ARASU ENGINEERING COLLEGE-KUMBAKONAM

DEPARTMENT OF BIOMEDICAL ENGINEERING

FLOOD MONITORING & EARLY WARNING SYSTEM

TEAM MEMBERS: S. AARTHI (820621121001)

M. DEVISRI (820621121004)

T. DHANUSIYA (820621121005)

A. DIVYA (820621121006)

V. HARINI (820621121007)

A. NIVITHA (820621121008)

P. YOGAPRIYA (820621121012)

FACULTY: B. VANMATHI

Table of contents

Abstract

The IoT-based Flood Monitoring System presented in this study addresses the critical need for real-time flood detection and early warning systems to mitigate the devastating impacts of flooding events. Leveraging the power of Internet of Things (IoT) technology, the system incorporates an array of sensors strategically deployed in flood-prone areas. These sensors continuously monitor water levels, rainfall intensity, and other relevant environmental parameters. Data collected by the sensors are transmitted wirelessly to a central server for processing and analysis.

The system employs advanced data analytics algorithms to assess the collected data in real-time, enabling the accurate and timely detection of potential flood events. Upon detection, the system initiates automated alerts via various communication channels, including SMS, email, and mobile applications, ensuring that relevant authorities and affected communities receive timely notifications. Additionally, the system provides a user-friendly interface for visualization and analysis of historical and real-time flood data, aiding in decision-making processes for disaster management.

Key features of the system include scalability, low power consumption, and adaptability to diverse geographical and climatic conditions. Furthermore, the integration of machine learning techniques allows for the refinement of flood prediction models over time, enhancing the system's accuracy and reliability. The proposed Flood Monitoring System offers a cost-effective and efficient solution for monitoring and managing flood-related risks, ultimately contributing to increased resilience in vulnerable regions. This system holds great potential for reducing flood-related casualties and property damage, thereby safeguarding lives and livelihoods in flood-prone areas.

introduction

Floods represent one of the most devastating natural disasters, causing widespread damage to communities, infrastructure, and the environment. Rapid urbanization, climate change, and changing weather patterns have heightened the frequency and intensity of flooding events, underscoring the urgent need for effective flood monitoring and early warning systems. In response to this critical challenge, the integration of Internet of Things (IoT) technology has emerged as a promising solution.

This paper introduces an innovative IoT-based Flood Monitoring System designed to revolutionize the way we detect and respond to flood events. By leveraging the capabilities of IoT, this system provides real-time data collection and analysis, enabling timely and accurate flood detection. Through strategically placed sensors in flood-prone areas, the system continuously monitors essential parameters such as water levels, rainfall intensity, and other relevant environmental conditions.

The collected data is transmitted wirelessly to a central server, where advanced data analytics algorithms process and interpret the information in real-time. This allows for the early identification of potential flood events, triggering automated alerts to relevant authorities and communities through multiple communication channels. Additionally, the system offers a user-friendly interface for visualizing and analyzing historical and real-time flood data, empowering decision-makers in disaster management.

This Flood Monitoring System exhibits key attributes such as scalability, low power consumption, and adaptability to various geographical and climatic contexts. Furthermore, the integration of machine learning techniques ensures continuous improvement of flood prediction models, enhancing the system's accuracy and reliability over time. Through this innovative approach, the system aims to significantly enhance preparedness, response, and resilience in flood-prone regions, ultimately safeguarding lives and livelihoods from the devastating impacts of flooding events.

Problem statement

Floods constitute one of the most destructive natural disasters, causing extensive damage to communities, infrastructure, and the environment. The escalating frequency and intensity of these events, exacerbated by factors such as rapid urbanization and climate change, necessitate a paradigm shift in flood monitoring and early warning systems. Traditional methods often lack the agility and accuracy required to provide timely alerts, leaving vulnerable populations exposed to significant risks.

Conventional monitoring systems rely on sporadic data collection, leading to delayed response times and limited capacity to predict and mitigate flood events. Additionally, many existing systems are characterized by high maintenance costs, limited scalability, and inadequate adaptability to diverse geographical and climatic conditions. This results in gaps in coverage, leaving numerous flood-prone areas underserved and ill-prepared for imminent threats.

Furthermore, the lack of integrated, real-time data analysis hampers the ability to make informed decisions during critical moments. The absence of automated alert systems exacerbates response delays, increasing the likelihood of casualties and property damage. Additionally, the absence of user-friendly interfaces for data visualization and analysis impedes effective collaboration between stakeholders involved in disaster management.

Addressing these challenges is imperative to minimize the human, economic, and environmental toll of flooding events. A comprehensive Flood Monitoring System leveraging Internet of Things (IoT) technology offers a promising solution to revolutionize flood detection, early warning, and disaster preparedness. This system aims to bridge existing gaps in monitoring capabilities, enhance data accuracy, and empower communities and authorities with timely and actionable information, ultimately mitigating the devastating impacts of floods on vulnerable regions.

Design thinking

Real-time flood monitoring:

1. Continuous Data Collection: Establish a system for real-time acquisition of critical data, including water levels, rainfall intensity, and weather conditions.

2. Early Warning System: Develop a mechanism to provide timely and accurate warnings to at-risk communities, allowing them to take necessary precautions.

3. Sensor Deployment and Network Setup: Implement a network of sensors strategically placed in flood-prone areas to ensure comprehensive coverage.

4. Data Processing and Analysis: Create algorithms to process incoming data and generate actionable insights, including flood forecasts, trends, and potential impacts.

5. Integration with GIS and Mapping Tools: Enable visualization of real-time flood data on geographical information systems (GIS), allowing for precise location-based assessments.

6. Alerting and Communication Protocols: Establish clear protocols for disseminating alerts to relevant stakeholders, such as local authorities, emergency services, and affected communities.

7. Accuracy and Reliability Optimization: Ensure that the monitoring system provides accurate and reliable data, even under challenging environmental conditions.

8. Scalability and Expansion: Design the system to accommodate future expansion, allowing for the inclusion of additional monitoring points or the integration of new technologies.

9. Resilience to System Failures: Implement backup and redundancy measures to guarantee the system's continued operation in the event of hardware or network failures.

10. Community Engagement and Education: Develop outreach programs to educate local communities about flood risks, safety measures, and how to interpret and respond to real-time alerts.

11. Historical Data Management: Create a database for storing and analyzing historical flood data, enabling trend analysis, long-term planning, and impact assessments.

12. Environmental Impact Assessment: Include features to monitor and assess the environmental impact of floods, aiding in post-flood recovery efforts and environmental restoration.

13. Integration with Emergency Response Services: Establish seamless communication channels with emergency services to ensure swift and coordinated responses during flood events.

14. Regulatory Compliance and Standards: Ensure that the system adheres to all relevant government regulations and standards related to flood monitoring and warning systems.

15. Cost-effectiveness and Sustainability: Strive to develop a system that is economically viable for deployment and operation, potentially incorporating energy-efficient technologies and sustainable practices.

16. User Training and Support: Provide comprehensive training materials and ongoing support for users, including local authorities, emergency responders, and community members, to maximize the effectiveness of the system.

