

FLOOD WATER MONITERING AND EARLY WARNING

IOT_PHASE 5

BY
ROSHAN.B

INTRODUCTION

OVERVIEW :

Human based resistive mechanisms towards flood control open up multitude problems like dynamic reactions of prior alien about the risky situations and stage of current water level. The growth of Internet of Things (IOT) paved the significant attention in all fields. Today, droughts and floods are a common feature and their co-existence poses a potent threat, which cannot be eradicated but has to be managed. Transfer of the surplus monsoon water to areas of water deficit is a potential possibility. This would also help create additional irrigation potential, the generation of hydropower, as well as overcoming regional imbalances. From the recent floods in Kolhapur, Satara, etc we observed the severe conditions people had to face due to improper management of the Almatti dam situated on the Maharashtra • Karnataka border. Ample people and animals lost their lives, all the living standard was disturbed. Generally, local people lose their important contact with the river once a dam comes up on it. They are scared to go near it fearing for their lives. Dams can easily become a weapon of terror unlike the temple of development they are projected to be as they have the capacity to suddenly release a lot of water downstream. Keeping this scenario and recent incident in mind, there is a need to develop some proper dam management system. An IoT based dam management system is being developed in this project to avoid floods occurring due to improper management of opening and closing of doors of dam. An attempt to make a system to sense the temperature, humidity, water level and rain prediction is made, depending on which the opening door percentage will be decided.

Node MCU is an IoT based platform which plays an important role in the project. Ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011.

Servo motor is used to control the opening and closing of doors of dam. Node MCU is an open

MX ice rot platform, whose firmware is developed for ESP8266 wi-fi chip. This is used to interface the ultrasonic sensors. The data from these sensors is sent to the cloud for decision making stage that is designed using machine learning algorithm using python. The output of machine learning algorithm will be sent back to node MCU which then will decide how much extent the gates of the dam should be opened.

n. LITERATURE REVIEW

As India faced recent devastating flood in Kerala, there arises a need of efficient flood monitoring system. Flood forecasting and the issuing of flood warnings are effective ways to reduce damage. The proposed system will be efficient because it has better coordination of monitoring, communication and transmission

technology which are adaptable to background condition. The proposed system in [1] also ensures increased accessibility for assessment of emergency situations and enhances effectiveness and efficiency in responding to catastrophic incidents. In summary, the proposed system would be beneficial to the community for decision making and evacuation planning purposes. Flooding is a natural phenomenon which has extracted global attention. Both the positive and negative impact on the world. Developing nations such as Nigeria have been predicted to experience increased flood occurrence in the coming decade. The events of flooding are unlikely to change, however, its impact on our society can be very well reduced. Paper [2] focuses on providing early warning for areas likely to be ravaged by flood even using Wireless Sensor Network (WSN). The system involves the deployment of sensor nodes at specific flood vulnerable locations for real time flood monitoring and detection. Flood events relating to flash flooding and run-off water or overflow are successfully monitored in real time which gives individuals plenty of time to prepare against predicted flood occurrence, saving them from the aftermath of flood disaster. The system was tested via simulation of different flood scenarios

ABSTRACTION :

Water plays an important role in our day to day life in various fields.

Introduction of new methods to solve the water related problems includes adaptive management, remote sensing with the new concepts such as water security, global integration of information, etc. Recently, we can see an increasing amount of dam damage or failure due to aging, earthquakes occurrence and unusual changes in weather. For this reason, dam safety is gaining more importance than ever before in terms of disaster management at a national level. Therefore, the government is trying to come up with an array of legal actions to secure consistent dam safety. Other dam management organizations are also taking various institutional and technical measures for the same purpose. What should be the standard operating procedure in case sudden release is required. This project proposes an IOT based. When it comes to dam safety issues, there is hardly a set of rules for how and when water can be released and dam management system. In this, four ultrasonic sensors namely, temperature sensor, humidity sensor, water level sensor and rain sensor are used to monitor the state of dam. Node MCU is used to interface all the sensors and send the data to the cloud. Machine learning algorithm is designed to control the opening of doors of dam via a servo motor.

Keyword: Flood detection. IoT, Python. Embedded C. Node MCU. DNN

SYSTEM ANALYSIS

EXISTING SYSTEM:

METHODOLOGY:

After going through literature survey and various research papers we finalized our hardware and software requirements. Various natural factors, which includes humidity, temperature, water level and flow level are observed by system to detect flood. Our system consists of different sensors which helps to collect data for individual parameters.

1. For detecting changes in humidity and temperature the system has a DHT11 Digital Temperature Humidity Sensor. It is a sensor which detects humidity and temperature
2. The water level is always under observation by an Ultrasonic sensor, which works by constantly monitoring as water levels rise and fall. Once the water level increases beyond threshold, a trigger is generated which sends an Email Alert indicating the rise of water and possibility of flood.
3. The Flow sensor on the system keeps eye on the flow of water. The speed changes when water falls on rotor which makes it to rotate. After the successful completion of hardware setup, we move towards software setup and using Arduino IDE and visual studio code

We created a Project Email for sending Email Alerts. Using Python Scripting and interfacing the Arduino outputs with Python, the program reads the inputs from the sensors via Arduino. We set a threshold value for the water level, once the water crosses the threshold level, a trigger causes the program to send an Email Alert notifying people regarding rise in water level and possibility of flood.

