FLOOD MONITORING AND EARLY WARNING

DATE	**31-10-2023**
TEAM ID	**539**
PROJECT NAME	**Flood Monitoring And Early Warning using IOT**

PHASE 4: Development Part -2 – Building Flood Monitoring And Early Warning using IOT project.

PROJECT OVERVIEW:

In regions susceptible to flooding, the need for a reliable Flood Monitoring & Early Warning System is undeniable. The central challenge of this project is to design and implement an IoT-based system that can accurately monitor water levels, weather conditions, and historical data to issue timely flood warnings. This system must contend with the complexity of flood prediction, taking into account numerous variables, and deliver early alerts that empower communities and authorities to mitigate the devastating effects of flooding.

BUILDING A FLOOD MONITORING PROJECT WITHSENSORS:

REAL TIME FLOOD MONOTORING INFORMATION:

1.Water level Monitoring :

- Water level monitoring with an ultrasonic sensor involves measuring liquid levels by sending and receiving sound waves. The sensor emits sound pulses and calculates the distance to the liquid's surface based on the echo. This non-contact method provides accurate and efficient monitoring for various applications, such as tanks, reservoirs, and water systems**.**

2. Location Monitoring :

- In flood monitoring systems, GPRS sensors are used to track the location of monitoring stations and transmit data over cellular networks. These sensors enable real-time data collection, helping authorities respond promptly to flood events, enhance disaster management, and protect vulnerable areas by providing critical location-based information.

3. **Blynk Platform:**
    - Blynk is a popular platform designed for developing IoT (Internet of Things) applications. It provides a user-friendly way for developers to create mobile apps to control various hardware using a wide range of microcontrollers and development boards. Blynk offers a drag-and-drop interface, allowing users to easily build custom applications without extensive coding knowledge. The platform supports a variety of devices and connectivity options, enabling remote monitoring and control of connected hardware via the internet. Blynk is widely used in the maker community, DIY projects, and in various industries where IoT applications are deployed.

BENEFITS:

Implementing IoT for environmental monitoring offers several benefits:

- **Real-time Data:** Instant access to data for timely decision-making.
- **Data Accuracy:** High precision and reliability in data collection.
- **Sustainability:** Facilitating eco-friendly practices and reducing environmental impact.
- **Public Awareness:** Engaging the public in environmental issues throughaccessible data.
- **Research and Policy Support:** Supporting research and government policies with reliable environmental data.

PROJECT PLANNING:

Define the scope, objectives, budget, and timeline for the project.

PROJECT COMPONENTS:

SENSOR SELECTION:

- We have selected a range of sensors, including ultrasonic sensor, GPRS sensor.
- These sensors will provide comprehensive data on environmental conditions within the park.

DEPLOYMENT OF IOT:

- The deployment of the Internet of Things (IoT) in flood monitoring and early warning in reserve areas is a powerful strategy to enhance the management, conservation, and enjoyment of natural resources and public spaces.

MICROCONTROLLER SELECTION:

- The ESP32 as is used in this project. The ESP32 offers robustprocessing power and GPIO capabilities, allowing us to collect, process, and transmit data effectively.

SENSOR INTEGRATION:

- Sensor integration in flood monitoring in water prone areas is essential for collecting accurate and comprehensive data to better manage and preserve natural resources.

- Integrating a variety of sensors can provide a holistic view of the nature ecosystem.

DATA COLLECTION AND PROCESSING:

- Data collection and preprocessing are crucial steps in flood monitoring and early warning.
- Accurate and reliable data collection and careful preprocessing ensure that the data is of high quality, which is essential for informed decision-making and effective management.

DATA TRANSMISSION:

- Data transmission in flood monitoring and early warning is a critical aspect of the monitoring process.

- Data is transmitted to the blynk cloud through Wi-Fi
- It involves the transfer of collected environmental data from sensors and monitoring devices to a central data repository or control center where it can be processed, analyzed, and utilized for various purposes, such as decision-making, research, and public information.

REAL TIME MONITORING:

Real-time monitoring in Flood monitoring and early warning involves the continuous and immediate collection, analysis, and dissemination of data on various natural parameters

The real time processed data is shown as the continuous graph in the blynk cloud platform channel.

DATA VISUALIZATION:

Data visualization in Flood monitoring and early warning is a crucial aspect of turning complex and voluminous data into clear and meaningful insights.

Effective data visualization helps officers, rescue team, and the public better understand the environmental conditions.

TESTING AND CALIBRATION:

- Testing and calibration are essential processes in in Flood monitoring and early warning to ensure that the data collected is accurate, reliable, and consistent.
- Proper testing and calibration procedures help maintain the integrity of monitoring.