Tailor these objectives to fit the specific needs, geography, and climate conditions of the area where the real-time flood monitoring system will be implemented.

Early warning issuance:

1. Timely Alert Generation: Develop a system that can generate warnings well in advance of potential flood events to provide communities with sufficient time to prepare.

2. Data Integration and Analysis: Implement mechanisms to gather and process real-time data from various sources, including rainfall, water levels, weather forecasts, and river/stream gauges.

3. Threshold Determination: Define precise thresholds for different flood scenarios based on historical data and local topography, ensuring accurate and context-specific warnings.

4. Customizable Alert Levels: Allow for different levels of alerts based on the severity of the flood threat, enabling tailored responses for varying situations.

5. Communication Channels: Establish reliable and redundant communication channels to disseminate warnings to relevant stakeholders, including local authorities, emergency services, and affected communities.

6. User-friendly Interface: Design an intuitive user interface that allows for easy monitoring of data and issuance of alerts, ensuring accessibility for both technical and non-technical users.

7. Geospatial Mapping and Visualization: Provide a clear visual representation of flood-prone areas on a map, aiding in effective decision-making and response coordination.

8. Integration with GIS and Mapping Tools: Enable integration with geographical information systems (GIS) for precise location-based assessments and visualization.

9. Validation and Verification Mechanisms: Implement checks to validate the accuracy and reliability of incoming data, reducing the likelihood of false alarms.

10. Feedback Loop for Continuous Improvement: Establish mechanisms for feedback from users and communities to improve the accuracy and effectiveness of the early warning system over time.

11. Community Engagement and Education: Develop outreach programs to educate local communities about flood risks, safety measures, and how to interpret and respond to early warnings.

12. Training for Users and Stakeholders: Provide comprehensive training materials and ongoing support for users, including local authorities, emergency responders, and community members.

13. Resilience to System Failures: Implement backup and redundancy measures to ensure the system's continued operation in the event of hardware or network failures.

14. Compliance with Regulatory Standards: Ensure that the early warning issuance system adheres to all relevant government regulations and standards related to flood monitoring and warning systems.

15. Cost-effectiveness and Sustainability: Strive to develop a system that is economically viable for deployment and operation, potentially incorporating energy-efficient technologies and sustainable practices.

Remember to adopt these objectives based on the specific needs and circumstances of the area where the early warning issuance system will be implemented.

Public safety:

1. Early Warning and Alert System: Develop a system that provides timely and accurate warnings to at-risk communities, allowing them to take necessary precautions well in advance of flood events.

2. Community Outreach and Education: Implement outreach programs to educate local communities about flood risks, safety measures, and how to respond effectively to warnings.

3. Evacuation Planning and Coordination: Establish protocols for coordinated evacuations, including designated safe zones, transportation arrangements, and communication channels.

4. Rescue and Relief Operations Coordination: Create mechanisms for efficient coordination between emergency services, including rescue teams, medical personnel, and other relevant agencies.

5. Special Needs Considerations: Account for the needs of vulnerable populations, such as the elderly, disabled, and those without access to transportation, in evacuation and relief planning.

6. Shelter and Provision Management: Establish procedures for setting up and managing temporary shelters, ensuring they are equipped with necessary supplies and facilities.

7. Communication Channels: Implement reliable and redundant communication channels to disseminate warnings and instructions to affected communities and emergency responders.

8. User-friendly Interface for Emergency Responders: Design an intuitive interface for emergency responders to quickly access critical information and respond effectively during flood events.

9. Medical and Health Services Integration: Coordinate with healthcare facilities and providers to ensure continuity of care for individuals with medical needs during floods.

10. Search and Rescue Capabilities: Equip emergency services with the necessary resources and training for effective search and rescue operations in flooded areas.

11. Post-Flood Recovery Planning: Develop plans and resources for post-flood recovery efforts, including debris removal, infrastructure repair, and restoration of basic services.

12. Psychosocial Support Services: Provide resources and training for mental health professionals to offer support to individuals and communities affected by flooding.

13. Environmental Impact Assessment and Mitigation: Include features to monitor and assess the environmental impact of floods, aiding in post-flood recovery efforts and environmental restoration.

14. Regulatory Compliance and Standards: Ensure that public safety measures align with relevant government regulations and standards related to flood monitoring and emergency response.

15. Community Feedback and Continuous Improvement: Establish mechanisms for gathering feedback from communities and emergency responders to continuously improve public safety measures in flood monitoring.

Remember to adopt these objectives based on the specific needs and circumstances of the area where the flood monitoring system focused on public safety will be implemented.

Emergency response coordination:

1. Timely Alert Dissemination: Develop a system that rapidly delivers accurate and timely flood alerts to relevant emergency response teams and agencies.

2. Real-time Data Integration: Implement mechanisms to collect and integrate real-time data from various sensors and sources, including rainfall, water levels, weather conditions, and river/stream gauges.

3. Resource Mobilization and Deployment: Establish protocols for efficiently deploying emergency response resources, including personnel, equipment, and supplies, to affected areas.

4. Coordination with Local Authorities: Foster strong communication and coordination between emergency response teams and local government authorities for effective decision-making and resource allocation.

5. Multi-Agency Collaboration: Facilitate seamless collaboration among multiple agencies involved in emergency response, including fire departments, police, medical teams, and search and rescue teams.

6. Incident Command System (ICS) Implementation: Adopt an incident command system to establish a clear hierarchy and chain of command for managing response efforts.

7. Evacuation Planning and Execution: Develop and implement evacuation plans, including identifying safe routes, establishing evacuation centers, and ensuring transportation for vulnerable populations.

8. Medical and Health Services Integration: Coordinate with healthcare facilities and providers to ensure timely medical care and support for individuals with special needs during flood events.

9. Search and Rescue Operations: Provide resources, training, and equipment for effective search and rescue operations in flooded areas, including swift-water rescue techniques.

10. Communication Infrastructure Resilience: Ensure that communication channels remain robust and reliable even in adverse weather conditions or during network disruptions.

11. Information Sharing and Situational Awareness: Implement systems for sharing critical information and maintaining situational awareness among all response teams involved in the operation.

12. Logistical Support and Supply Chain Management: Establish procedures for managing logistics, including supply procurement, distribution, and maintenance of emergency response equipment.

13. Post-Flood Recovery and Restoration: Develop plans and resources for post-flood recovery efforts, including debris removal, infrastructure repair, and restoration of basic services.

14. Training and Exercises for Emergency Responders: Conduct regular training sessions and exercises to ensure that emergency responders are well-prepared and familiar with protocols.

15. Community Engagement and Support: Involve and inform affected communities in the emergency response process, providing them with guidance and support during and after flood events.

Remember to adapt these objectives based on the specific needs and circumstances of the area where the flood monitoring system with a focus on emergency response coordination will be implemented.

Innovation

Incorporating IoT (Internet of Things) into flood monitoring systems can significantly enhance their effectiveness. Here are some innovative ideas:

1. Multi-Sensor Integration: Utilize various types of sensors (water level, rainfall, weather, soil moisture) to gather comprehensive data for more accurate flood predictions.