PROPOSED SYSTEM :

Flood is a huge threat to humanity as it is also considered one of the most devastating natural disasters in the world. Since flooding results in great damage to agricultural land, residential area, and the economy of the country. In a country like India, with extreme weather and climatic conditions, the occurrence of heavy rainfall. We are not just monitoring the water level using Ultrasonic Sensor, but also monitoring the flow of water using Flow Sensor which gives an upper hand for immediate alerting the danger. We are using advance Temperature and Humidity Sensor for getting more accurate and correct values of temperature and humidity of surrounding, this sensor works in all extreme conditions. We are using a Wi-Fi module which can send data over the internet. We have used Python for sending Email Alerts interfacing it water.

ADVANTAGE :

- Timely detection of possible flood risks and floods.
- Highly reliable and available real-time data.
 - Tailored solution that can be integrated with external developments at any level (device, connectivity, cloud or user application).
- Total adaptation and integration with emergency plans.
- Creation of historic data for Administrations.
- Low energy consumption.
- An unlimited number of devices can be included in future extensions.
- Far-reaching bidirectional communications.
- Long working life of the equipment.

DISADVANTAGES :

- Inability to produce highly accurate results,
- If there is no sufficient data flood prediction cannot be done.
- It saves the data into the database but doesn't use it
- hence wasting the space. -It cannot predict the flood with its historic data.
- GPS module track is not upgraded in the system
- Hyper pipes algorithm considered as having the lowest accuracy percentage.

SYSTEM SPECIFICATION :



HARDWARE SPECIFICATION :

- Raspberry Pi 3
- Wifi Module
- LCD Display

- ☐ Water Sensor
- ☐ Rain Drop Sensor
- ☐ Resistors
- ☐ Capacitors
- ☐ Transistors
- ☐ Cables and
- ☐ Connectors Diodes
- ☐ PCB and
- ☐ Breadboards LED
- ☐ Transformer/Adapter
- ☐ Push Buttons
- ☐ Switch
- ☐ IC
- ☐ IC Sockets

☐ **SOFTWARE SPECIFICATION :**

- ☐ Linux
- ☐ Programming Language: Python

☐ **SOFTWARE DESCRIPTION :**

A software specification for flood monitoring and early warning typically involves the following components:

☐ **User Requirements:**

Define the user requirements, such as government agencies, emergency responders, and the general public. Specify the geographical scope and areas to be covered by the system.

☐ **System Overview:**

Provide an overview of the software system, including its purpose and main functions. Define the scope of the system, including the types of floods it will monitor (e.g., riverine, flash floods).

☐ **Data Sources:**

Identify the sources of data to be used, such as weather data, river level sensors, and satellite imagery. Specify data acquisition methods and data formats.

☐ Data Processing:

Describe how raw data will be processed and transformed into meaningful information.
Explain algorithms for flood prediction and early warning.

☐ User Interface:

Define the user interface design, including maps, dashboards, and alerts.
Specify user roles and permissions for accessing the system.

☐ Alerting and Notification:

Detail how warnings and alerts will be generated and delivered to users.
Specify communication channels (e.g., SMS, email, mobile app notifications).

☐ Geospatial Mapping:

Describe the mapping components, including GIS integration, mapping layers, and geospatial data visualization.

☐ Historical Data:

Include a database for storing historical flood data for analysis and future planning.

☐ System Performance:

Define performance metrics, such as response time and accuracy of flood predictions. Specify hardware and software requirements.

☐ Testing and Validation:

Describe the testing procedures, including unit testing, integration testing, and user acceptance testing.
Specify validation criteria for the accuracy of flood predictions.

☐ Maintenance and Support:

Detail the procedures for ongoing maintenance and updates.
Specify support channels for users to report issues and request assistance.
Security. Outline security measures to protect data and the system from unauthorized access or breaches.

☐ Compliance and Regulations:

Ensure that the system complies with relevant environmental and safety regulations.

Documentation:



Provide documentation for users, administrators, and developers, including user manuals and system architecture documentation.

☐ Deployment and Scalability:

Describe how the system will be deployed, scaled, and managed over time.

☐ Budget and Resources:

Estimate the budget required for software development, hardware, and ongoing operations.

☐ Timeline:

Create a timeline for the development, testing, and deployment phases of the system.

☐ Risk Assessment:

Identify potential risks and mitigation strategies, such as data inaccuracies, system failures, and user adoption challenges.

