INTERIM PROGRESS REPORT

on

IOT BASED FLOOD MONITORING SYSTEM

Submitted by

ARUNKUMAR S - RA2211004010039
BHABIL HARITHRA A - RA2211004010050
PADMESH M.S - RA2211004010056

Semester – V
Academic Year: 2024–25

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

College of Engineering and Technology ,
SRM Institute of Science and Technology
SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamil Nadu.

ABSTRACT

In recent years, climate change and unpredictable weather patterns have led to a significant increase in flood-related disasters. This project presents an IoT-based flood monitoring and detection system designed to provide real-time alerts and data on water levels, flow rate, and environmental conditions. The system utilizes the ESP8266 NodeMCU WiFi module, which serves as the primary controller and communication device, transmitting data to a centralized monitoring platform. The integration of an ultrasonic sensor, water flow sensor, and a DHT11 temperature and humidity sensor allows for comprehensive flood detection, where water levels, flow speed, and environmental factors are continuously monitored.

The ultrasonic sensor measures water levels, while the water flow sensor tracks the rate of water movement, providing early warnings of potential flooding. The DHT11 sensor captures ambient temperature and humidity data, which is crucial for understanding conditions that may precede heavy rainfall and flooding. Through the ESP8266 NodeMCU, the collected data is sent to a cloud-based platform, enabling remote monitoring and timely notifications to authorities and residents in flood-prone areas. This real-time flood detection and alert system aims to reduce response time, minimize damage, and potentially save lives by leveraging IoT technology for disaster preparedness and response.

OBJECTIVE

Real-time Monitoring: Continuously monitor water levels using ultrasonic sensors and detect any rise above pre-set thresholds.

Flow Rate Measurement: Measure water flow rates in real-time to help assess flooding severity and rate of water rise.

Environmental Data Collection: Capture temperature and humidity data to analyze environmental conditions that may indicate impending flood risks.

Data Transmission: Utilize the ESP8266 NodeMCU's WiFi capability to send data wirelessly to a central server or cloud platform for remote monitoring.

Alert Generation: Provide timely alerts to concerned authorities and residents through the cloud interface or mobile app upon detecting flood conditions.

Data Logging and Analysis: Maintain a historical log of water levels, flow rates, and environmental data to analyze patterns and enhance future flood prediction accuracy.

CHAPTER 1

INTRODUCTION

Flooding is a recurring and devastating issue faced by many states in India, causing significant loss of life and property each year. With climate change exacerbating weather patterns and unpredictable monsoon seasons, early detection and warning systems have become critical in mitigating the impact of floods. To address this growing concern, our project focuses on the development of an advanced flood detection and alert system that leverages sensor technology for real-time monitoring of key environmental factors that contribute to flooding.

This system monitors natural indicators such as humidity, temperature, water flow, and water level in riverbeds. By using sensors like DHT11, Flow Sensor, and HC-SR04 (Ultrasonic Sensor), the system continuously gathers data on these factors, enabling timely detection of potential flood threats.

The **DHT11 Temperature-Humidity Sensor** is a widely used module that monitors both temperature and humidity. It plays an important role in detecting atmospheric changes that may signal impending heavy rainfall, which can trigger floods. The sensor uses resistive components to measure humidity levels and a thermistor to measure temperature, providing crucial environmental data.

The **Flow Sensor** is responsible for monitoring the flow of water in rivers and streams. The sensor consists of a plastic valve body, a water rotor, and a Hall-Effect sensor. When water levels fluctuate, the flow sensor detects the changes by completing or breaking the circuit based on water flow rates. This helps the system understand how fast the water is rising, an important parameter in flood prediction.

The **HC-SR04 Ultrasonic Sensor** Measures the distance to an object (e.g., the water level) using ultrasonic waves. It sends a trigger signal and calculates the time it takes for the echo to return, allowing it to determine the distance (in cm).

In summary, this sensor-based flood detection system offers a comprehensive solution to address the challenges posed by flooding in India. By monitoring critical environmental parameters and providing early warnings, it aims to safeguard communities, reduce damage, and enhance preparedness in flood-prone areas.

