**FLOOD MONITORING AND EARLY WARNING**

**PHASE 3: Development part 1**

**Datasets:**

**1. Weather Data:**

Obtain historical and real-time weather data, including rainfall, temperature, and humidity. Organizations like NOAA and ECMWF provide access to such data**.**

**2. Hydrological Data:**

Collect river flow, water level, and groundwater data from relevant agencies or organizations.

**3. Topographical Data:**

Use elevation and terrain data to create flood risk models. Data sources like USGS can be valuable**.**

**4. Satellite Imagery:**

Utilize satellite imagery for monitoring large areas. Sources like NASA and ESA provide such data**.**

**5. Sensor Networks:**

Deploy flood sensors and IoT devices to collect data from flood-prone areas.

**6. Historical Flood Data:**

Historical records of floods and their impacts can help in building predictive models.

**Algorithms:**

**1. Machine Learning Models:**

Utilize supervised learning algorithms like Random Forest, Gradient Boosting, or deep learning models for flood prediction. These models can analyse historical data to make predictions**.**

**2. Hydrological Models:**

Implement hydrological models such as HEC-HMS or SWAT to simulate river flow and predict flood events**.**

**3. Geospatial Analysis:**

Use GIS (Geographic Information Systems) for spatial analysis and modeling . Tools like ArcGIS or open-source alternatives can be helpful.

**4. Time Series Analysis:**

Apply time series analysis techniques to monitor changes in weather and river data over time. Techniques like ARIMA or LSTM can be used.

**5. Data Fusion:**

Combine data from various sources to create a comprehensive flood monitoring system. Kalman filtering and data assimilation techniques can help integrate different data types**.**

**6. Early Warning Systems:**

Develop alerting algorithms that trigger warnings based on thresholds and predictive models. This may involve defining critical levels for rivers or rainfall.

**7. Remote Sensing:**

Use remote sensing technologies such as RADAR and LiDAR for flood detection and monitoring, especially in real-time scenarios**.**

**8. Social Media and Crowdsourced Data:**

Consider incorporating social media and crowdsourced data for real-time reports on flood situations.

**9. Communication Systems:**

Ensure that the system is integrated with communication channels to disseminate warnings effectively.

**10. Machine Vision for Image Analysis:**

If using satellite or drone imagery, machine vision algorithms can be used for image analysis to detect water bodies and changes in water levels.

The effectiveness of your flood monitoring and early warning system will depend on the quality and availability of data, as well as the accuracy of the algorithms used. Collaboration with relevant government agencies, meteorological departments, and environmental organizations can be beneficial in acquiring data and building an effective system.

**HARDWARE COMPONENTS:**

1.Ultrasonic sensor:

An ultrasonic sensor is a device that uses high-frequency sound waves (ultrasonic waves) to measure distances and detect objects. It typically consists of a transmitter that emits ultrasonic pulses and a receiver that listens for the echoes of these pulses. By measuring the time it takes for the sound waves to bounce off an object and return to the sensor, it can calculate the distance to the object. Ultrasonic sensors are commonly used in robotics, industrial automation, and various applications where non-contact distance measurement or object detection is required.

2.Raspberry pi:

The Raspberry Pi is a series of small, affordable single-board computers developed by the Raspberry Pi Foundation. These compact computers are designed to be versatile, low-cost, and energy-efficient, making them popular for a wide range of projects and applications.