☐ Integration:

Specify how the software specification for flood monitoring and early warning typically involves the following components:

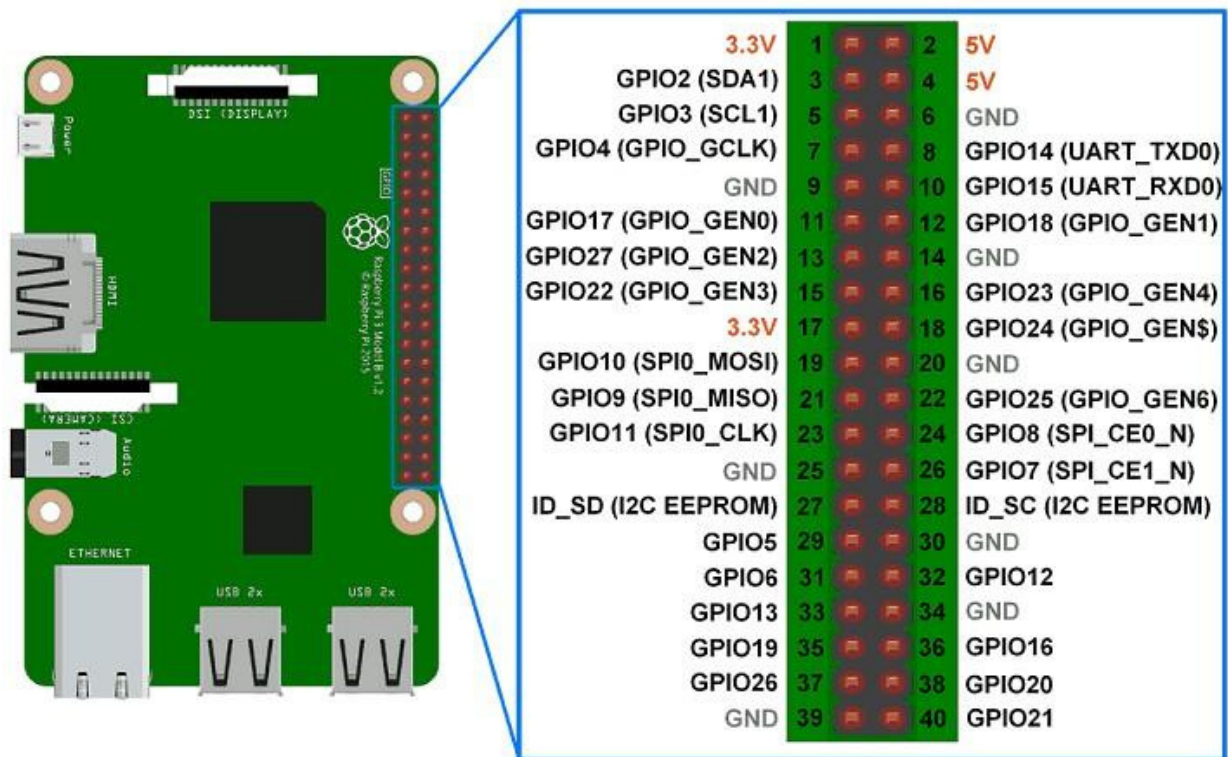
☐ User Requirements:

Define the user requirements, such as government agencies, emergency responders, and the general public.

Specify the geographical scope and areas to be covered by the system.

SYSTEM DESIGN

DHT11:



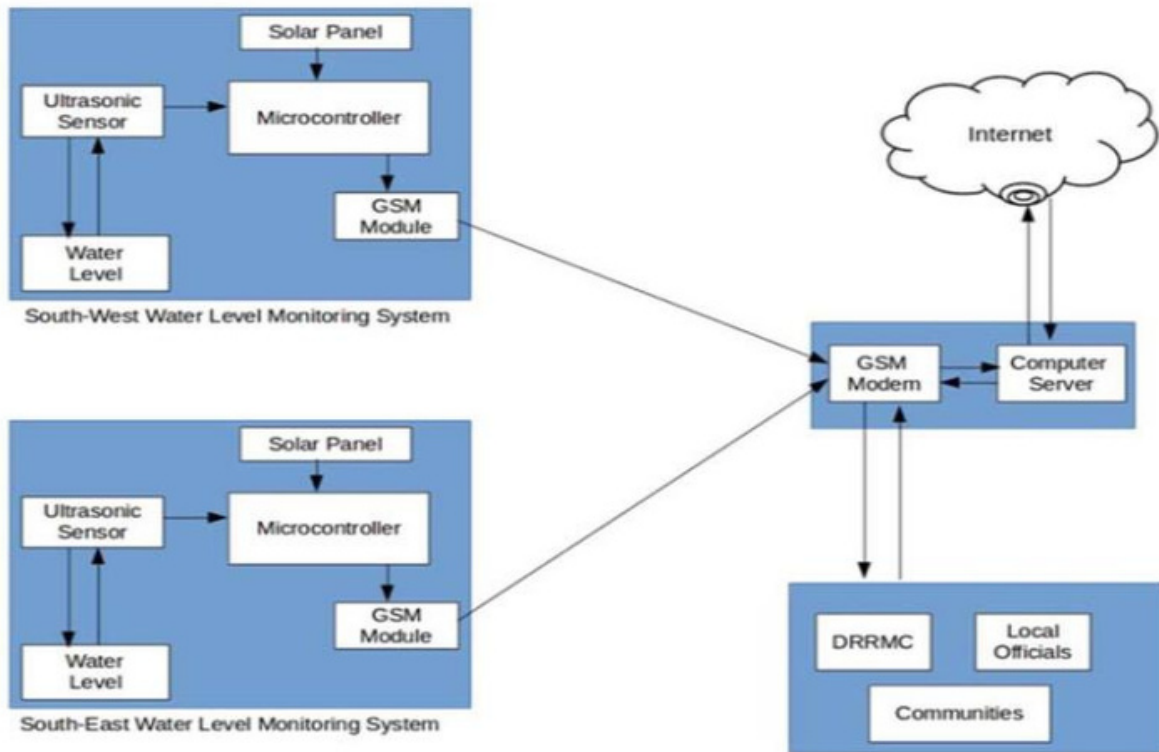


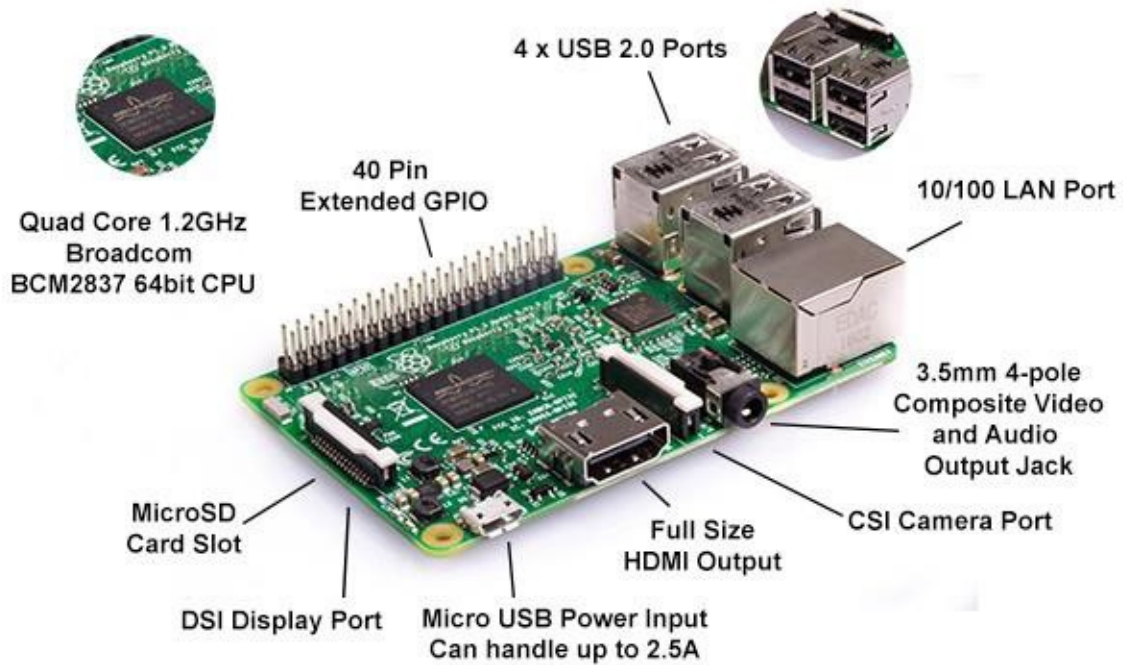
Fig 1.1 system architecture.



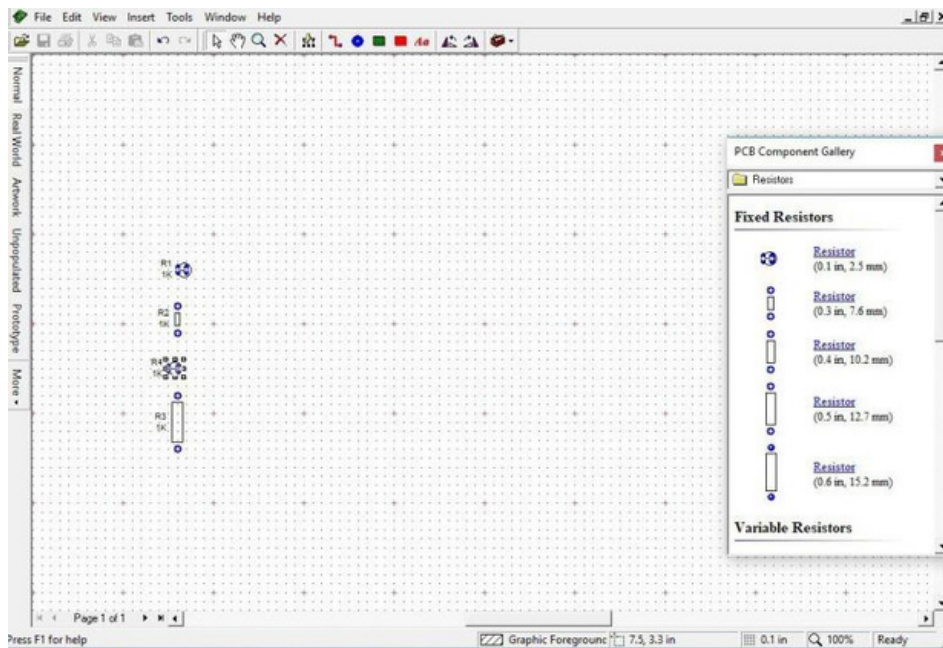
Fig 1.2 Geographical Map of Flooded Areas in the Northern Portion of Isabela.

This paper presents a project that is more localized to help the communities affected by flood in the province of Isabela particularly in the northern area by providing an interactive and real-time information on the current water level in the two major portions of the province. This project also widens the coverage of people that can receive the information to improve the emergency measures during floods.