CHAPTER 2
LITERATURE SURVEY

Journal	Method Used	Inference
Hadi, M. I., F. Yakub, A. Fakhrurradzi, C. X. Hui, A. Najiha, N. A. Fakharulrazi, A. N. Harun, Z. A. Rahim, and A. Azizan. "Designing early warning flood detection and monitoring system via IoT." In *IOP Conference Series: Earth and Environmental Science*, vol. 479, no. 1, p. 012016. IOP Publishing, 2020.	The proposed system uses an ultrasonic sensor to measure water levels, categorizing them into safe, warning, and critical stages. Integrated with IoT, it sends real-time notifications and lets users monitor levels remotely. A solenoid valve and water pump help redirect excess water, effectively reducing flood risks.	This IoT-based early flood detection system monitors water levels with ultrasonic sensors, sends alerts based on safe, warning, and critical thresholds, and includes mechanisms to prevent flooding using solenoid valves and pumps.
Iyekekpolo, Uyioghosa B., Francis E. Idachaba, and Segun I. Popoola. "Early flood detection and monitoring system based on Wireless Sensor Network." In *Proc. Int. Conf. Ind. Eng. Oper. Manag*, vol. 2018, pp. 1381-1394. 2018.	The method used in this flood detection system involves deploying sensor nodes as part of a Wireless Sensor Network (WSN) at locations vulnerable to flooding. These nodes continuously monitor flood indicators, such as water levels and flow, in real time. Simulations were conducted for various flood scenarios, demonstrating the system's efficiency and accuracy in early flood detection.	This paper discusses a Wireless Sensor Network (WSN) system for real-time flood monitoring, aimed at providing early warnings to flood-prone areas, reducing the impact of floods and enhancing preparedness.
Siddique, Mohammed, Tasneem Ahmed, and Mohammad Shahid Husain. "Flood Monitoring and Early Warning Systems–An IoT Based Perspective." *EAI Endorsed Transactions on Internet of Things* 9, no. 2 (2023).	The method used in this study involves integrating Internet of Things (IoT) technology with Synthetic Aperture Radar (SAR) data for efficient flood monitoring and early warning systems. IoT-based sensor networks collect real-time data on environmental conditions, which is then processed using cloud computing and machine learning algorithms to predict floods and issue alerts.	Flood monitoring systems refers to the disruption or distortion of signal transmission, often caused by environmental factors or other electronic devices. It can negatively impact the accuracy of data collected by sensors, leading to unreliable readings. Effective mitigation strategies are necessary to ensure data integrity in real-time monitoring systems.

CHAPTER 3

SYSTEM DESCRIPTION

HARDWARE SPECIFICATIONS

(a) ESP8266 NodeMCU

Figure 1.

The ESP8266 NodeMCU is a WiFi development board that helps connect devices to the internet for IoT projects. It has an ESP8266 microcontroller with a 32-bit CPU, running at 80 MHz (can go up to 160 MHz) for quick data processing. It comes with 4 MB of flash memory for storing programs and data. The board operates at 3.3V, with pins for power and ground, digital I/O (GPIO), and analog input (ADC). It also has built-in WiFi (802.11 b/g/n), making it easy to connect to networks. Lightweight and small, it's ideal for low-cost, energy-efficient IoT applications.

(b) Ultrasonic sensor

Figure 2.

The ultrasonic sensor, commonly represented by models like the HC-SR04, measures distance using sound waves. It has two main parts: a transmitter, which sends out ultrasonic pulses, and a receiver, which listens for the pulse to return. When the pulse hits an object, it bounces back, allowing the sensor to calculate the distance based on the time taken. The HC-SR04 can measure distances between 2 cm to 400 cm, with an accuracy of around 3 mm. It operates on 5V DC and requires low power, making it ideal for Arduino and microcontroller projects like obstacle detection and distance measurement.

(c) Water Flow Sensor

Figure 3.

