Project Title: IoT Flood Monitoring System

Overview:

Create a flood monitoring system that uses IoT technology to detect and monitor water levels in areas prone to flooding. This system will provide real-time data, alerts, and remote monitoring capabilities to help mitigate flood-related risks.

Components:

Water Level Sensors: Use water level sensors like ultrasonic or pressure sensors to measure water levels at different locations.

Microcontrollers: Employ microcontrollers (e.g., Arduino, Raspberry Pi) to gather data from sensors and send it to a central control unit.

Central Control Unit: This unit collects data from multiple sensors, processes it, and sends it to a cloud server or a local database.

Internet Connectivity: Add a Wi-Fi or cellular module to the central control unit for internet connectivity.

Cloud Server: If using the cloud, set up a cloud server (e.g., AWS, Azure) to store and manage data.

Mobile App/Web Interface: Develop a user-friendly app or web interface to display real-time flood data and send alerts.

Alert System: I

Real-time Data: Provide real-time water level data for multipleUser

Create iot project code for flood monitoring system

#include <Arduino.h>

#include <ESP8266WiFi.h>

#include <ESP8266HTTPClient.h>

// Wi-Fi settings

Const char\* ssid = “YourWiFiSSID”;

Const char\* password = “YourWiFiPassword”;

// Ultrasonic sensor pins

Const int trigPin = D1; // Trigger pin

Const int echoPin = D2; // Echo pin

// Threshold water level (adjust as needed)

Const int floodThreshold = 30; // Example threshold in centimeters

// Cloud server settings (replace with your server details)

Const String serverAddress = <http://yourserver.com/flood-monitor>;

Const String apiKey = “YourAPIKey”;

Void setup() {

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

// Connect to Wi-Fi

WiFi.begin(ssid, password);

While (WiFi.status() != WL\_CONNECTED) {

Delay(1000);

Serial.println(“Connecting to WiFi…”);

}

Serial.println(“Connected to WiFi”);

}

Void loop() {

// Measure the water level

Long duration;

Int distance;

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance = duration / 58.2;

// Print distance to the serial monitor

Serial.print(“Distance: “);

Serial.print(distance);

Serial.println(“ cm”);

// Check if the water level exceeds the threshold

If (distance <= floodThreshold) {

Serial.println(“Flood Alert!”);

sendFloodAlert();

delay(60000); // Wait for 1 minute before sending another alert

}

Delay(10000); // Delay for a period before taking another measurement

}

Void sendFloodAlert() {

// Create an HTTP client object

HTTPClient http;

// Construct the URL with parameters

String url = serverAddress + “?apikey=” + apiKey + “&alert=flood”;

// Send an HTTP GET request to the server

Int httpResponseCode = http.GET(url);

// Check for a successful request

If (httpResponseCode == HTTP\_CODE\_OK) {

Serial.println(“Flood alert sent to server”);

} else {

Serial.println(“Failed to send flood alert to server”);

}

// End the HTTP connection

http.end();

}