Furthermore, this study builds a prototype that detects the current water level across the watershed of Cagayan River and its surrounding areas through ultrasonic sensors. The geographical area was sub-divided into two, where monitoring devices were installed. Specifically, the objectives of this study .

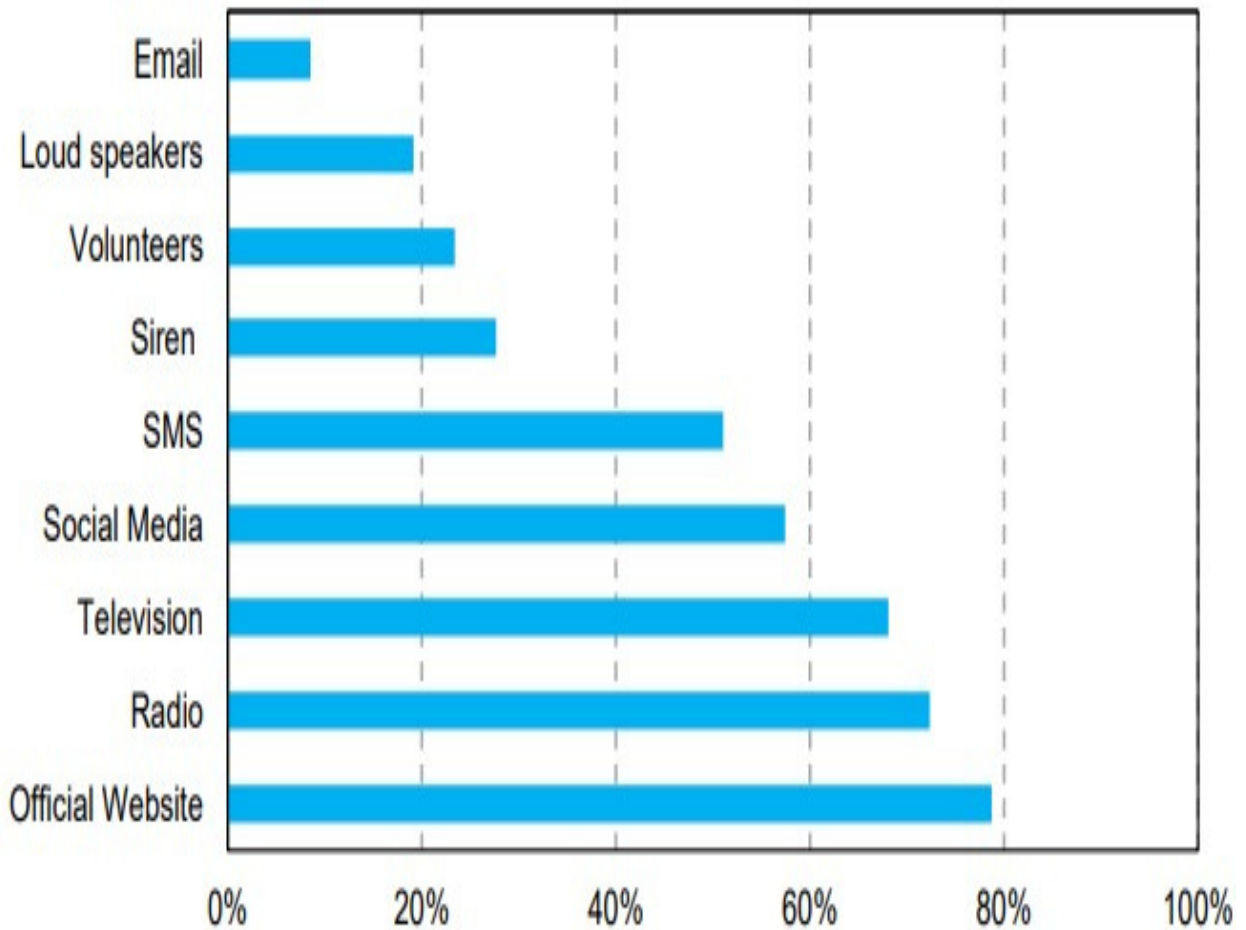


Ultrasonic sensor

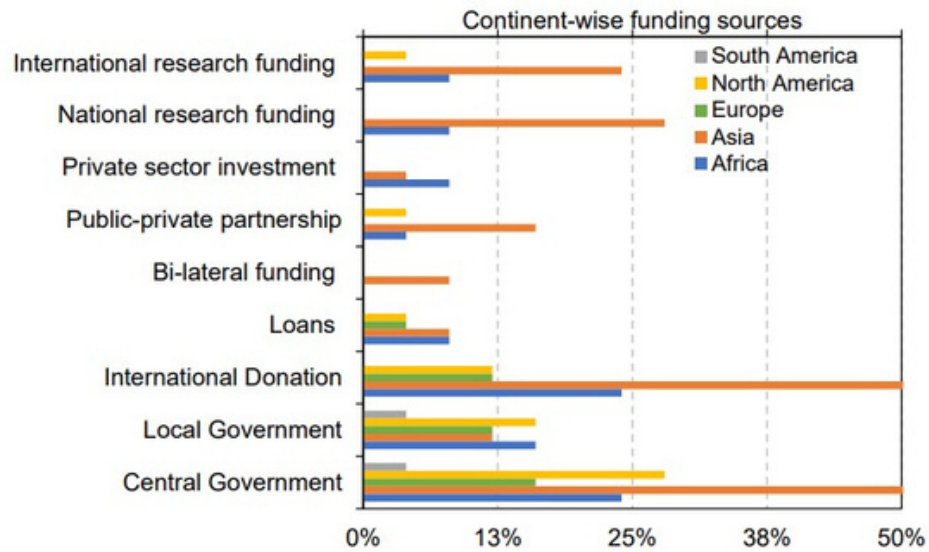


Software description

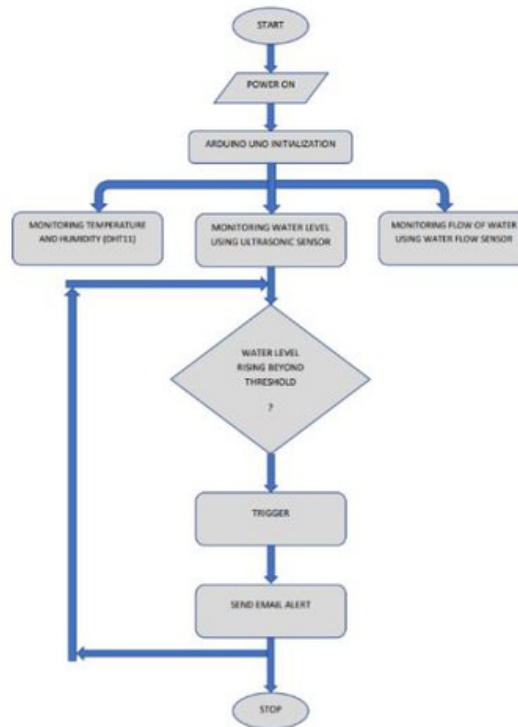
GRAPH REPRESENTATION:



Warning dissemination methods used by survey respondents (%)



Global (top) and continental (bottom) distribution of investments in FEWS



Flow Chart of the System

Overview of the project :

Ideally, the collected hydroclimatic data are stored in a database and processed in real time by hydrologic/hydraulic models. Not all countries have a centralized database that is fed continually up to date. The data are sometimes temporarily in spreadsheets to undergo quality control before being fed in the central database days or months later. One of the significant challenges faced by operational systems is the lack of technical expertise and human resources. Trained personnel with flood forecasting expertise and adequate forecast group staffing are required by the FFCs to issue timely warnings effectively. However, 74% of the flood forecasting personnel confirm that their centers do not have the experts and staff capable of integrating data, performing forecasts, and disseminating information [1].

SYSTEM IMPLEMENTATION :

PROJECT DESCRIPTION ;



Frontend python code :

```
import time
import machine
import dht

# Define GPIO pins
TRIG_PIN = machine.Pin(2, machine.Pin.OUT)
ECHO_PIN = machine.Pin(3, machine.Pin.IN)
BUZZER_PIN = machine.Pin(4, machine.Pin.OUT)