A water flow sensor measures the flow rate of water in a pipe or system, using a rotor and a magnetic hall-effect sensor to detect the movement. When water passes through, it spins the rotor, creating pulses in the sensor's magnetic field. These pulses are then counted to calculate the flow rate. The sensor typically operates within a 5V to 24V range and can measure flow rates from around 1 liter per minute to over 30 liters per minute, depending on the model. It's made from durable materials like plastic or stainless steel to withstand water pressure and corrosion.

(d) Temperature and Humidity Sensor:

Figure 4.

The DHT11 Temperature and Humidity Sensor is a popular, low-cost sensor that measures air temperature and humidity. It has a single-wire digital output and operates with a voltage range of 3.3 to 5V, making it compatible with various microcontrollers, including Arduino and ESP8266. The sensor has a temperature range of 0–50°C with an accuracy of ±2°C and a humidity range of 20–90% with an accuracy of ±5%. Its compact size and ease of use make it ideal for basic weather or environmental monitoring applications, though it responds slowly and may not be suitable for high-precision needs.

(e) 12 V Adapter:

Figure 5.

A 12V adapter is an electronic device that converts AC (alternating current) from a wall outlet into 12V DC (direct current), which many electronic devices require to operate. This adapter provides a steady 12V output voltage and is commonly used for powering small electronics like routers, LED lights, and Arduino projects. Specifications often include a power rating (usually measured in amps or watts), which indicates how much current it can safely supply. For example, a 12V, 1A adapter delivers 12 watts of power. Make sure its plug type and polarity match your device to ensure compatibility and safe operation.

(f) 16x2 Lcd Display:

Figure 6.

A 16x2 LCD display is a simple, widely-used display module showing 16 characters per line across two lines, ideal for basic projects and devices. It typically operates on a 5V power supply, consuming low power. Each character is displayed within a 5x8 pixel matrix, allowing clear, alphanumeric text. It uses either 4-bit or 8-bit parallel communication to connect with microcontrollers, and it often has an HD44780 driver, which simplifies interfacing. Backlighting enhances readability, and contrast can be adjusted via a potentiometer. Compact and affordable, this display is popular for Arduino and microcontroller projects to show information like sensor readings.

(g) PCF8574A

Figure 7.

The PCF8574A is an I2C I/O expander that allows you to increase the number of digital input/output pins available on microcontrollers like Arduino. It has 8 pins, which can be configured as either inputs or outputs, making it useful for connecting buttons, LEDs, and other devices. The PCF8574A communicates with microcontrollers using the I2C protocol, which requires only two wires (SDA for data and SCL for clock). It operates at voltages from 2.5V to 6V and has a built-in pull-up resistor feature, making it versatile for various electronic projects.

SOFTWARE SPECIFICATIONS

(a) Arduino ide

The Arduino IDE (Integrated Development Environment) is a user-friendly software that allows you to write, edit, and upload code to Arduino boards. It's designed for beginners and experienced users alike, making it easy to create projects with Arduino hardwareWhen you open the IDE, you'll find a simple text editor where you can write your code, known as sketches. The IDE uses a programming language similar to C++, which is easy to learn. It also provides helpful features like syntax highlighting, which makes your code easier to read by coloring keywords and functions.Once you've written your code, you can check it for errors using the "Verify" button. If everything is correct, you can upload your code directly to the Arduino board using a USB cable. The IDE also includes a Serial Monitor, which lets you see messages from your board in real time, helping you debug and understand your project better.Overall, the Arduino IDE is a powerful tool that simplifies programming for Arduino projects, making it accessible for anyone interested in electronics and coding. Whether you want to build a simple blinking LED or a complex robot, the IDE has everything you need to get started.

Figure 8.

(b) Blynk

Figure 9.

Blynk is a user-friendly platform that helps you create Internet of Things (IoT) projects easily. It allows you to connect different devices, like sensors and microcontrollers, to the internet so you can monitor and control them remotely using your smartphone or computer.

The software consists of two main parts: the Blynk app and the Blynk server. The Blynk app is available for iOS and Android, and it lets you design your project's interface. You can add buttons, sliders, and other controls to create a custom dashboard. The Blynk server manages the data exchange between your devices and the app.

