KGiSL INSTITUTE OF TECHNOLOGY

(Approved By AICTE, New Delhi, Affiliate to Anna University

Recognized by UGC, Accredited by NBA(IT)

265, KGISL Campus, Thudiyalur Road, Saravanampatti, Coimbatore-641035.)

DEPARTMENT OF

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

NAAN MUDHALVAN - INTERNET OF THINGS

FLOOD MONITORING AND EARLY WARNING

NAME: AAKASH B D

REG NO: 711721243001

NM ID: au711721243001

MAIL ID: aakash.d@kgkite.ac.in

TEAM MENTOR: Mr. Mohankumar M

TEAM EVALUATOR: Ms. Akilandeeshwari M

Phase 5: Project Documentation & Submission

Problem Statement:

The problem at hand is to create an effective and integrated flood monitoring and early warning system for flood-prone regions. This system must incorporate IoT sensor technology for real-time data collection, employ accurate data analysis algorithms to predict potential floods, and implement a responsive notification system to swiftly alert both the public and relevant authorities. The ultimate objective is to mitigate flood-related risks, reduce property damage, and save lives by significantly enhancing public safety and emergency response coordination in areas susceptible to flooding.

PROCEDURE:

Step 1: Define Requirements

Clarify project objectives and key features.

Step 2: Hardware Setup

Select IoT sensors.

Install sensors and ensure connectivity.

Step 3: Data Transmission and Collection

Define data transmission protocol.

Set up data processing on a server or cloud service.

Step 4: Data Storage and Management

Create a database for data storage.

Implement data cleansing and transformation.

Step 5: Real-time Data Analysis

Develop algorithms for real-time data analysis.

Step 6: API Development

Build APIs for mobile app data access.

Step 7: Mobile App Development

Design a user-friendly mobile app.

Integrate with IoT data.

Step 8: User Testing and Feedback

Thoroughly test the system.

Gather user feedback for improvements.

Step 9: Deployment and Maintenance

Deploy the system.

Implement maintenance and updates.

Step 10: Scalability and Optimization

Plan for system scalability.

Continuously optimize the system for better performance.

IOT REQUIRMENTS:

1. **IoT Sensors:**
    - Water Level Sensors: Deploy sensors that measure water levels in rivers, streams, and flood-prone areas.
    - Weather Sensors: Use sensors to collect data on rainfall, temperature, humidity, wind speed, and atmospheric pressure.
    - GPS Sensors: Implement GPS for accurate geographical location tracking of each sensor.
2. **Data Transmission:**
    - Wireless Communication: Utilize reliable and secure wireless communication protocols (e.g., Wi-Fi, LoRa, NB-IoT, or cellular) for transmitting data from sensors to the central platform.
    - Real-Time Data: Ensure real-time or near-real-time data transmission to enable timely flood monitoring and alerts.
3. **Power Supply:**

- Battery Backup: Equip sensors with battery backup systems to ensure continuous operation during power outages.
- Solar Panels: Incorporate solar panels to recharge batteries and extend sensor lifespan.

4. **Data Quality and Accuracy:**
    - Data Validation: Implement data validation and quality control mechanisms to filter out erroneous data.
    - Calibration: Regularly calibrate sensors to maintain data accuracy.
5. **Scalability:**
    - Design the system to be scalable, allowing for the addition of more sensors as the network expands.
6. **Security:**
    - Secure Data Transmission: Encrypt data in transit to protect it from interception.
    - Authentication: Implement authentication mechanisms to ensure that only authorized devices can connect to the network.
7. **Data Storage:**
    - Choose a robust and scalable database system (e.g., MySQL or NoSQL) for storing the **collected sensor data.**
8. **Data Analysis:**
    - Develop data analysis algorithms that can process incoming data to detect patterns and predict potential floods.
9. **Early Warning System:**
    - Implement a notification system that generates alerts for the public and relevant authorities when flood risks are detected.
    - Utilize multiple communication channels (e.g., SMS, email, mobile apps) to ensure alerts reach a wide audience.
10. **User Interface:**
    - Create a user-friendly web-based interface for users to access sensor data, analysis results, and early warnings.
11. **Redundancy and Reliability:**
    - Include redundancy in the system's components and communication channels to ensure reliability.
    - Implement failover mechanisms to maintain operation during system failures.

12.Regulatory Compliance:

- Ensure compliance with local, state, and national regulations and standards related to data privacy, safety, and emergency notification.