2. Machine Learning Algorithms: Implement ML models to analyze historical data and provide more precise flood forecasts, considering various environmental factors.

3. Real-time Data Transmission: Ensure swift and continuous data transmission from sensors to a central monitoring system for timely alerts and decision-making.

4. Predictive Analytics: Employ advanced analytics to anticipate potential flood-prone areas based on historical data, weather patterns, and other relevant parameters.

5. Crowdsourced Data: Allow citizens to contribute data through mobile apps or IoT devices, providing additional information for flood prediction and response.

6. Drone Technology: Use drones equipped with sensors to monitor flood conditions in remote or hard-to-reach areas, providing valuable insights to emergency responders.

7. Automated Response Systems: Implement automated protocols for deploying resources (such as sandbags or rescue teams) based on real-time data.

8. Smart Infrastructure: Integrate IoT into urban infrastructure (e.g., smart buildings, flood barriers) to enhance resilience and adaptability during flood events.

9. AI-powered Image Recognition: Develop systems that can analyze images or video feeds to identify flood-related issues, such as blocked drainage systems.

10. Community Engagement Platforms: Create platforms for communities to access real-time flood data, receive alerts, and contribute information, fostering a collaborative approach to flood management.

11. Blockchain for Data Security: Use blockchain to secure and validate data from IoT sensors, ensuring the integrity and authenticity of flood-related information.

12. Energy-efficient Sensors: Develop sensors that are powered by renewable energy sources or have low energy consumption to extend their operational lifespan.

13. Dynamic Risk Assessment: Utilize IoT data to dynamically assess risk levels in different areas, allowing for adaptive response strategies as flood conditions change.

14. Integration with Emergency Services: Establish seamless communication channels between IoT systems and emergency services for efficient coordination during flood events.

15. Public Awareness Campaigns: Leverage IoT to disseminate real-time information to the public through various channels (smartphones, social media, public displays) to promote safety and preparedness.

Remember, while these ideas offer innovative approaches, it's crucial to consider factors like scalability, cost-effectiveness, and local regulations when implementing any IoT-based flood monitoring system. Additionally, involving local communities

and stakeholders in the design and deployment process can lead to more effective and sustainable solutions.

Development of part 1

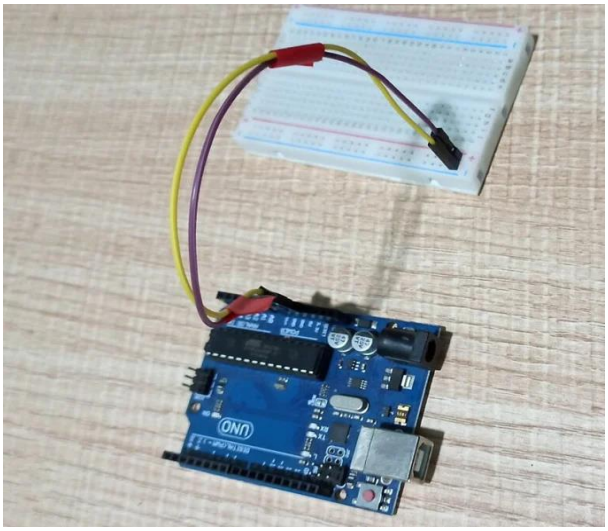
HARDWARE COMPONENTS -

- Bolt-IoT Wi-Fi module
- Arduino uno
- Breadboard- 400 tie points
- 5mm LED:(Green, Red, Orange) and Buzzer
- 16×2 LCD Display
- LM35 Temperature Sensor
- HC-SR04 Ultrasonic Sensor
- Some Jumper Wires
    - Male to Female Jumper Wires- 15 pcs
    - Male to Male Jumper Wires- 10 pcs
    - Female to Female Jumper Wires- 5 pcs
- 9v Battery and Snap Connector
- USB Cable Type B

HARDWARE SETUP:

For Building this project we first configure the hardware connections. Then later on moving to the software part.

Step 1: Connecting 5v and GND of Arduino to the Breadboard for power connection to other components.



Step 2: Connecting LED's

For Green LED:

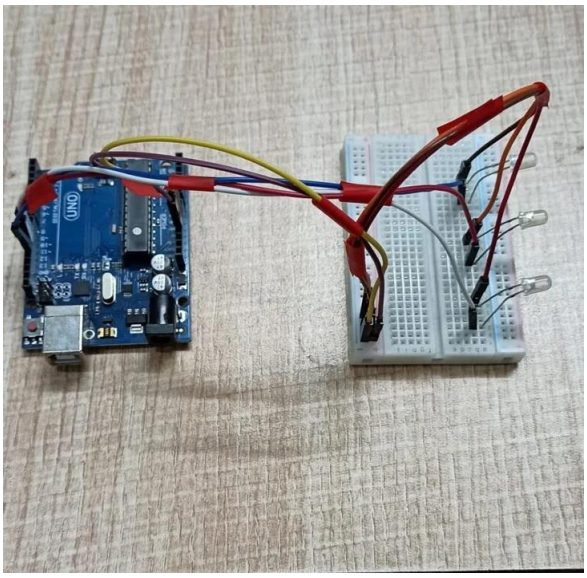
- VCC of Green Color LED to Digital Pin '10' of the Arduino.
- GND of Green Color LED to the GND of Arduino.

For Orange LED:

- VCC of Orange Color LED to Digital Pin '11' of the Arduino.
- GND of Orange Color LED to the GND of Arduino.

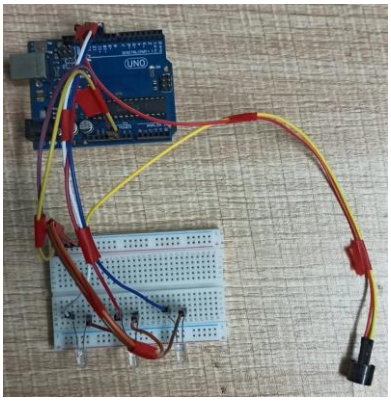
For Red LED:

- VCC of Red Color LED to Digital Pin '12' of the Arduino.
- GND of Red Color LED to the GND of Arduino.



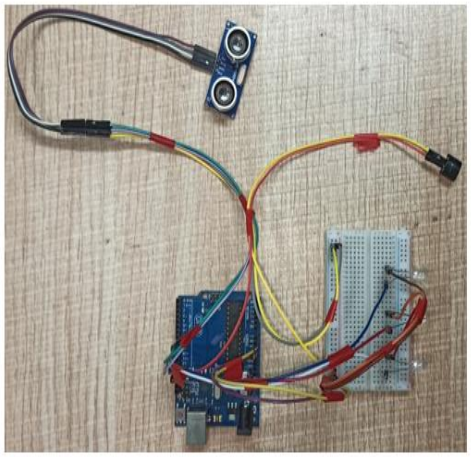
Step 3: Connecting Buzzer

- VCC of Buzzer to Digital Pin ‘13’ of the Arduino.
- GND of Buzzer to the GND of Arduino.