To use Blynk, you start by creating a project in the app. You then add widgets, which are small tools that let you interact with your device, like turning lights on or off or checking the temperature. After setting up your project, you write a few lines of code using Blynk's library to connect your hardware to the app.

With Blynk, even beginners can make smart home systems, weather stations, or any IoT project. It's a powerful tool that makes it easy to bring your ideas to life without needing advanced programming skills.

CHAPTER 4

METHODOLOGY

1. Initialization and Setup:

- Essential libraries are included for Blynk, Wi-Fi connection, and LCD.

- The Wi-Fi credentials and Blynk authentication token are defined to connect the ESP8266 to the cloud service.

- The DHT11 temperature and humidity sensor, ultrasonic sensor, and flow sensor are initialized with their respective pins.

- An interrupt is attached to the flow sensor to count pulses for flow rate calculation.

2. Wi-Fi and Blynk Connection:

- The ESP8266 attempts to connect to Wi-Fi using the provided credentials.

- Once connected, it uses the Blynk library to connect to the Blynk cloud service.

3. Reading Temperature and Humidity (DHT11 Sensor):

- In the main loop, the DHT11 sensor reads the temperature and humidity.

- These values are saved in temp and hum variables for display and transmission to Blynk.

4. Water Level Measurement (Ultrasonic Sensor):

- The ultrasonic sensor uses trigger and echo pins to measure the time taken for sound waves to bounce back after hitting the water surface.

- The time is then converted to distance (in cm) to determine the water level. This value is saved in the variable cm.

5. Water Flow Rate Measurement (Flow Sensor):

- The flow sensor generates pulses as water flows through it. The pulse count is measured by an interrupt-based function, pulseCounter.

- Every second, the loop calculates the flow rate in liters per minute using the pulse1Sec count and a calibration factor.

- The flow rate is converted to milliliters and added to the cumulative total volume (in liters and milliliters).

6. Data Display (LCD):

- The I2C LCD displays the current temperature, humidity, flow rate, and water level on two rows.

- The display is updated every second with the latest data.

7. Data Transmission (Blynk):

- Every few iterations, the code sends temperature, humidity, flow rate, and water level to the Blynk app via Blynk.virtualWrite.

- This allows real-time monitoring of the data on a mobile device.

Working Process Overview

- The system collects environmental data (temperature, humidity, flow rate, and water level).

- The ESP8266 module sends the data to the Blynk cloud over Wi-Fi, allowing remote monitoring on the Blynk mobile app.

- The LCD screen continuously displays the data locally for on-site observation.

- Flow rate and total volume calculations are updated regularly, allowing real-time tracking of water movement and flood levels.

This setup creates an efficient flood monitoring and alert system, where environmental and water-related data can be tracked remotely, giving early insights into potential flood conditions.

CIRCUIT DIAGRAM :

