**FLOOD MONITORING AND EARLY WARNING**

**PHASE 3- DEVELOPMENT PART 1**

**TEAM MEMBERS:**

* **R.ARUNA**
* **M.HARI BAALAJI**
* **B.SABARI**
* **A.UDHAYA KUMAR**

**PROBLEM STATEMENT:**

Start building the IOT flood monitoring and early warning.

**INTRODUCTION:**

Floods are one of the most devastating natural disaster, causing immense damage to both human lives and property. The ability to predict and monitor water levels in flood – prone areas is critical for disaster management and mitigation . In recent years, the deployment of internet of things technology has emerged as a powerful tool in this regard.

IOT refers to the interconnections of everyday objects and devices to the internet, enabling them to collect and exchange data. When applied to flood-prone areas, IOT technology can play a pivotal role in monitoring and measures water level in real-time, allowing for more efficient early warning system and flood management.

**DEPLOY IOT SENSORS IN FLOOD-PRONE AREAS AND CONFIGURE THEM TO MEASURES WATER LEVELS:**

* **Identify Flood-Prone Areas:** Start by identifying the specific locations where you want to deploy the water level sensors. These should be areas that are known to be at risk of flooding.
* **Select Suitable Sensors:** Choose water level sensors that are designed for outdoor use and can withstand exposure to water and extreme weather conditions. Make sure they are IOT-enabled, allowing them to transmit data over the internet.
* **Power Supply:** Ensure that you have a reliable power source for the sensors. This could be batteries, solar panels, or a connection to the local power grid, depending on the availability of resources.

* **Connectivity:** Set up a communication network, such as a cellular network or a dedicated IOT network, to enable the sensors to transmit data to a central server or cloud platform. Ensure that the connectivity is reliable in these remote areas.
* **Sensor Placement:** Install the water level sensors at strategic points in the flood-prone areas. They should be placed near water bodies, rivers, or areas where flooding is likely to occur. Proper mounting and calibration are essential.

* **Data Transmission:** Configure the sensors to measure water levels at regular intervals. They should be programmed to send this data to a central database or cloud server. Ensure data encryption and security measures are in place to protect the information.
* **Data Analysis and Visualization:** Implement a system for data analysis and visualization. This can be a dashboard or software that processes the data and provides real-time insights into water levels. It should trigger alerts when levels rise to a dangerous point.
* **Alerting and Response**: Set up a system for alerting relevant authorities or local residents when water levels reach a critical threshold. This could involve sending SMS alerts, emails, or activating sirens and warning systems.
* **Maintenance and Monitoring:** Regularly maintain and monitor the sensors to ensure they are functioning correctly. Replace batteries or perform necessary maintenance to keep the system operational.
* **Community Awareness:** Inform the local community about the deployment of these sensors and how they can access flood information. Public awareness and education are important for community safety.
* **Data Storage and Historical Analysis:** Store historical data for future analysis and understanding of flood patterns. This can help in better preparation and response planning.

By following these steps, you can deploy IOT sensors in flood-prone areas to measure water levels and improve early warning systems, ultimately helping to mitigate the impact of floods on communities and the environment.

**PYTHON SCRIPT (IOT SENSORS TO SEND COLLECTED WATER LEVEL DATA TO THE EARLY WARNING PLATFORM)**

import RPi. GPIO as GPIO

import time

from BlynkLib import Blynk

# Blynk authentication token

BLYNK\_AUTH\_TOKEN = "Your\_Blynk\_Auth\_Token"

# Initialize Blynk

blynk = Blynk(BLYNK\_AUTH\_TOKEN)

# Ultrasonic sensor GPIO pins

TRIG\_PIN = 2 # BCM GPIO 2 (Physical Pin 3)

ECHO\_PIN = 4 # BCM GPIO 4 (Physical Pin 7)

# Servo motor GPIO pin

SERVO\_PIN = 18 # BCM GPIO 18 (Physical Pin 12)

# Set the GPIO mode to BCM

GPIO.setmode(GPIO.BCM)

# Set up GPIO pins

GPIO.setup(TRIG\_PIN, GPIO.OUT)

GPIO.setup(ECHO\_PIN, GPIO.IN)

GPIO.setup(SERVO\_PIN, GPIO.OUT)

# Initialize the servo

pwm = GPIO.PWM(SERVO\_PIN, 50)

pwm.start(0)

# Function to measure distance using the ultrasonic sensor

def measure\_distance():

GPIO.output(TRIG\_PIN, True)

time.sleep(0.00001)

GPIO.output(TRIG\_PIN, False)

start\_time = time.time()

stop\_time = time.time()

while GPIO.input(ECHO\_PIN) == 0:

start\_time = time.time()

while GPIO.input(ECHO\_PIN) == 1:

stop\_time = time.time()

elapsed\_time = stop\_time - start\_time

distance = (elapsed\_time \* 34300) / 2 # Speed of sound = 343 m/s

return distance

# Function to control the servo motor

def control\_servo(angle):

duty = angle / 18 + 2

GPIO.output(SERVO\_PIN, True)

pwm.ChangeDutyCycle(duty)

time.sleep(1)

GPIO.output(SERVO\_PIN, False)

# Blynk virtual write handler

@blynk.handle\_event('write V0')

def write\_virtual\_pin\_handler(pin, value):

# Measure distance and send it to Blynk

distance = measure\_distance()

blynk.virtual\_write(0, distance)

@blynk.handle\_event('write V1')

def write\_virtual\_pin\_handler(pin, value):

if int(value[0]) == 1:

# Open the gates

control\_servo(90)

blynk.virtual\_write(1, "FLOOD DETECTED, GATES OPENED")

else:

# Close the gates

control\_servo(0)

blynk.virtual\_write(1, "SAFE CONDITIONS, GATES CLOSED")

try:

while True:

blynk.run()

except KeyboardInterrupt:

  GPIO.cleanup()

**CONCLUSION:**

Deploying IOT sensors to measure water levels in flood-prone areas offers significant benefits for flood management, safety, and data-driven decision-making. However, ongoing maintenance and integration with existing systems are key to the success of such deployments.