DHT_PIN = machine.Pin(5)
LED_PIN = machine.Pin(6, machine.Pin.OUT)

def distance_measurement():
    # Trigger ultrasonic sensor
    TRIG_PIN.on()
    time.sleep_us(10)
    TRIG_PIN.off()

    # Wait for echo to be HIGH (start time)
    while not ECHO_PIN.value():
        pass
    pulse_start = time.ticks_us()

    # Wait for echo to be LOW (end time)
    while ECHO_PIN.value():
```

```

pass

pulse_end = time.ticks_us()

# Calculate distance
pulse_duration = time.ticks_diff(pulse_end, pulse_start)
distance = pulse_duration / 58 # Speed of sound (343 m/s) divided by 2

return distance

def read_dht_sensor():
    d = dht.DHT22(DHT_PIN)
    d.measure()
    return d.temperature(), d.humidity()

buzz_start_time = None # To track when the buzzer started

while True:
    dist = distance_measurement()
    temp, humidity = read_dht_sensor()

    # Check if the distance is less than a threshold (e.g., 50 cm)
    if dist < 50:
        # Turn on the buzzer and LED
        BUZZER_PIN.on()
        LED_PIN.on()
        status = "Flooding Detected"
        buzz_start_time = time.ticks_ms()

        elif buzz_start_time is not None and time.ticks_diff(time.ticks_ms(), buzz_start_time) >=
        60000: # 1 minute

```

```

# Turn off the buzzer and LED after 1 minute
BUZZER_PIN.off()
LED_PIN.off()
status = "No Flooding Detected"
else:
status = "No Flooding Detected"

print(f"Distance: {dist:.2f} cm")
print(f"Temperature: {temp:.2f}°C, Humidity: {humidity:.2f}%")
print("Status:", status)

time.sleep(2)

```

BACKEND SOURCE CODE :

```

{
  "version": 1,
  "author": "Anonymous maker",
  "editor": "wokwi",
  "parts": [
    {
      "type": "board-pi-pico-w",
      "id": "pico",
      "top": -118.45,
      "left": 32.35,
      "attrs": { "env": "micropython-20231005-
v1.21.0" } },
    {