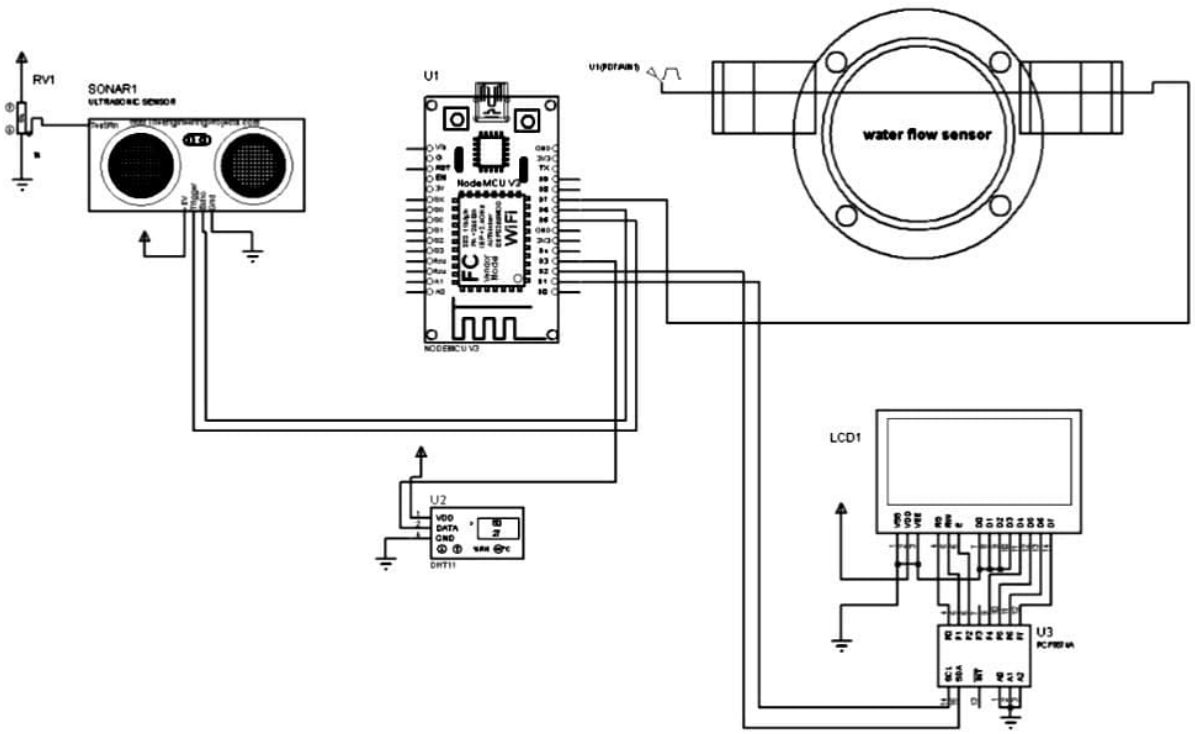

Figure 11.

HARDWARE MODEL DEVELOPMENT:

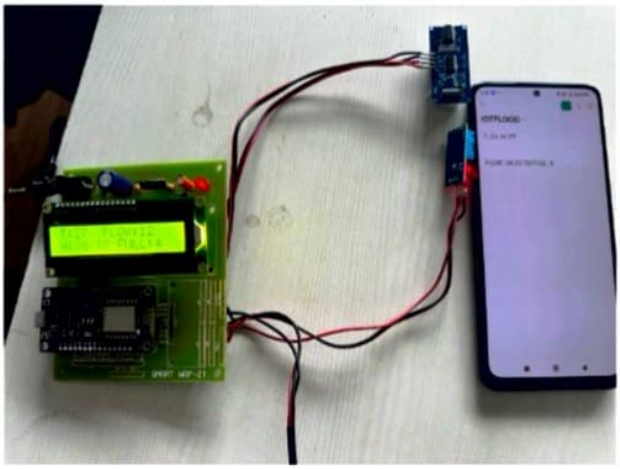

Figure12.

Our IoT-based early flood detection system leverages the ESP8266 NodeMCU along with ultrasonic sensors, a water level sensor, and a temperature and humidity sensor to monitor flood risk factors. Real-time data on water levels, flow, temperature, and humidity is displayed on a 16x2 LCD screen and integrated with the Blynk software, enabling remote monitoring. When water levels exceed a critical threshold, the system sends alerts to authorities and the public, facilitating timely evacuations and risk mitigation. This project is aimed at reducing flood impacts in vulnerable regions.

Output of Hardware Model:

Figure 13.

The flood detection system uses a 16x2 LCD to display real-time data on temperature, humidity, water flow, and water level. This setup provides immediate, accessible monitoring for on-site flood risk assessment.

