```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Ultrasonic & Water Flow Monitoring</title>
  <style>
     body {
        font-family: Arial, sans-serif;
        text-align: center;
        margin: 20px;
     }
     input, button {
        padding: 10px;
        margin: 10px;
        font-size: 16px;
```

```
    }
    #warningMessage {
      color: red;
      font-weight: bold;
    }
    .thresholds {
      margin-top: 30px;
      font-weight: bold;
      text-align: left;
    }
  </style>
</head>
<body>
  <h1>Ultrasonic & Water Flow Monitoring</h1>

  <!-- Display the fetched data -->
  <div id="dataDisplay">
```

```
    <p id="sonic">Ultrasonic: <span id="ultra">-</span> cm</p>
    <p id="waterFlow">Water Flow Rate: <span id="flowValue">-</span> L/min</p>
    <p id="ultrasonic">Water Level: <span id="levelValue">-</span> cm</p>
  </div>

  <!-- Display the warning message -->
  <p id="warningMessage"></p>

  <!-- Display the predefined threshold values -->
  <div class="thresholds">
    <h2>Predefined Thresholds</h2>
    <p><strong>Ultrasonic Range:</strong>Lower: <span id="ultraLowerThreshold">-</span> cm, Upper: <span id="ultraUpperThreshold">-</span> cm</p>
    <p><strong>Water Flow Rate:</strong> Lower: <span id="flowLowerThreshold">-</span> L/min, Upper: <span id="flowUpperThreshold">-</span> L/min</p>
    <p><strong>Water Level:</strong> Lower: <span id="levelLowerThreshold">-</span> cm, Upper: <span id="levelUpperThreshold">-</span> cm</p>
  </div>
```

```
<script>
    // Define initial threshold values (lower and upper)
    const thresholds = {
        ultrasonic:{
            lower: 0,
            upper: 200
        },
        waterFlow: {
            lower: 2,  // lower threshold for water flow rate
            upper: 20  // upper threshold for water flow rate
        },
        waterLevel: {
            lower: 10, // lower threshold for water level
            upper: 200 // upper threshold for water level
        }
```

```
};

async function fetchData() {
    const channelId = "2753392"; // Replace with your ThingSpeak channel ID
    const apiKey = "PQMMJ7UTMLJAJU70"; // Replace with your Read API Key

    try {
        // Fetch data from ThingSpeak API for water flow and ultrasonic sensors
        const responseUltra = await fetch(`https://api.thingspeak.com/channels/${channelId}/fields/1.json?results=1&api_key=${apiKey}`);
        const dataUltra = await responseUltra.json();
        const responseFlow = await fetch(`https://api.thingspeak.com/channels/${channelId}/fields/4.json?results=1&api_key=${apiKey}`);
        const dataFlow = await responseFlow.json();
        const responseLevel = await fetch(`https://api.thingspeak.com/channels/${channelId}/fields/5.json?results=1&api_key=${apiKey}`);
        const dataLevel = await responseLevel.json();

        // console.log(datalevel)
```

```
// Get the latest data values
const ultraData = dataUltra.feeds[dataUltra.feeds.length-1]?.field1 ?? 0;
const flowData = dataFlow.feeds[dataFlow.feeds.length - 1]?.field4;
const levelData = dataLevel.feeds[dataLevel.feeds.length - 1]?.field5;



if (flowData && levelData && ultraData) {
    // Update displayed data
    document.getElementById("ultra").innerText = ultraData;
    document.getElementById("flowValue").innerText = flowData;
    document.getElementById("levelValue").innerText = levelData;

    // Check for threshold violations
    let warningMessage = "";
```

```
if (ultraData < thresholds.ultrasonic.lower) {
    warningMessage += "Underranged!\n";
} else if (ultraData > thresholds.ultrasonic.upper) {
    warningMessage += "OverRanged!\n";
}

// Check for water flow rate threshold violations
if (flowData < thresholds.waterFlow.lower) {
    warningMessage += "Water flow rate is below the lower threshold!\n";
} else if (flowData > thresholds.waterFlow.upper) {
    warningMessage += "Water flow rate is above the upper threshold!\n";
}

// Check for water level threshold violations
if (levelData < thresholds.waterLevel.lower) {
    warningMessage += "Water level is below the lower threshold!\n";
} else if (levelData > thresholds.waterLevel.upper) {
```

```
            warningMessage += "Water level is above the upper threshold!\n";
        }

        // Display warning message if any threshold is violated
        if (warningMessage) {
            document.getElementById("warningMessage").innerText = warningMessage;
        } else {
            document.getElementById("warningMessage").innerText = "All readings are within the thresholds.";
        }
    } else {
        alert("Failed to retrieve data from ThingSpeak.");
    }
} catch (error) {
    console.error("Error fetching data:", error);
    alert("An error occurred while fetching data from ThingSpeak.");
}
```

```
}

// Fetch data every 10 seconds
setInterval(fetchData, 10000);

// Initial fetch when the page loads
fetchData();

// Display the predefined thresholds on the page
window.onload = () => {
    document.getElementById("ultraLowerThreshold").innerText = thresholds.ultrasonic.lower;
    document.getElementById("ultraUpperThreshold").innerText = thresholds.ultrasonic.upper;
    document.getElementById("flowLowerThreshold").innerText = thresholds.waterFlow.lower;
    document.getElementById("flowUpperThreshold").innerText = thresholds.waterFlow.upper;
    document.getElementById("levelLowerThreshold").innerText = thresholds.waterLevel.lower;
    document.getElementById("levelUpperThreshold").innerText = thresholds.waterLevel.upper;
};
```

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