Phase 4 : Building the IoT Flood Monitoring and Early Warning System

Project Description:

Creating a platform for real-time water level data and flood warnings that receives data from IoT sensors and issues warnings is a more complex project. It would typically require a server backend to handle sensor data and push notifications to the web page. Below is a simplified example of how to structure the frontend (HTML, CSS, and JavaScript) for such a project.

Applications :

1.Frontend Web Application:

Develop a web interface using HTML, CSS, and JavaScript to display the water level data and flood warnings.

Create a user-friendly dashboard with visualizations and maps to represent the data.

2.Backend Server:

Set up a backend server using a server-side language like Node.js, Python, or Ruby to handle data processing and communication with IoT sensors.

3.IoT Sensor Integration:

Integrate IoT sensors that measure water levels and transmit data to the server in real-time.

Use protocols like MQTT, HTTP, or WebSocket for sensor data communication.

4.Data Storage:

Implement a database (e.g., MySQL, MongoDB) to store historical sensor data for analysis and reference.

5. Real-time Data Processing:

Develop server-side scripts to process incoming sensor data and calculate water level trends.

6.Flood Warning Logic:

Implement algorithms to detect abnormal water level changes or thresholds indicative of potential floods.

Trigger flood warnings based on these thresholds.

7.Notifications:

Integrate a notification system (e.g., email, SMS, push notifications) to alert users and authorities when a flood warning is issued.

8.Map Integration:

Utilize mapping libraries like Leaflet or Google Maps to display the sensor locations and water level data on the frontend.

9.User Authentication and Authorization:

Implement user authentication and authorization to control access to the platform and ensure data security.

10.APIs:

Create RESTful or GraphQL APIs for communication between the frontend and backend components.

Languages : python,html5,css3,javascript.

Hardware : Ultrasonic sensor,Buzzer,dht22,Raspberry pi pico,LED light,Register.

To create a platform that displays real-time water level data and flood warnings.

HTML

Fmes.html

```
<!DOCTYPE html>
```

```
<html>
<head>
    <meta charset="UTF-8">
    <title>Real-Time Water Level Data & Flood Warnings</title>
    <link rel="stylesheet" type="text/css" href="flood.css">
</head>
<body>
    <h1>Flood monitoring system model</h1>
    <div class="hed">
    <div class="header">
        <h1>Real-Time Water Level Data & Flood Warnings</h1>
    </div>
    <div class="content">
        <div class="water-level">
            <h2>Current Water Level: <span id="waterLevel">loading...</span></h2>
        </div>
        <div class="flood-warning">
            <h2>Flood Warning detection: <span
id="floodWarning">loading...</span></h2>
        </div>
    </div>
    <script src="flood.js"></script>
</div>
</body>
</html>
```

CSS:

Fmes.css

```
body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
    margin: 110px;
    padding: 10px;

}

.hed{
    border-style: solid;
}
.header {
    background-color: #d84155;
    color: white;
    text-align: center;
    padding: 20px;
}

.content {
    margin: 20px;
    text-align: center;
}
```

```
.water-level, .flood-warning {
    background-color: rgba(18, 179, 207, 0.486);
    padding: 10px;
    margin: 10px;
    border: 1px solid #ccc;
}
```

JS:

Fmes.js

```
const WaterLevelData = () => (Math.random() * 10).toFixed(2);
const FloodWarningData = () => Math.random() > 0.7;

function updateData() {
    const waterLevelElement = document.getElementById("waterLevel");
    const floodWarningElement = document.getElementById("floodWarning");

    const waterLevel = WaterLevelData();
    const isFloodWarning = FloodWarningData();

    waterLevelElement.textContent = waterLevel + " meters";
    floodWarningElement.textContent = isFloodWarning ? "Yes" : "No";
    floodWarningElement.style.color = isFloodWarning ? "red" : "green";
}
setInterval(updateData, 5000);
updateData();
```