Software Model Development:

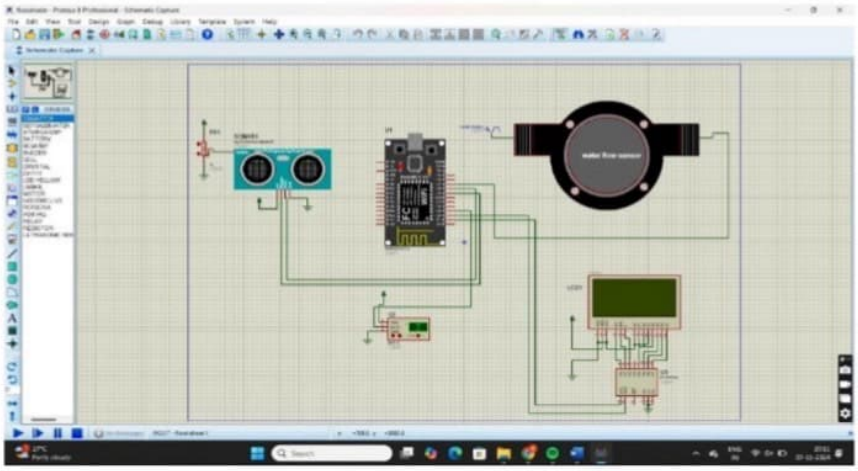

Figure 14.

Our IoT early flood detection simulation, developed using Proteus, incorporates an ESP8266 module along with an ultrasonic sensor, water flow sensor, and temperature and humidity sensor to monitor environmental conditions that signal potential flooding. The simulation allows for real-time analysis of water levels, flow rate, temperature, and humidity, visualized within the Proteus environment. This model enables us to test and validate system responses to simulated flood scenarios before deploying the physical hardware, ensuring reliable performance. Designed for regions at high flood risk, the model aims to enhance early warning and improve response times.

Output :

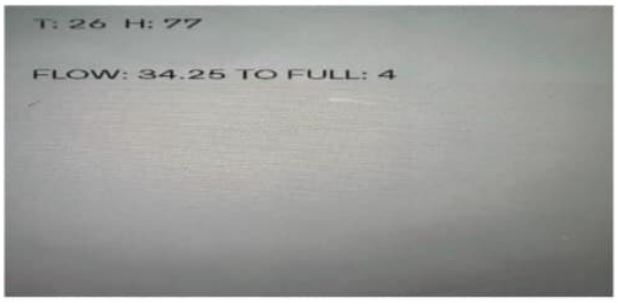

Figure 15.

The IoT early flood detection system updates data every 20 seconds on Blynk, using the ESP8266 to track real-time changes. It monitors water levels, flow rate, temperature, and humidity with ultrasonic and flow sensors, providing timely information. This ensures remote monitoring and quick response to flood risks.