Step 4: Connecting HC-SR04 Ultrasonic Sensor

- VCC of Ultrasonic Sensor to 5v of Arduino.
- GND of Ultrasonic Sensor to GND of Arduino.
- Echo of Ultrasonic Sensor to Digital Pin ‘8’ of Arduino.
- Trig of Ultrasonic Sensor to Digital Pin ‘9’ of Arduino.

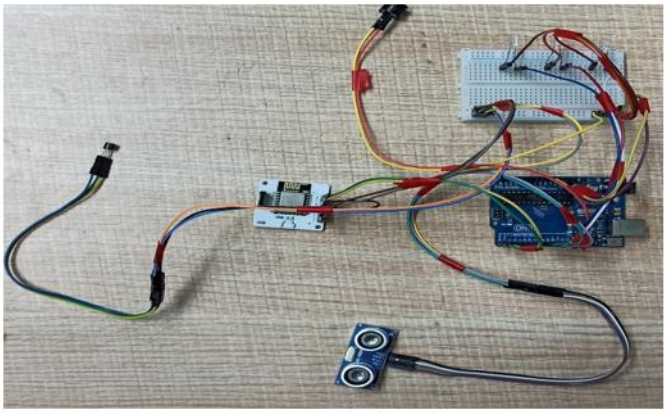


Step 5: Connecting Bolt Wi-Fi Module

- 5v of Bolt Wi-Fi Module to 5v of Arduino.
- GND of Bolt Wi-Fi Module to GND of Arduino.
- TX of Bolt Wi-Fi Module to RX of Arduino.
- RX of Bolt Wi-Fi Module to TX of Arduino.

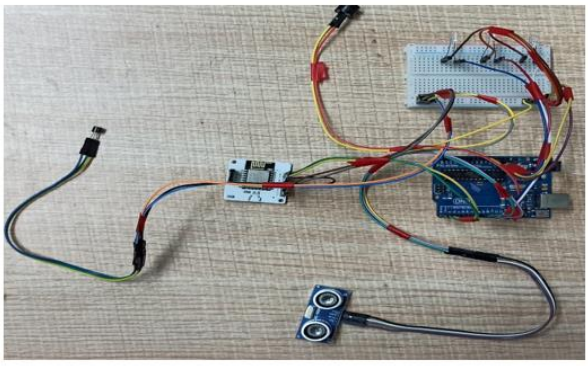
Step 6: Connecting LM35 Temperature Sensor

- VCC of LM35 to 5v of Bolt Wi-Fi Module.
- Output Pin of LM35 to Pin ‘A0’ of Bolt Wi-Fi Module.
- GND of LM35 to GND of Bolt Wi-Fi Module.

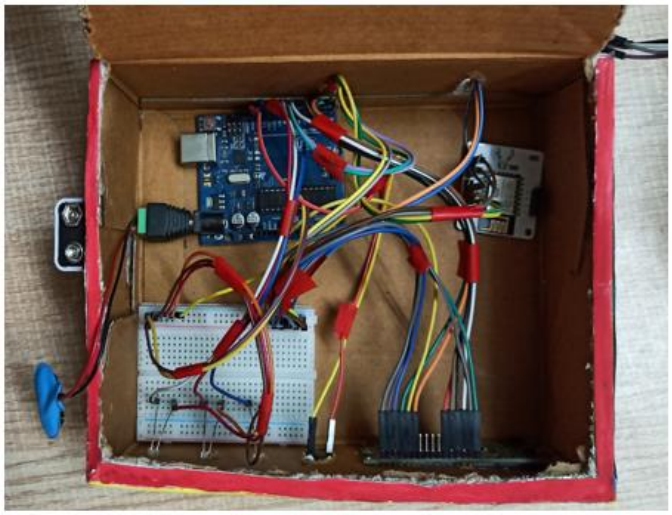


Step 7: Connecting 16×2 LCD Display

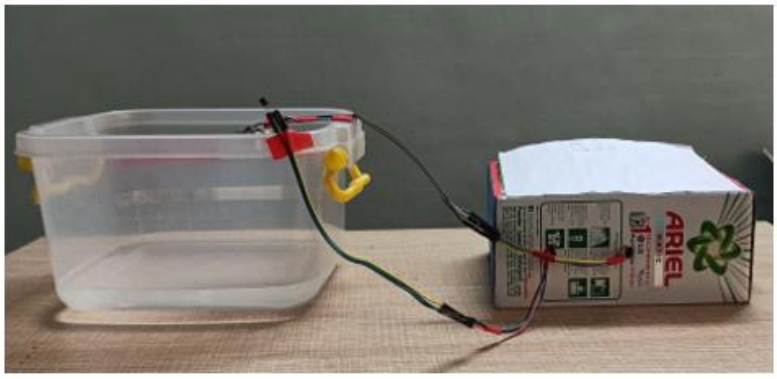
- Pin 1,3,5,16 of 16×2 LCD to GND of Arduino.
- Pin 2,15 of 16×2 LCD to 5v of Arduino.
- Pin 4 of 16×2 LCD to Digital Pin ‘2’ of Arduino.
- Pin 6 of 16×2 LCD to Digital Pin ‘3’ of Arduino.
- Pin 11 of 16×2 LCD to Digital Pin ‘4’ of Arduino.
- Pin 12 of 16×2 LCD to Digital Pin ‘5’ of Arduino.
- Pin 13 of 16×2 LCD to Digital Pin ‘6’ of Arduino.
- Pin 14 of 16×2 LCD to Digital Pin ‘7’ of Arduino.



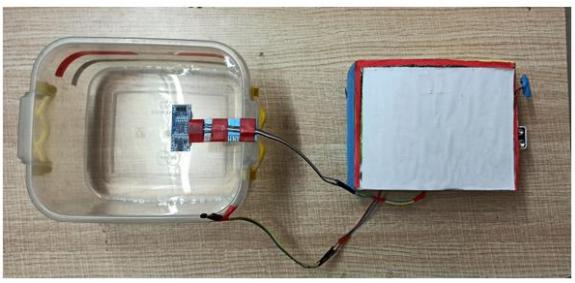
After making the hardware connection put all the hardware components in one box.



Also attach LM35 Temperature Sensor on the side of the container.



Also attach Ultrasonic sensor on the top of the container.



