FLOOD MONITORING

POWERED BY:

JAYAPRAKESH S

422621106016

COMPONENTS:

- Hardware Components
- Software Components

HARDWARE COMPONENTS:

- Raspberry Pi 3
- Sensors
- Communication Module
- Power Supply

SOFTWARE COMPONENTS:

- Operating System
- Python or Node.js
- IoT Platform
- Database

SENSORS:

- Water Level Sensors
- Rainfall Sensors

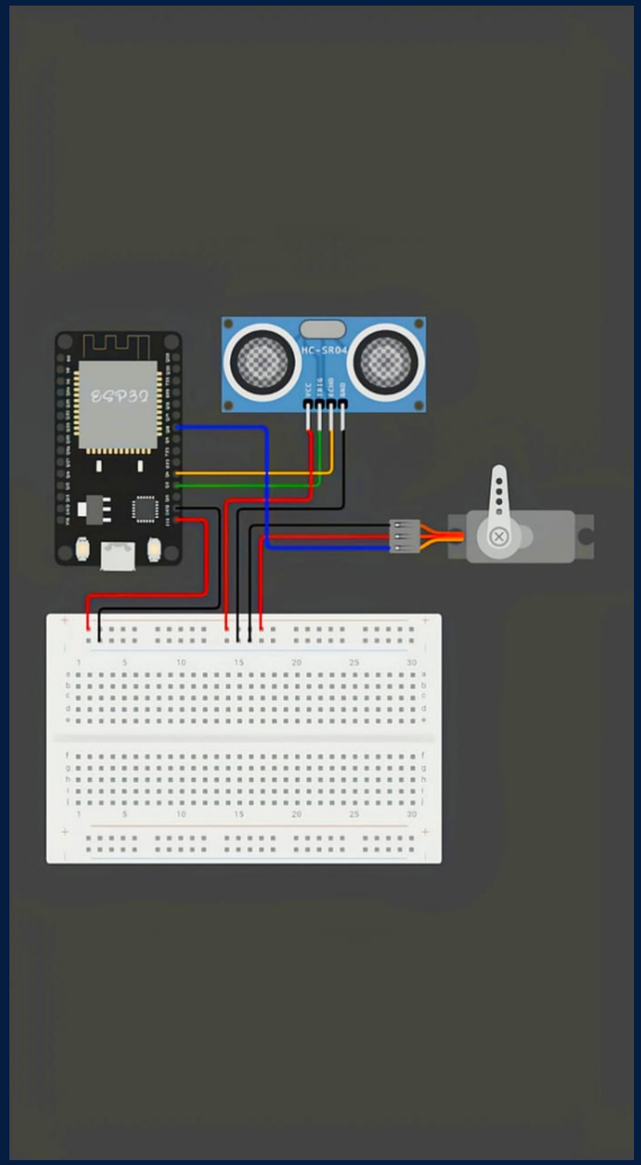
IoT Platform:

- AWS IoT Core
- Google Cloud IoT Core
- Microsoft Azure IoT Hub
- IBM Watson IoT
- Blynk (for a simpler solution)

WORKING PRINCIPLE:

The working principle of a flood monitoring circuit involves the use of sensors to measure water levels or rainfall. These sensors are connected to a data logger, which collects and stores the data. The circuit may also include communication devices to transmit the data to a central monitoring system for analysis and decision-making.

PROTOTYPE:



WORKING:

- Connect your water level and rainfall sensors to the GPIO pins of the Raspberry Pi. Make sure to follow the manufacturer instructions for wiring.

- Develop code to read data from the sensors using GPIO pins and interface libraries for your chosen programming language.Store this data in variables or files.

- Analyze and process the sensor data on the Raspberry Pi. This may include setting threshold values for flood warnings and integrating the rainfall and water level data to assess the situation.
    - Depending on your communication module, send the processed data to your chosen IoT platform. Use MQTT or HTTP for this purpose.

- Set up your IoT platform, create a device,and configure the platform to accept data from your Raspberry Pi. Implement security measures like device authentication.

- Store the incoming data in your chosen database for historical analysis.
- Implement alerting mechanisms in your code to trigger notifications (email, SMS, etc.) when flood conditions are met.

- Create a dashboard to monitor the sensor data in real-time. Most IoT platforms provide visualization tools or you can build a custom dashboard using web development frameworks.

- Ensure that your system can scale to handle a larger number of sensors and data points if needed.

PYTHON SCRIPT:

```
#define BLYNK_TEMPLATE_ID "TMPL3tobBFjj-"
#define BLYNK_TEMPLATE_NAME "IOT FLOOD MONITORING"
#define BLYNK_AUTH_TOKEN "gy2bzR-i-RbPW3oWOpAiDgr6sSVzIHVZ"

char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "Wokwi-GUEST";
char pass[] = "";

#define BLYNK_PRINT Serial
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <ESP32Servo.h>
Servo gate;
const int trigPin=2;//d2
const int echoPin=4;//d4
const int servoPin = 18;//d18
long duration;
int distance;
```

```
void setup() {
  Serial.begin(9600);
  Blynk.begin(auth, ssid, pass);
   pinMode(trigPin, OUTPUT);
   pinMode(echoPin, INPUT);
   gate.attach(servoPin, 500, 2400);
}

void loop()
{
 digitalWrite(trigPin, LOW);
 delay(2);
 digitalWrite(trigPin,HIGH);
 delay(10);
 digitalWrite(trigPin, LOW);


 duration=pulseIn(echoPin,HIGH);
 distance=duration*0.034/2;
 Serial.println(distance);
 Blynk.virtualWrite(V0,distance);


if(distance<50)
{
  gate.write(90);
 Blynk.virtualWrite(V1,"FLOOD DETECTED GATES OPENED");
}
else
{
  gate.write(0);
Blynk.virtualWrite(V1,"SAFE CONDITIONS GATES CLOSED");
}


}
```

THANK YOU