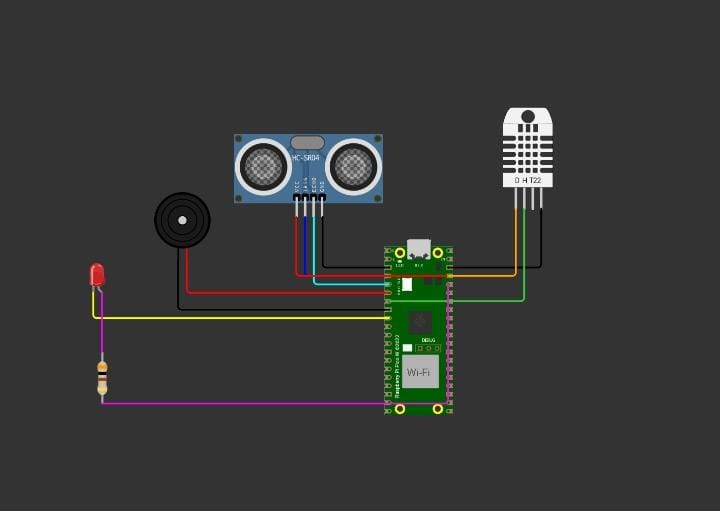
3.Buzzers:

A buzzer is an electromechanical device that produces a buzzing or beeping sound when an electrical current is applied to it. Buzzer is a simple and widely used acoustic signaling device in various applications.

4.DHT22:

The DHT22, also known as the AM2302, is a digital temperature and humidity sensor. It is commonly used in electronic projects and IoT (Internet of Things) applications to measure both temperature and humidity. The sensor uses a capacitive humidity sensor and a thermistor to provide accurate and reliable readings.

**SIMULATION**:



**CODE**:

import RPi.GPIO as GPIO

import Adafruit\_DHT

import time

# GPIO pin numbers

ULTRASONIC\_TRIGGER = 23

ULTRASONIC\_ECHO = 24

BUZZER\_PIN = 17

LED\_PIN = 18

DHT\_SENSOR = Adafruit\_DHT.DHT22

DHT\_PIN = 22

# Constants

FLOOD\_ALERT\_DISTANCE = 30 # Adjust this distance as needed (in centimeters)

TEMP\_THRESHOLD = 25.0 # Adjust this temperature threshold as needed (in Celsius)

HUMIDITY\_THRESHOLD = 60 # Adjust this humidity threshold as needed (%)

# Set up GPIO

GPIO.setmode(GPIO.BCM)

GPIO.setup(ULTRASONIC\_TRIGGER, GPIO.OUT)

GPIO.setup(ULTRASONIC\_ECHO, GPIO.IN)

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

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

def measure\_distance():

# Send a trigger pulse to the ultrasonic sensor

GPIO.output(ULTRASONIC\_TRIGGER, True)

time.sleep(0.00001)

GPIO.output(ULTRASONIC\_TRIGGER, False)

start\_time = time.time()

stop\_time = time.time()

# Wait for the echo signal

while GPIO.input(ULTRASONIC\_ECHO) == 0:

start\_time = time.time()

while GPIO.input(ULTRASONIC\_ECHO) == 1:

stop\_time = time.time()

# Calculate distance

elapsed\_time = stop\_time - start\_time

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

return distance

def read\_dht22():

humidity, temperature = Adafruit\_DHT.read\_retry(DHT\_SENSOR, DHT\_PIN)

return humidity, temperature

def alert\_flood():

GPIO.output(BUZZER\_PIN, GPIO.HIGH)

GPIO.output(LED\_PIN, GPIO.HIGH)

print("Flood Alert!")

# You can implement additional actions here, e.g., sending notifications

def reset\_alert():

GPIO.output(BUZZER\_PIN, GPIO.LOW)

GPIO.output(LED\_PIN, GPIO.LOW)

try:

while True:

distance = measure\_distance()

humidity, temperature = read\_dht22()

print(f"Distance: {distance} cm")

print(f"Temperature: {temperature}°C, Humidity: {humidity}%")

if distance < FLOOD\_ALERT\_DISTANCE:

alert\_flood()

elif temperature > TEMP\_THRESHOLD or humidity > HUMIDITY\_THRESHOLD:

alert\_flood()

else:

reset\_alert()

time.sleep(1)

except KeyboardInterrupt:

GPIO.cleanup()