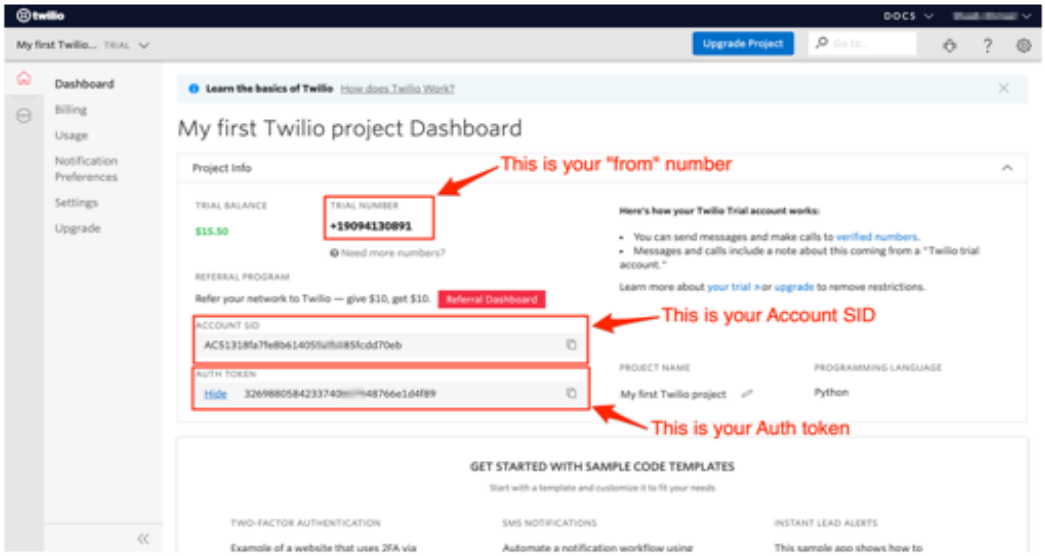
Development of part 2

SOFTWARE PROGRAMMING:

After the successful completion of hardware setup. Now it's time to do software setup for the project. For that you have to first Download and Install Arduino IDE and Python IDE from the link given above in the software apps and online services section. Also Creating accounts on various online app services and noting down the important keys and ids. Below are all the steps given to create an account on online app services and note down the keys.

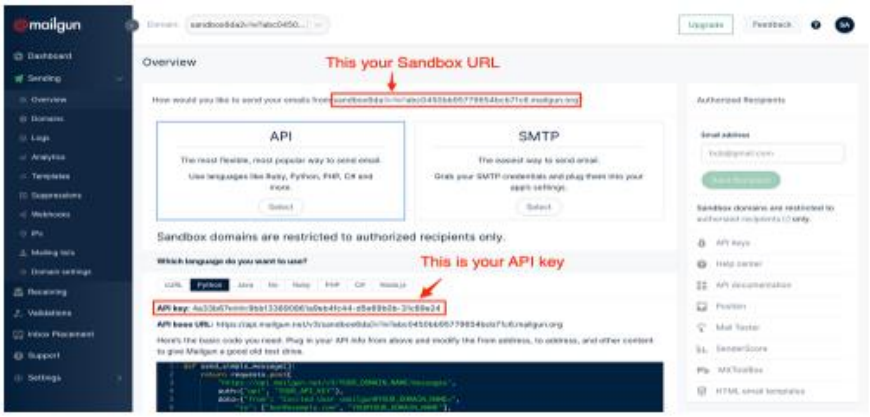
Step 1: Creating an account on Twilla and setting up Twilla for sending SMS alerts.

- Visit https://www.twilio.com/ .
- Create an account by clicking sign up, fill required details.
- Confirm your email.
- You will need to authenticate your phone number on which the sms alerts will be notified.
- Enter the code sent to your phone
- When prompted" Do you write code?" Click yes
- Select python as your programming language
- When prompted "What is your target today? "Choose" Twilio as a project.
- When prompted "What do you want to do first? "Choose" Send or receive a message.
- My First Twilio Project Dashboard page will open. Now you can Edit your Project as "My Project".
- Get a trial number and save it somewhere and then choose to use this number.
- You will see the ACCOUNT SID and AUTH TOKEN.
- We will need Account Sid, Auth Token and Trial Number of these so save them somewhere.



Step 2: Creating an account on Mailgun and setting up Mailgun for sending Email alerts.

- Visit https://www.mailgun.com/.
- Create an account by clicking on the start sending option and by filling in details.
- Verifying your Account.
- Once you have verified your Email, after that you have added your phone number.
- After Entering your number. Click on send activation code. After some time, you will receive an OTP. Enter the OTP. Click on Enter.
- After Creating account on Mailgun go to the overview option. Click on API and Click on Python.
- After doing this you will receive an API Key and Sandbox URL. Save both these credentials somewhere you will be further using in this project.



Step 4: Creating an account on Bolt Cloud and Bolt Android App and Link the Bolt Module to Cloud.

- Visit https://cloud.boltiot.com.
- Create an account using Email-Id and password. (Use the same email which was used to order hardware kit also use same email for app for linking the hardware to cloud.)
- After creating an account on cloud. Then Download Bolt Android App from play store.
- Create an account on the Bolt app with the same email-Id then use the mobile hotspot for linking the Bolt Wi-Fi module to cloud.
- After successfully linking the device to the cloud then go to the cloud website. The Bolt device will show the device as online.
- Go to API section make the API as enable. Copy the API and save it somewhere.
- Also copy the Bolt Device Id which is present on Bolt IoT dashboard and save it somewhere.

Bolt Device Id

API Key

Step 5: Coding

After setting online app services and saving the keys somewhere. Now the most important thing is to write code and allow sensors attached to microcontroller to take specific decisions.

Basically, this project contains two editors to write the code. First is Arduino IDE in that we will write the Arduino code. Second the Python IDE in that we will write the configuration file and the main code. Also, the download link of the editor can find above in the online app services section.

Step 5.1: Writing the code in the Arduino IDE

- Open the Arduino IDE (Downloaded from the above section).
- Click on new file. Choose the correct file path to save the file. Give appropriate name to the file and add. into extension to the file and save the file.
- Now the core part of the project is writing code for Arduino Uno. Below this line complete code is given. You can refer the below code.

```
//IOT Based Flood Monitoring And Alerting System.
#include<LiquidCrystal.h>

LiquidCrystal lcd(2, 3, 4, 5, 6, 7);

const int in = 8;
const int out = 9;
const int green = 10;
const int orange = 11;
const int red = 12;
const int buzz = 13;

void setup() {
Serial.begin(9600);
lcd.begin(16, 2);
pinMode( in , INPUT);
pinMode(out, OUTPUT);
pinMode(green, OUTPUT);
pinMode(orange, OUTPUT);
pinMode(red, OUTPUT);
pinMode(buzz, OUTPUT);
digitalWrite(green, LOW);
```

```
pinMode(buzz, OUTPUT);
digitalWrite(green, LOW);
digitalWrite(orange, LOW);
digitalWrite(red, LOW);
digitalWrite(buzz, LOW);
lcd.setCursor(0, 0);
lcd.print("Flood Monitoring");
lcd.setCursor(0, 1);
lcd.print("Alerting System");
delay(5000);
lcd.clear();
}

void loop() {
long dur;
long dist;
long per;
digitalWrite(out, LOW);
delayMicroseconds(2);
digitalWrite(out, HIGH);
delayMicroseconds(10);
digitalWrite(out, LOW);
```

```
delayMicroseconds(10);
digitalWrite(out, LOW);
dur = pulseIn( in , HIGH);
dist = (dur * 0.034) / 2;
per = map(dist, 10.5, 2, 0, 100);
#map
function is used to convert the distance into percentage.
if(per < 0) {
per = 0;
}
if (per > 100) {
per = 100;
}
Serial.println(String(per));
lcd.setCursor(0, 0);
lcd.print("Water Level:");
lcd.print(String(per));
lcd.print("% ");
if (per >= 80) #MAX Level of Water--Red Alert!{
lcd.setCursor(0, 1);
lcd.print("Red Alert! ");
```

```
digitalWrite(green, LOW);
digitalWrite(orange, LOW);
digitalWrite(buzz, HIGH);
delay(2000);
digitalWrite(buzz, LOW);
delay(2000);
digitalWrite(buzz, HIGH);
delay(2000);
digitalWrite(buzz, LOW);
delay(2000);

}
else if (per >= 55) #Intermedite Level of Water--Orange Alert!{
lcd.setCursor(0, 1);
lcd.print("Orange Alert! ");
digitalWrite(orange, HIGH);
digitalWrite(red, LOW);
digitalWrite(green, LOW);
digitalWrite(buzz, HIGH);
delay(3000);
digitalWrite(buzz, LOW);
delay(3000);
```

```
lcd.print("Orange Alert! ");
digitalWrite(orange, HIGH);
digitalWrite(red, LOW);
digitalWrite(green, LOW);
digitalWrite(buzz, HIGH);
delay(3000);
digitalWrite(buzz, LOW);
delay(3000);

}
else #MIN / NORMAL level of Water--Green Alert!{
lcd.setCursor(0, 1);
lcd.print("Green Alert! ");
digitalWrite(green, HIGH);
digitalWrite(orange, LOW);
digitalWrite(red, LOW);
digitalWrite(buzz, LOW);
}

delay(15000);
}
```

