ENVIRONMENTAL MONITORING

Creating an environmental monitoring platform that displays real-time temperature and humidity data from IoT devices involves several steps. Here's a general outline of how you can develop such a platform using web development technologies like HTML, CSS, and JavaScript:

1. **Project Setup:** Set up your development environment. You'll need a code editor and a web server to host your platform. Popular code editors include Visual Studio Code, Sublime Text, or Atom. For the server, you can use Node.js and Express.js for backend if necessary.
2. **Design the User Interface:** Start by designing the user interface of your platform. This typically involves creating the layout and visual elements using HTML and CSS. Consider a responsive design so that your platform works well on various devices (e.g., desktops, tablets, and smartphones). You can use frameworks like Bootstrap or CSS grids for a responsive layout.

```
<!DOCTYPE html>
<html>
<head>
   <link rel="stylesheet" type="text/css" href="styles.css">
</head>
<body>
   <div class="container">
      <div class="data-display">
         <div class="temperature">
            <h2>Temperature</h2>
            <p id="temperature-value">--°C</p>
         </div>
         <div class="humidity">
            <h2>Humidity</h2>
            <p id="humidity-value">--%</p>
         </div>
      </div>
   </div>
   <script src="script.js"></script>
</body>
</html>
```

3.Real-time Data Integration: To display real-time temperature and humidity data, you'll need to communicate with IoT devices. This can be achieved using technologies like MQTT or WebSocket for real-time communication. You will need JavaScript to handle data updates and visualization.

```
// Sample code to subscribe to MQTT data
const mqtt = require('mqtt');
const client = mqtt.connect('mqtt://your-mqtt-broker-url');

client.on('connect', () => {
   client.subscribe('temperature');
   client.subscribe('humidity');
});

client.on('message', (topic, message) => {
   if (topic === 'temperature') {
```

```
        document.getElementById('temperature-value').textContent = message.toString() + '°C';
    } else if (topic === 'humidity') {
        document.getElementById('humidity-value').textContent = message.toString() + '%';
    }
});
```

4.Styling: Use CSS to style your platform. Make it visually appealing and user-friendly. You can define colors, fonts, and layouts to match the aesthetics you want for your monitoring platform.

```
/* styles.css */
.container {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

.data-display {
    display: flex;
    justify-content: space-around;
    width: 80%;
}

.temperature, .humidity {
    text-align: center;
}
```

5. **Testing:** Test your platform thoroughly, especially the real-time data updates. Ensure that it works correctly across different web browsers and devices.
6. **Deployment:** Once you are satisfied with your platform, deploy it to a web server. This can be a cloud-based server or a hosting service.
7. **Security:** Ensure that your platform and communication with IoT devices are secure. Implement authentication and authorization mechanisms to protect your data and control who can access the platform.
8. **Monitoring and Maintenance:** After deployment, continuously monitor your platform's performance and data accuracy. Regularly update the platform to fix bugs and add new features as needed.

Remember to adapt the above steps to the specific requirements of your project and the IoT devices you are working with. Additionally, consider data visualization libraries or frameworks like D3.js for creating interactive charts and graphs to represent your environmental data effectively.

BY
I KRISHNA ANAND
AHMAD RAFFIQ
LALITH KUMAR