```

```

"type": "wokwi-hc-sr04",
"id": "ultrasonic1",
"top": -238.5,
"left": -138.5,
"attrs": { "distance": "257" }
},
{
"type": "wokwi-buzzer",
"id": "bz1",
"top": -180,
"left": -228.6,
"attrs": { "volume": "0.1" }
},
{ "type": "wokwi-dht22", "id": "dht1", "top": -268.5, "left": 167.4, "attrs": {} },
{ "type": "wokwi-led", "id": "led1", "top": -99.6, "left": -313, "attrs": { "color": "red" } },
{
"type": "wokwi-resistor",
"id": "r1",
"top": 33.6,
"left": -317.35,
"rotate": 90,
"attrs": { "value": "300" }
}
],
"connections": [
[ "ultrasonic1:TRIG", "pico:GP2", "blue", [ "v0" ] ],
[ "ultrasonic1:ECHO", "pico:GP3", "cyan", [ "v0" ] ],
[ "ultrasonic1:GND", "pico:GND.1", "black", [ "v0" ] ],

```

```
[ "bz1:2", "pico:GP4", "red", [ "v0" ] ],
[ "bz1:1", "pico:GND.2", "black", [ "v0" ] ],
[ "dht1:GND", "pico:GND.8", "black", [ "v0" ] ],
[ "dht1:SDA", "pico:GP5", "limegreen", [ "v0" ] ],
[ "ultrasonic1:VCC", "pico:3V3_EN", "red", [ "v0" ]
], [ "dht1:VCC", "pico:3V3_EN", "orange", [ "v0" ] ],
[ "led1:C", "pico:GP6", "yellow", [ "v0" ] ],
[ "led1:A", "r1:1", "magenta", [ "v0" ] ],
[ "r1:2", "pico:3V3", "magenta", [ "h0" ] ]
],
"dependencies": {}
}
```

SAMPLE CODE OUTPUT :

Screenshot image:

WOKWI SAVE SHARE Battery saver Save battery and browse longer SIGN IN

main.py diagram.json

```
1 import time
2 import machine
3 import dht
4
5 # Define GPIO pins
6 TRIG_PIN = machine.Pin(2, machine.Pin.OUT)
7 ECHO_PIN = machine.Pin(3, machine.Pin.IN)
8 BUZZER_PIN = machine.Pin(4, machine.Pin.OUT)
9 DHT_PIN = machine.Pin(5)
10 LED_PIN = machine.Pin(6, machine.Pin.OUT)
11
12 def distance_measurement():
13     # Trigger ultrasonic sensor
14     TRIG_PIN.on()
15     time.sleep_us(10)
16     TRIG_PIN.off()
17
18     # Wait for echo to be HIGH (start time)
19     while not ECHO_PIN.value():
20         pass
21     pulse_start = time.ticks_us()
22
23     # Wait for echo to be LOW (end time)
24     while ECHO_PIN.value():
25         pass
26     pulse_end = time.ticks_us()
27
28     # Calculate distance
29     pulse_duration = time.ticks_diff(pulse_end, pulse_start)
30     distance = pulse_duration / 58 # Speed of sound (343 m/s) divided by 2
```

Simulation

Start the simulation

WOKWI SAVE SHARE Battery saver Save battery and browse longer SIGN IN

main.py diagram.json

```
1 import time
2 import machine
3 import dht
4
5 # Define GPIO pins
6 TRIG_PIN = machine.Pin(2, machine.Pin.OUT)
7 ECHO_PIN = machine.Pin(3, machine.Pin.IN)
8 BUZZER_PIN = machine.Pin(4, machine.Pin.OUT)
9 DHT_PIN = machine.Pin(5)
10 LED_PIN = machine.Pin(6, machine.Pin.OUT)
11
12 def distance_measurement():
13     # Trigger ultrasonic sensor
14     TRIG_PIN.on()
15     time.sleep_us(10)
16     TRIG_PIN.off()
17
18     # Wait for echo to be HIGH (start time)
19     while not ECHO_PIN.value():
20         pass
21     pulse_start = time.ticks_us()
22
23     # Wait for echo to be LOW (end time)
24     while ECHO_PIN.value():
25         pass
26     pulse_end = time.ticks_us()
27
28     # Calculate distance
29     pulse_duration = time.ticks_diff(pulse_end, pulse_start)
30     distance = pulse_duration / 58 # Speed of sound (343 m/s) divided by 2
```

Simulation

Restart the simulation

Distance: 260.57 cm
Temperature: 24.00°C, Humidity: 40.00%
Status: No Flooding Detected

WOKWI SAVE SHARE ♥ Battery saver Save battery and browse longer Save battery Not now 1:333 789 SIGN IN

main.py diagram.json

Simulation

Restart the simulation

```

1 import time
2 import machine
3 import dht
4
5 # Define GPIO pins
6 TRIG_PIN = machine.Pin(2, machine.Pin.OUT)
7 ECHO_PIN = machine.Pin(3, machine.Pin.IN)
8 BUZZER_PIN = machine.Pin(4, machine.Pin.OUT)
9 DHT_PIN = machine.Pin(5)
10 LED_PIN = machine.Pin(6, machine.Pin.OUT)
11
12 def distance_measurement():
13     # Trigger ultrasonic sensor
14     TRIG_PIN.on()
15     time.sleep_us(10)
16     TRIG_PIN.off()
17
18     # Wait for echo to be HIGH (start time)
19     while not ECHO_PIN.value():
20         pass
21     pulse_start = time.ticks_us()
22
23     # Wait for echo to be LOW (end time)
24     while ECHO_PIN.value():
25         pass
26     pulse_end = time.ticks_us()
27
28     # Calculate distance
29     pulse_duration = time.ticks_diff(pulse_end, pulse_start)
30     distance = pulse_duration / 58 # Speed of sound (343 m/s) divided by :