- After writing the code. Verify the code and then upload the code to the specific Arduino using USB Cable type A. Remember while uploading select specific board you want to upload.

Step 5.2: Writing the code in Python IDE.

- For writing python code, we will be using python IDE.
- In this project we will be making two python files. One will be saved in the name of conf.py and the other will be main.py.
- The purpose of making two files is to make the code understandable. Also, both these python files will be useful in sending SMS and emails alerts to users.
- Now the most important part is writing code in Python IDE. The full code is divided into two parts. The detailed code is given below.
- Open Python 3.7 IDE (Downloaded from the above section).
- Click on new file. Save the file in the name conf.py.
- **conf.py:** The file consists of important Api keys, Device id of Bolt IoT Wi-Fi Module. Also, it consists of important keys of Twilla and Mailgun respectively which will be further useful in this project.
- Below is the complete structure of conf.py file. Make sure that you add the updated Bolt API key, device id and Mailgun and Twillo details respectively:

```
#twillo details for sending alert sms
SID = 'You can find SID in your Twilio Dashboard'
AUTH_TOKEN = 'You can find  on your Twilio Dashboard'
FROM_NUMBER = 'This is the no. generated by Twilio. You can fin
TO_NUMBER = 'This is your number. Make sure you are adding +91

#bolt iot details
API_KEY = 'XXXXXXXXX'
   #This is your Bolt cloud API
Key.
DEVICE_ID = 'BOLTXXXXXXXXX' #This is the ID of your Bolt device

#mailgun details for sending alert E-mails
MAILGUN_API_KEY = 'This is the private API key which you can fi
SANDBOX_URL= 'You can find this on your Mailgun Dashboard'
SENDER_EMAIL = 'test@ + SANDBOX_URL' # No need to modify this.
RECIPIENT_EMAIL = 'Enter your Email ID Here'
```

- After writing the conf.py now the last part is to write the main.py code. This code will be helpful to send SMS and email alerts when the water level crosses the threshold.
- Open the Python IDE.
- Click on new file. Save the file in the name main.py. Save the file in the same path where conf.py is saved.
- **main.py:** This file consists of the main coding facility. Discussed earlier, it will be used to send SMS and emails alerts. It will be also helpful to keep a close monitor of water levels to send alerts whenever required.
- Below is the complete code of main.py.

```
import conf
from boltiot import Sms, Email, Bolt
import json, time

intermediate_value = 55
max_value = 80


mybolt = Bolt(conf.API_KEY, conf.DEVICE_ID)
sms = Sms(conf.SID, conf.AUTH_TOKEN, conf.TO_NUMBER, conf.FROM_
mailer = Email(conf.MAILGUN_API_KEY, conf.SANDBOX_URL, conf.SEN


def twillo_message(message):
  try:
     print("Making request to Twilio to send a SMS")
     response = sms.send_sms(message)
     print("Response received from Twilio is: " + str(response)
     print("Status of SMS at Twilio is :" + str(response.status
  except Exception as e:
     print("Below are the details")
```

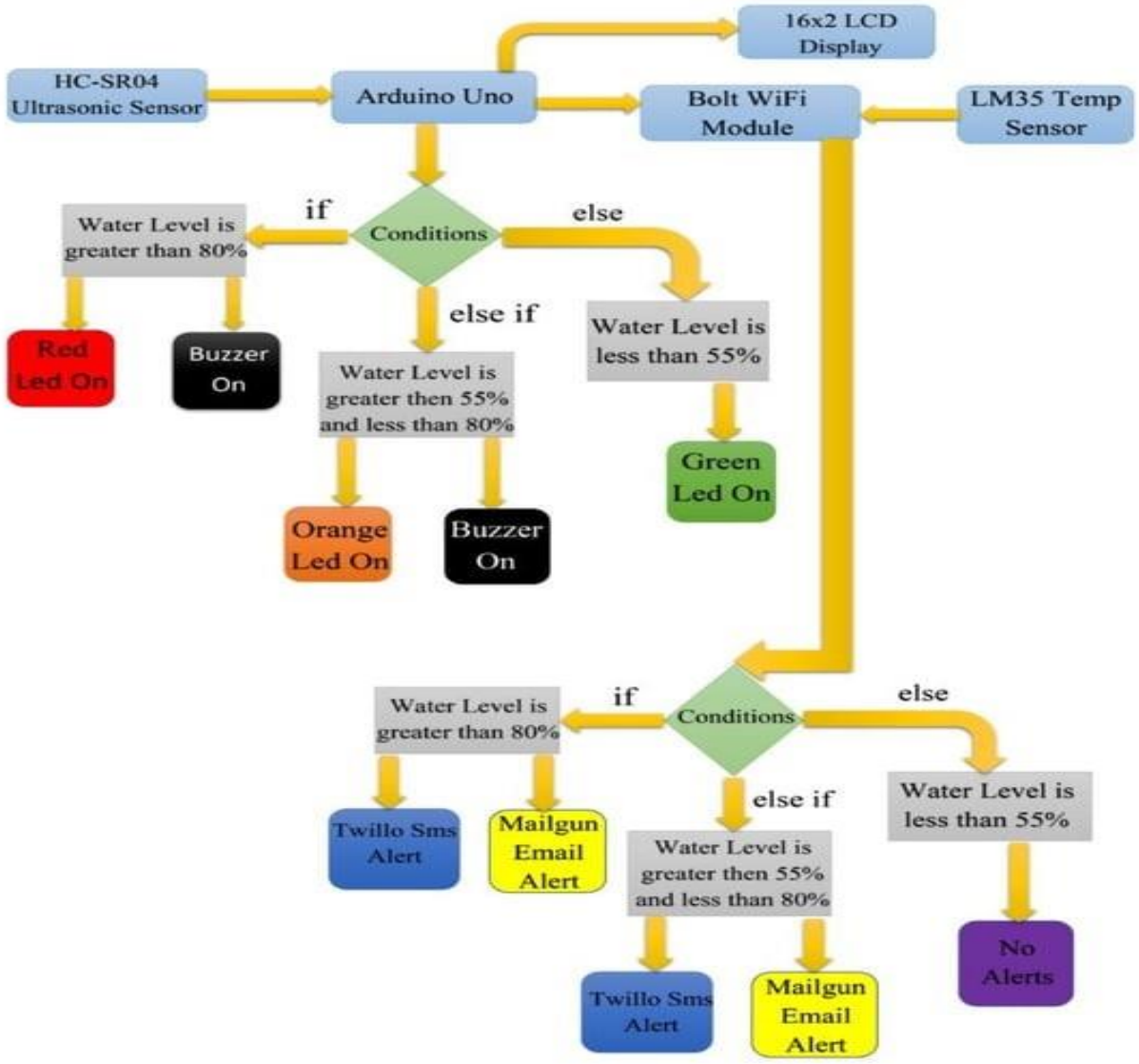
```
  try:
     print("Making request to Mailgun to send an email")
     response = mailer.send_email(head,message_1)
     print("Response received from Mailgun is: " + response.tex
  except Exception as e:
     print("Below are the details")
     print(e)

while True:
    print ("Reading Water-Level Value")
    response_1 = mybolt.serialRead('10')
    response = mybolt.analogRead('A0')
    data_1 = json.loads(response_1)
    data = json.loads(response)
    Water_level = data_1['value'].rstrip()
    print("Water Level value is: " + str(Water_level) + "%")
    sensor_value = int(data['value'])
    temp = (100*sensor_value)/1024
    temp_value = round(temp,2)
    print("Temperature is: " + str(temp_value) + "°C")
    try:
```

```
try:

    if int(Water_level) >= intermediate_value:
        message ="Orange Alert!. Water level is increased by
        head="Orange Alert"
        message_1="Water level is increased by " + str(Wate
        twillo_message(message)
        mailgun_message(head,message_1)

    if int(Water_level) >= max_value:
       message ="Red Alert!. Water level is increased by "
       head="Red Alert!"
       message_1="Water level is increased by " + str(Water
       twillo_message(message)
       mailgun_message(head,message_1)

except Exception as e:
    print ("Error occured: Below are the details")
    print (e)
time.sleep(15)
```

