: 'Soil Moisture (%)': ชื่อชุดข้อมูล

data: moistureLevels: กำหนดแกน Y ด้วยค่าความชื้น

borderColor และ backgroundColor: กำหนดสีของเส้นและพื้นหลัง

borderWidth: 2: ความหนาของเส้น

```
  options: {
    responsive: true,
    plugins: {
      legend: {
        position: 'top'
      },
      tooltip: {
        callbacks: {
          label: function(tooltipItem) {
            return `Moisture: ${tooltipItem.raw}%`;
          }
        }
      }
    },
    scales: {
      x: {
        title: {
          display: true,
          text: 'Timestamp'
        }
      },
      y: {
        title: {
          display: true,
          text: 'Moisture (%)'
        },
        min: 0,
        max: 100,
        ticks: {
          stepSize: 10
        }
      }
    }
```

ส่วนที่ 5 จะเป็นส่วนตัวเลือกเพิ่มเติม (Options)

responsive: true: ทำให้กราฟปรับขนาดอัตโนมัติตามหน้าจอ

plugins:

legend: { position: 'top' }: ตำแหน่งคำอธิบายชุดข้อมูลอยู่ด้านบน

tooltip:

callbacks: { label: function(tooltipItem) { ... } }: ปรับแต่งข้อความที่แสดงตอนเอาเมาส์ชี้บนกราฟ

scales:

x: ตั้งค่าของแกน X เช่น ชื่อแกน (Timestamp)

y: ตั้งค่าของแกน Y เช่น

ขอบเขต (min = 0, max = 100)

ระยะระหว่างจุด (stepSize = 10)

ส่วนที่ 6 จะเป็นส่วนเรียกใช้ฟังก์ชัน

```
// เรียกใช้ฟังก์ชันเมื่อโหลดหน้าเว็บ
fetchData();
```

เรียกใช้ฟังก์ชัน fetchData() ทันทีที่หน้าเว็บโหลด เพื่อดึงข้อมูลและสร้างกราฟ