```

Distance: 260.57 cm
Temperature: 24.00°C, Humidity: 40.00%
Status: No Flooding Detected

WOKWI SAVE SHARE ♥ Battery saver Save battery and browse longer Save battery Not now 1:799 849 SIGN IN

main.py diagram.json

Simulation

Restart the simulation

```

1 import time
2 import machine
3 import dht
4
5 # Define GPIO pins
6 TRIG_PIN = machine.Pin(2, machine.Pin.OUT)
7 ECHO_PIN = machine.Pin(3, machine.Pin.IN)
8 BUZZER_PIN = machine.Pin(4, machine.Pin.OUT)
9 DHT_PIN = machine.Pin(5)
10 LED_PIN = machine.Pin(6, machine.Pin.OUT)
11
12 def distance_measurement():
13     # Trigger ultrasonic sensor
14     TRIG_PIN.on()
15     time.sleep_us(10)
16     TRIG_PIN.off()
17
18     # Wait for echo to be HIGH (start time)
19     while not ECHO_PIN.value():
20         pass
21     pulse_start = time.ticks_us()
22
23     # Wait for echo to be LOW (end time)
24     while ECHO_PIN.value():
25         pass
26     pulse_end = time.ticks_us()
27
28     # Calculate distance
29     pulse_duration = time.ticks_diff(pulse_end, pulse_start)
30     distance = pulse_duration / 58 # Speed of sound (343 m/s) divided by :

```

Distance: 260.57 cm
Temperature: 24.00°C, Humidity: 40.00%
Status: No Flooding Detected
Distance: 260.64 cm
Temperature: 24.00°C, Humidity: 40.00%
Status: No Flooding Detected
Distance: 260.64 cm
Temperature: 24.00°C, Humidity: 40.00%
Status: No Flooding Detected

CONCLUSION AND FUTURE ENHANCEMENT:

IoT sensors-based flood monitoring systems tend to be lower cost, consistent and portable. However, when there are large areas, these systems are not recommended due to the fact that every sensor is generally invigorated by a vitality restricted battery. This paper reviewed and clarified different ecological and flood monitoring systems and various communication technologies that support enhancing the detection of viable floods and identifying cautioning issues. Further, these systems that are having highly reliable sensors with powerful IoT cloud platforms can be fundamentally utilized for large-scale environmental monitoring, and flood prediction and prevent damage caused by it. Even though the methodology of utilizing IoT in flood monitoring is not extensively explored at this point, we will see a colossal utilization of IoT and some new advancements in the near future. For example, AI and 5G techniques meet up for the prediction of floods as well as other natural calamities. The use of satellite images could be very helpful in flood monitoring as they help to keep an eye on the water bodies and the change in their behaviour from above. Some researchers have utilized data based on Google Maps to build a detection model. GSM modules also have been used in different ways similarly. Close consultation with hydrologists and learning machine-learning algorithms can further support building efficient monitoring and alert system. In the future, the usage of SAR data from the Sentinel-1 satellite is an added advantage in handling rescue operations and damage assessments based on data before and after floods. The wireless sensors can help in gathering flood related data by creating a database for further analysis. As a recommendation, there is a tremendous opportunity to explore the combination of IoT systems and SAR data to classify the images from floodprone areas and develop robust and secure Flood monitoring and early warning system

REFERENCE :

[1] Wesley Mendes-Da-Silva, Eduardo Flores, David L. Eckles. Informe DDecisions Regarding Flood Events Induces Propensity for Insurances ,Environmental Science&Policy.Men

(2) P. Jain, B. Schoen-Phelan, and R. Ross, "Automatic flood detection in Sentinel-2 images using deep convolutional neural networks," in Proceedings of the 35th Annual ACM Symposium on Applied Computing, Brno Czech Republi

(3) Parveez K. A Smart Zigbee Based Wireless Weather Station Monitoring System. In: ICCCE 2012. Proceedings of the International Conference on Computing and Control Engineering; April 12-13; Chennai. Coimbatore. Coimbatore Institute of Information Technology; 2012.

(4) Jamali A, Abdul Rahman A. Flood Mapping using Synthetic Aperture Radar: A Case Study of Ramsar Flash Flood. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. 2019;

(5) Yawut C, Kilaso S. A wireless sensor network for weather and disaster alarm systems. In: IPCSIT. Proceedings of the International Conference on Information and Electronics Engineering; May 28-29; Bangkok. Singapore: IACSIT Press. 2011.

(6) Udo E. N, Isong E. B. Flood monitoring and detection system using wireless sensor network. Asian Journal of Computer and Information Systems.