After Successfully writing code for Arduino and Python. Now it is the time to test and demonstrate the project. Move to the next section for a demonstration of the project.

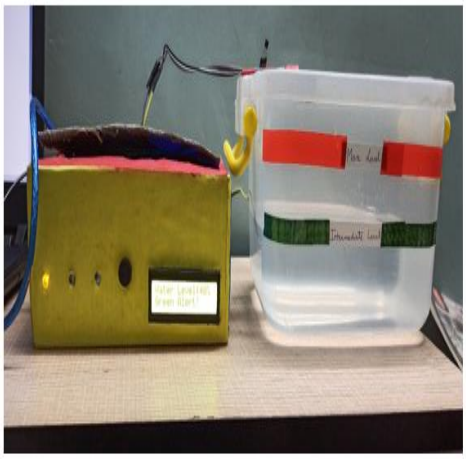
Demonstration

Let's First have a look at the workflow of this project.

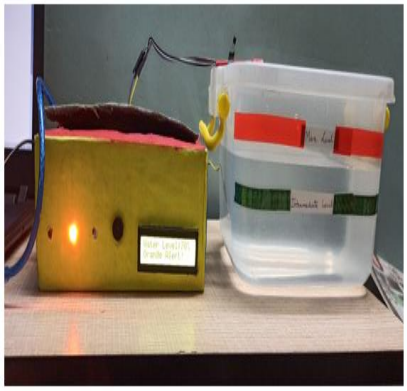


For doing the practical demonstration. First connect the USB cable type-B to the Laptop's USB slot for power supply. Also simultaneously run the python program (i.e., Main.py). Firstly, the ultrasonic sensor will sense the water level in distance and then the Arduino program will help to convert it into percentage. Also, the sensed water level will be displayed on Lcd display (In Percentage) along with zone/area the water level is present. The full water tank/container is divided into 3 zones i.e., Green, Orange and Red. Now let's investigate each zone.

- When the water level is at Min/Normal level. That resembles ‘Green Alert’. This means that the water is at a normal position and there is no sign of flood condition. Also, green lead will glow, and it will also show green alert in Lcd display with water level.

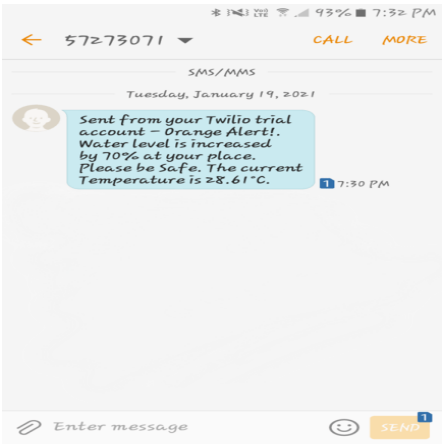


- When the water level crosses the Intermediate level. That resembles ‘Orange Alert’. This means that water has crossed the 55% mark and there can be chances of flood condition at that place. With the increase in water level the system sends SMS and Email alerts to the authority or registered user from Twillo and Mailgun Services respectively. Also, orange lead will glow, and buzzer will buzz. It will also show orange alert in Lcd display.



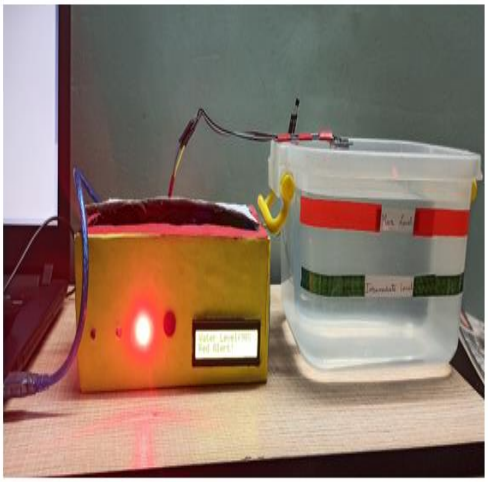
Also, SMS and Email is sent to registered user with proper message and current temperature of that place.