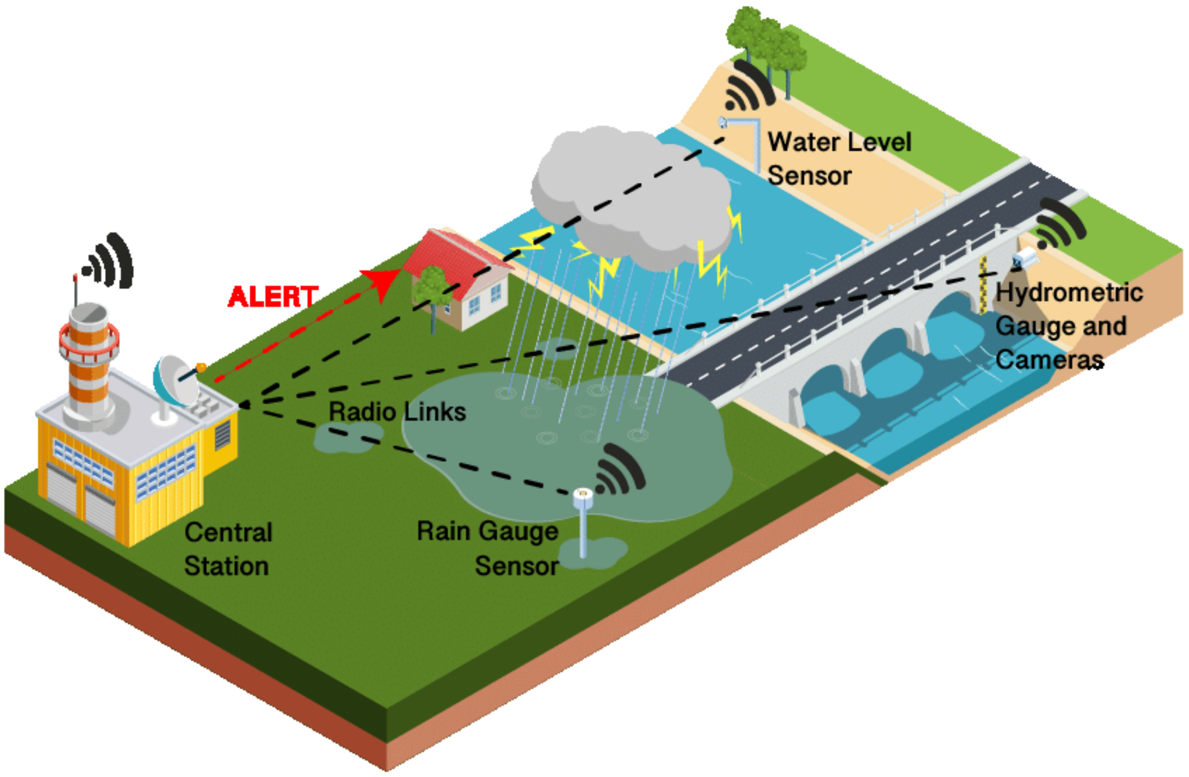
13.Maintenance and Support:

- Plan for regular maintenance and support to keep the IoT sensors and the entire system in good working order.

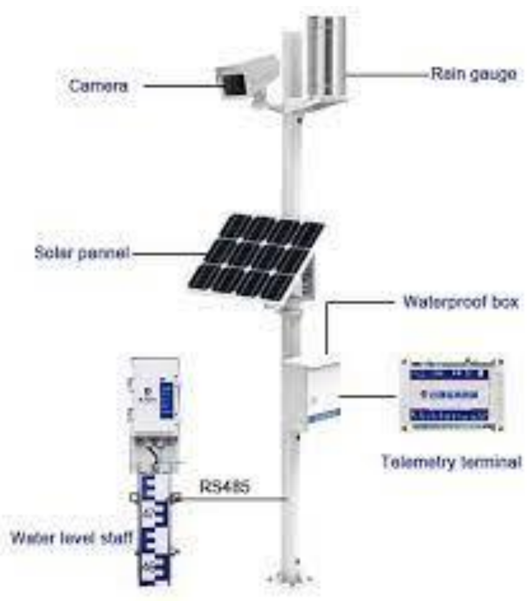
CODE:

```
import RPi.GPIO as GPIO
import time
import smtplib
from email.mime.text import MIMEText

# Set up GPIO pins for the water level sensor
TRIG = 23
ECHO = 24

GPIO.setmode(GPIO.BCM)
GPIO.setup(TRIG, GPIO.OUT)
GPIO.setup(ECHO, GPIO.IN)

# Function to measure the water level
def measure_water_level():
    GPIO.output(TRIG, True)
    time.sleep(0.00001)
    GPIO.output(TRIG, False)

    while GPIO.input(ECHO) == 0:
        pulse_start = time.time()

    while GPIO.input(ECHO) == 1:
        pulse_end = time.time()

    pulse_duration = pulse_end - pulse_start
    distance = pulse_duration * 17150  # Speed of sound = 34300 cm/s

    return distance

# Function to send email alerts
def send_email_alert(subject, message):
    sender_email = 'your_email@gmail.com'
    sender_password = 'your_password'
    receiver_email = 'recipient_email@gmail.com'

    msg = MIMEText(message)
    msg['Subject'] = subject
    msg['From'] = sender_email
    msg['To'] = receiver_email

    try:
        server = smtplib.SMTP('smtp.gmail.com', 587)
        server.starttls()
        server.login(sender_email, sender_password)
        server.sendmail(sender_email, receiver_email, msg.as_string())
        server.quit()
        print("Email sent successfully")
    except Exception as e:
        print("Error sending email:", str(e))

# Main loop
while True:
    try:
        water_level = measure_water_level()

        if water_level < 20:  # Adjust this threshold based on your sensor and needs
            alert_subject = "Flood Warning"
            alert_message = "Water level is high. Potential flood!"
            send_email_alert(alert_subject, alert_message)

        time.sleep(600)  # Check water level every 10 minutes
    except KeyboardInterrupt:
        break

GPIO.cleanup()
```

PROJECT SCREENSHOTS:



Picture of flood monitoring and early warning device



FLOOD MONITORING DEVICE

CONCLUSION:

In conclusion, Flood Monitoring and Early Warning systems are indispensable tools in mitigating the devastating impacts of floods. These innovative solutions leverage technology, data, and community engagement to enhance public safety, minimize property damage, and foster long-term resilience in the face of climate-related disasters. By providing timely alerts and promoting preparedness, these systems not only save lives but also serve as cornerstones for sustainable development. As the world faces increasing climate uncertainties, the continued advancement and implementation of these systems are imperative for the well-being and security of communities worldwide.