CPP SCRIPT DEVELOPMENT

- Develop a CPP script that will run on the IoT for in Flood monitoring and early warning
- This script should be responsible for the following tasks:
- Formatting the collected data for transmission to the IoT platform.
- Managing secure device communication with the IoT platform.
- Implementing security measures, such as data encryption and device authentication.

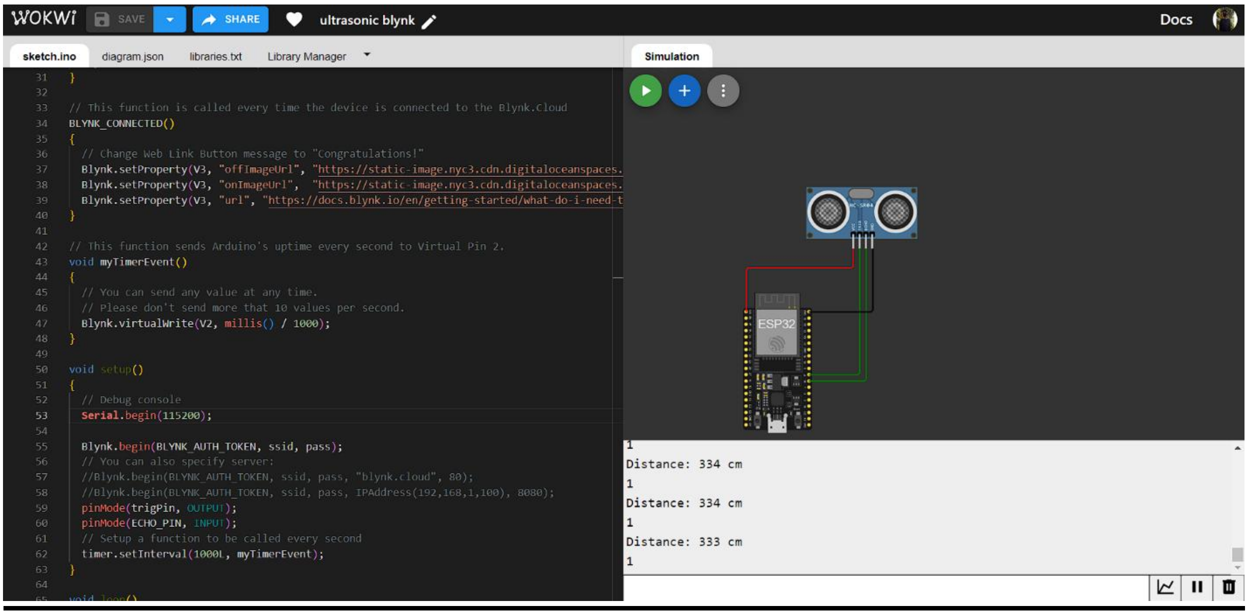
CPP PROGRAM:

```
#define BLYNK_TEMPLATE_ID "TMPL3X0boWy-l"
#define BLYNK_TEMPLATE_NAME "Quickstart Template"
#define BLYNK_AUTH_TOKEN "ya5czx19Uj6izgwze5-Lwkl_FkRUb9JD"

/* Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial


#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#define trigPin 17 // GPIO 4 (D4) for the trigger pin
#define ECHO_PIN 16
long duration;
int distance;
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "Wokwi-GUEST";
char pass[] = "";

BlynkTimer timer;

// This function is called every time the Virtual Pin 0 state changes
BLYNK_WRITE(V0)
{
  // Set incoming value from pin V0 to a variable
  int value = param.asInt();

  // Update state
  Blynk.virtualWrite(V1, value);
}

// This function is called every time the device is connected to the Blynk.Cloud
BLYNK_CONNECTED()
{
  // Change Web Link Button message to "Congratulations!"
  Blynk.setProperty(V3, "offImageUrl", "https://static-
image.nyc3.cdn.digitaloceanspaces.com/general/fte/congratulations.png");
```

```
  Blynk.setProperty(V3, "onImageUrl",  "https://static-
image.nyc3.cdn.digitaloceanspaces.com/general/fte/congratulations_pressed.png");
  Blynk.setProperty(V3, "url", "https://docs.blynk.io/en/getting-started/what-
do-i-need-to-blynk/how-quickstart-device-was-made");
}

// This function sends Arduino's uptime every second to Virtual Pin 2.
void myTimerEvent()
{
  // You can send any value at any time.
  // Please don't send more that 10 values per second.
  Blynk.virtualWrite(V2, millis() / 1000);
}

void setup()
{
  // Debug console
  Serial.begin(115200);

  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
  // You can also specify server:
  //Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass, "blynk.cloud", 80);
  //Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass, IPAddress(192,168,1,100), 8080);
  pinMode(trigPin, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  // Setup a function to be called every second
  timer.setInterval(1000L, myTimerEvent);
}

void loop()
{
  Blynk.run();
  timer.run();
  // You can inject your own code or combine it with other sketches.
  // Check other examples on how to communicate with Blynk. Remember
  // to avoid delay() function!

  digitalWrite(trigPin, HIGH);
  delayMicroseconds(2);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(ECHO_PIN, HIGH);
  distance = 405-(duration / 58); // Calculate distance in centimeters

   // Check if a valid distance is obtained
    Serial.print("Distance: ");
    Serial.print(distance);
    Serial.println(" cm");


  Blynk.virtualWrite(V0, distance);  // Send flood level value to Blynk

  if(distance<20){
```

```
    Blynk.virtualWrite(V1, "LOW");
    Serial.println("1");
  }else if(distance>20 && distance<100){
    Blynk.virtualWrite(V1, "MODERATE");
    Serial.println("1");
  }else {
    Blynk.virtualWrite(V1, "HIGH");
    Serial.println("1");
  }
}
```