Twillo SMS Alert:

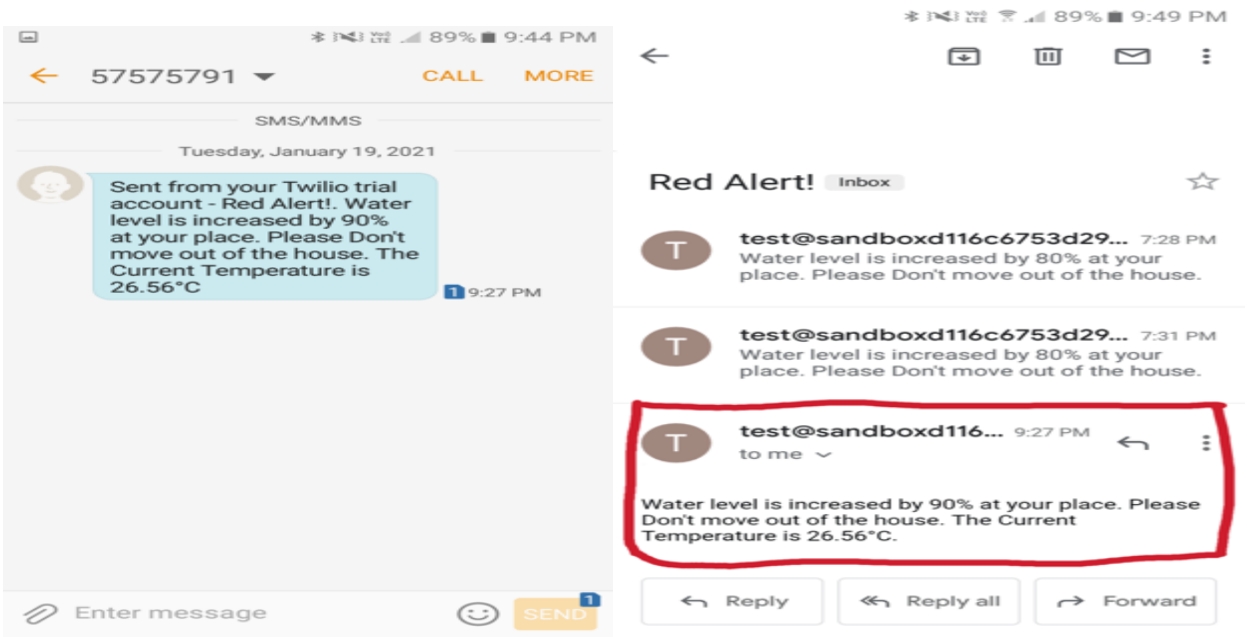


Mailgun Email Alert:

- When the water level crosses the Max Level. That resembles ‘Red Alert’. This means that the water level has exceeded 80% and a flood situation has occurred at that place. With the increase in water level the system sends SMS and Email alerts to the authority or registered user from Twillo and Mailgun Services respectively. Also red lead will glow, and buzzer will buzz two times. It will also show red alert in Lcd display.



Also SMS and Email is sent to registered user with proper message and current temperature of that place.



Advantages

The flood monitoring system using IoT (Internet of Things) offers several advantages:

1. Real-Time Monitoring: It provides real-time data on water levels, allowing for timely response and intervention in case of a flood.

2. Early Warning Systems: IoT sensors can detect rising water levels and send alerts, giving residents and authorities more time to prepare and evacuate if necessary.

3. Data Accuracy: Sensors provide precise measurements, reducing the margin of error associated with manual monitoring.

4. Cost-Effective: Compared to traditional monitoring methods, IoT-based systems can be more cost-effective in the long run, as they require less manpower and resources.

5. Remote Accessibility: Data can be accessed remotely via the internet, allowing for monitoring and management from anywhere with an internet connection.

6. Integration with Other Systems: IoT devices can be integrated with existing disaster management systems, enhancing the overall efficiency of response efforts.

7. Customizable Thresholds: Sensors can be programmed to trigger alerts based on specific water level thresholds, allowing for tailored responses to different flood scenarios.

8. Historical Data and Trend Analysis: IoT systems can store historical data, enabling the analysis of flood patterns and trends over time. This information can be invaluable for long-term planning and mitigation strategies.

9. Reduced Response Time: With automated alerts and real-time data, response times can be significantly reduced, potentially saving lives and minimizing property damage.

10. Environmental Impact Assessment: IoT flood monitoring systems can aid in assessing the environmental impact of floods, helping with post-disaster recovery and restoration efforts.

11. Scalability: The system can be easily scaled to cover larger areas or be integrated with other monitoring networks for a more comprehensive view of flood-prone regions.

12. Energy Efficiency: Many IoT devices are designed to be energy-efficient, which helps in maintaining continuous monitoring without significant power consumption.

Overall, the use of IoT in flood monitoring brings about a more efficient and effective approach to disaster management, ultimately enhancing public safety and minimizing the impact of floods on communities.

Disadvantages

While flood monitoring systems using IoT offer numerous advantages, they also come with some potential disadvantages:

1. Cost of Implementation: Setting up an IoT-based flood monitoring system can be initially expensive due to the cost of acquiring and installing sensors, as well as the necessary communication infrastructure.

2. Maintenance and Upkeep: IoT devices require regular maintenance and may need to be replaced or updated over time, incurring additional costs.

3. Dependence on Internet Connectivity: The system relies on a stable internet connection. In areas with unreliable or limited connectivity, this can pose a significant challenge.

4. Power Dependency: Many IoT sensors require a power source. In areas prone to power outages, ensuring continuous operation may require backup power solutions, which can add to the overall cost.

5. Data Security and Privacy Concerns: Transmitting sensitive flood data over the internet can raise concerns about privacy and security. Measures must be in place to safeguard the information from unauthorized access or cyberattacks.

6. Limited Coverage Area: The effectiveness of the system is contingent on the density of deployed sensors. In less densely populated or remote areas, achieving comprehensive coverage may be challenging.

7. Sensor Calibration and Accuracy: IoT sensors need to be accurately calibrated to provide reliable data. Incorrect measurements or sensor malfunctions can lead to false alarms or missed warnings.

8. Environmental Factors: Harsh environmental conditions, such as extreme temperatures, flooding, or physical damage, can impact the performance and longevity of IoT devices.

9. Integration Challenges: Integrating IoT systems with existing infrastructure or coordinating with other emergency response mechanisms can be complex and may require significant coordination and planning.

10. Technological Obsolescence: Rapid advancements in technology can lead to the obsolescence of certain components or systems, potentially necessitating costly upgrades.

11. Ethical Considerations: There may be ethical concerns related to the collection, storage, and use of data, especially when it involves private properties or sensitive locations.

12. Over-Reliance on Technology: In some cases, there might be a tendency to rely solely on technology for flood monitoring, potentially leading to complacency or a lack of preparedness for alternative scenarios.

It's important to carefully consider these disadvantages when implementing an IoT-based flood monitoring system and take steps to mitigate potential challenges.

Conclusion

In conclusion, the implementation of a flood monitoring system using IoT technology presents a transformative approach to disaster management. The numerous advantages, including real-time monitoring, early warning systems, data accuracy, and remote accessibility, demonstrate its potential to significantly enhance public safety and reduce the impact of floods on communities. The system's ability to provide historical data, integrate with existing infrastructure, and scale for broader coverage further solidifies its effectiveness.

However, it's crucial to acknowledge the associated challenges. These include initial implementation costs, ongoing maintenance requirements, dependency on reliable internet connectivity, and potential privacy and security concerns. Overcoming these hurdles necessitates careful planning, investment, and a commitment to robust cybersecurity measures.

Despite these considerations, the benefits outweigh the drawbacks, making IoT-based flood monitoring systems a valuable tool in modern disaster preparedness and response efforts. With continuous advancements in technology and a commitment to addressing potential limitations, these systems have the potential to save lives, protect property, and foster more resilient communities in the face of flood-related challenges.