SIMULATION OUTPUT:

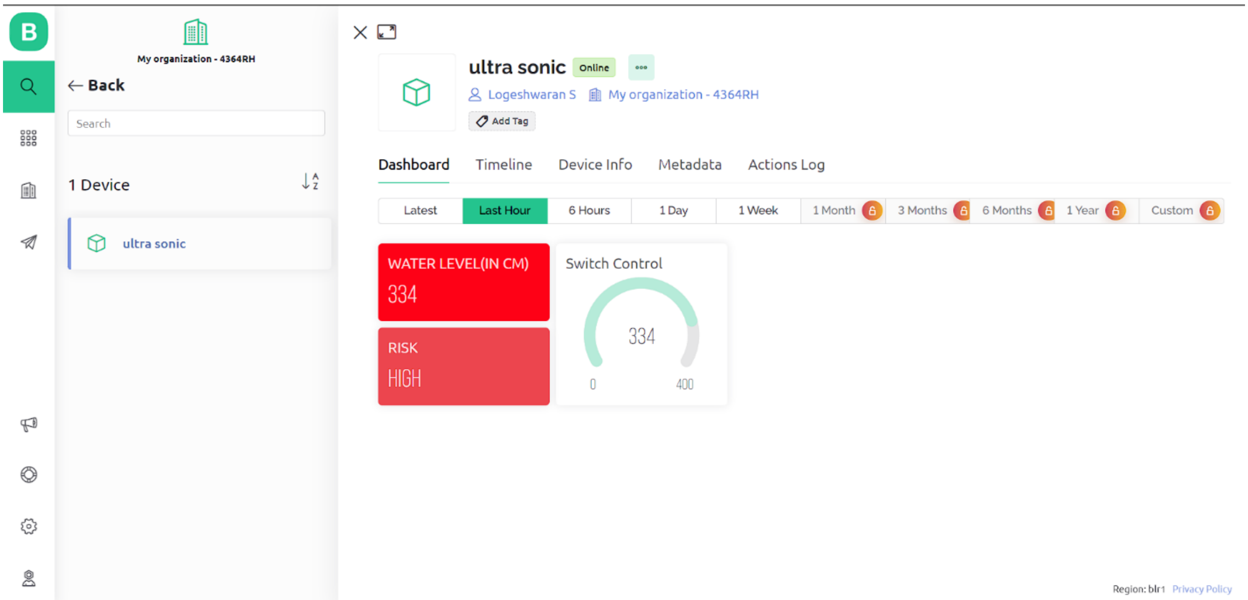


BLYNK PLATFORM OUTPUT:

There different flood alert levels are assigned in the program that is high risk of the flood the water level reached to the peak of water prone zone areas

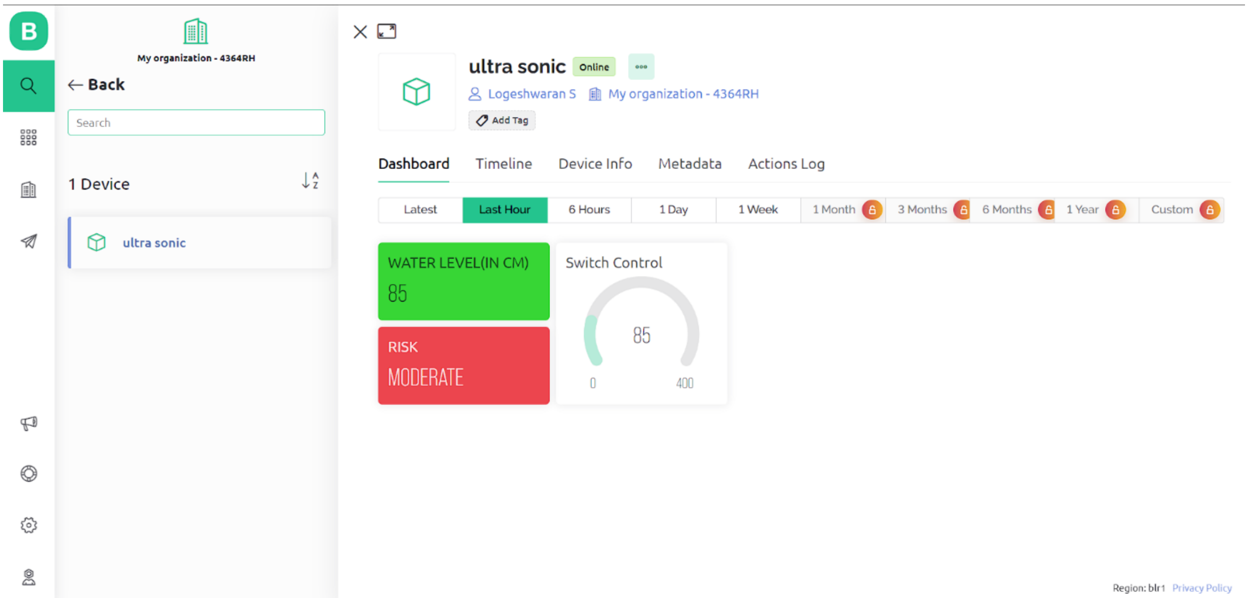
HIGH RISK OF FLOOD:

If the water level is above 100 below 400cm then there is the low risk of flood in those areas.



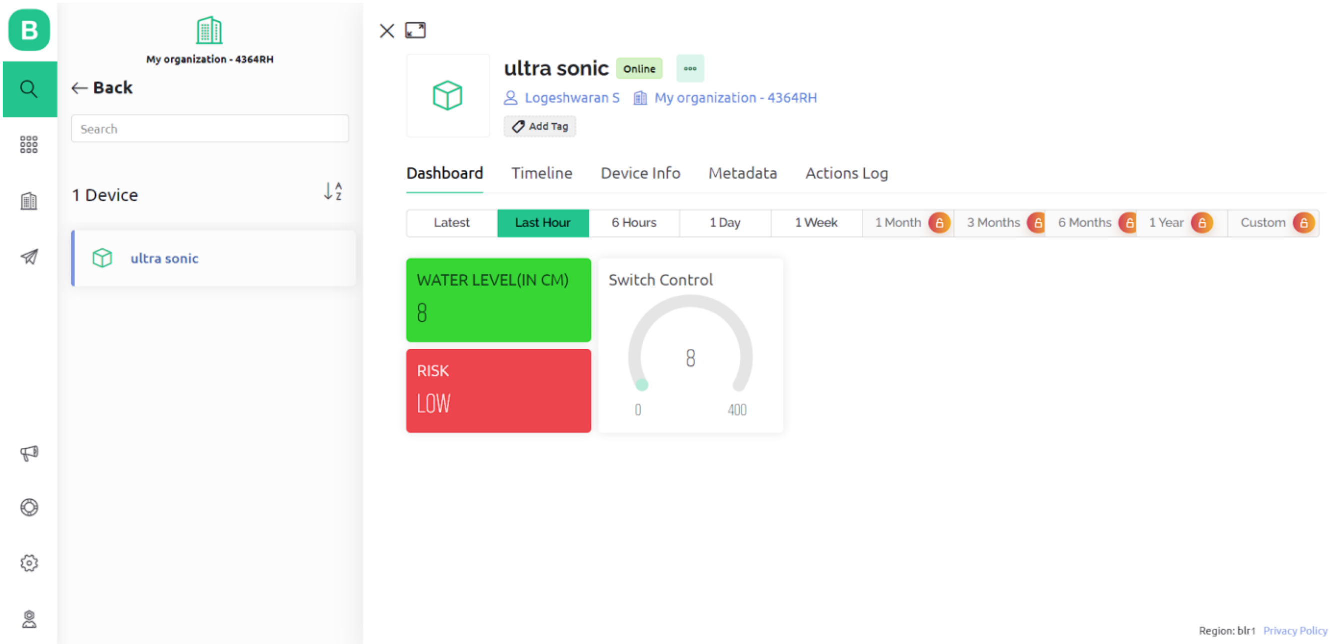
MODRATE RISK OF FLOOD:

If the water level is above 20 below 100cm then there is the low risk of flood in those areas.



LOW RISK OF FLOOD:

If the water level is below 20cm then there is the low risk of flood in those areas.



4. Conclusion

In this document, we have simulated and displayed the output of our project using wokwi and blynk to addressing the challenge of flood monitoring and early awareness using IoT technology. By empathizing with users, defining clear objectives, ideating innovative solutions, prototyping, testing, implementing, and iterating, we aim to develop a robust and user-friendly system that enhances early flood warnings and supports emergency response efforts. Our ultimate goal is to contribute to the safety and resilience of communities in flood-prone areas by providing timely and accurate flood data.