CODE:

```
#define BLYNK_TEMPLATE_ID "TMPL3J-FcKrPY"

#define BLYNK_TEMPLATE_NAME "IOTFLOOD"

#define BLYNK_AUTH_TOKEN "RUJQp6ynPxPSwqasikbSM_8rcykDhdD0"


#include <Wire.h>

#include <LCD_I2C.h>

LCD_I2C lcd(0x27);


#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>



const char* ssid     = "IOT1";

const char* password = "123456789";



char auth[] = BLYNK_AUTH_TOKEN;


#include <DHT.h>


#define DHTPIN D3

#define DHTTYPE    DHT11


DHT dht(DHTPIN, DHTTYPE);
```

```
int temp,hum = 0;
#define SENSOR  D7
 int cnt=0;
long currentMillis = 0;
long previousMillis = 0;
int interval = 1000;
boolean ledState = LOW;
float calibrationFactor = 4.5;
volatile byte pulseCount;
byte pulse1Sec = 0;
float flowRate;
unsigned long flowMilliLitres;
unsigned int totalMilliLitres;
float flowLitres;
float totalLitres;
int trigPin = D5;    // Trigger
int echoPin = D6;    // Echo
long duration, cm,inches;

void IRAM_ATTR pulseCounter()
{
  pulseCount++;
}

WiFiClient client;
```

```
void setup()
{
 Serial.begin(9600);
 lcd.begin();              //Init the LCD
 lcd.backlight();          //Activate backlight
 lcd.home();
 lcd.setCursor(0,0);lcd.print("CONNECTING...");
 Blynk.begin(auth, ssid, password, "blynk.cloud", 80);
 dht.begin();
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);
 pinMode(SENSOR, INPUT_PULLUP);

 pulseCount = 0;
 flowRate = 0.0;
 flowMilliLitres = 0;
 totalMilliLitres = 0;
 previousMillis = 0;

 attachInterrupt(digitalPinToInterrupt(SENSOR), pulseCounter, FALLING);
}

void loop()
{
 temp=dht.readTemperature();
 hum=dht.readHumidity();
```

```
digitalWrite(trigPin, LOW);
delayMicroseconds(5);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
pinMode(echoPin, INPUT);
duration = pulseIn(echoPin, HIGH);

// Convert the time into a distance
cm = (duration/2) / 29.1;     // Divide by 29.1 or multiply by 0.0343
//inches = (duration/2) / 74;   // Divide by 74 or multiply by 0.0135

currentMillis = millis();
if (currentMillis - previousMillis > interval)
{

 pulse1Sec = pulseCount;
 pulseCount = 0;

 // Because this loop may not complete in exactly 1 second intervals we calculate
 // the number of milliseconds that have passed since the last execution and use
 // that to scale the output. We also apply the calibrationFactor to scale the output
 // based on the number of pulses per second per units of measure (litres/minute in
 // this case) coming from the sensor.
 flowRate = ((1000.0 / (millis() - previousMillis)) * pulse1Sec) / calibrationFactor;
 previousMillis = millis();
```

```
  // Divide the flow rate in litres/minute by 60 to determine how many litres have
  // passed through the sensor in this 1 second interval, then multiply by 1000 to
  // convert to millilitres.
  flowMilliLitres = (flowRate / 60) * 1000;
  flowLitres = (flowRate / 60);

  // Add the millilitres passed in this second to the cumulative total
  totalMilliLitres += flowMilliLitres;
  totalLitres += flowLitres;

  // Print the flow rate for this second in litres / minute
  Serial.print("Flow rate: ");
  Serial.print(float(flowRate));  // Print the integer part of the variable
  Serial.println("L/min");



}
cnt=cnt+1;
if(cnt>20){
Blynk.virtualWrite(V0,"T: "+String(temp)+"  H: "+String(hum));
Blynk.virtualWrite(V1,"FLOW: "+String(flowRate)+" TO FULL: "+String(cm));
cnt=0;
}
```

```
 lcd.clear();
 lcd.setCursor(0,0);lcd.print("T:");
 lcd.print(temp);
 lcd.setCursor(0,1);lcd.print("H:");
 lcd.print(hum);

 lcd.setCursor(6,0);lcd.print("FLOW:");
 lcd.print(int(flowRate));
 lcd.setCursor(5,1);lcd.print("TO FULL:");
 lcd.print(cm);
 delay(1000);



}
```

CHAPTER 5

CONCLUSION

The IoT-based flood detection system built with an ESP8266 NodeMCU and sensors such as ultrasonic, water flow, and temperature sensors provides a reliable, real-time monitoring solution for flood-prone areas. This system can detect rising water levels, monitor water flow rates, and track temperature changes, all of which are key indicators of potential flooding events.

By utilizing the ESP8266's Wi-Fi capability, the system connects to a cloud platform (such as Blynk), allowing data to be transmitted to remote servers. Users can monitor water levels, flow rates, and environmental conditions through a mobile app, receiving alerts if thresholds are exceeded. This real-time data transmission and alerting allow timely responses, potentially reducing damage to property and enhancing public safety.

In summary:

- Efficiency: The system provides continuous and accurate data on water levels, flow rates, and temperatures.
- Accessibility: Remote monitoring through Wi-Fi connectivity ensures users can access data and receive alerts from anywhere.
- Scalability: The system can be adapted and scaled for different locations or more extensive monitoring needs.

This project demonstrates the potential of IoT technology in disaster prevention and early warning systems, offering a cost-effective and scalable solution to mitigate flood risks..

REFERENCES

- Hadi, M. I., Yakub, F., Fakhrurradzi, A., Hui, C. X., Najiha, A., Fakharulrazi, N. A., ... & Azizan, A. (2020, June). Designing early warning flood detection and monitoring system via IoT. In *IOP Conference Series: Earth and Environmental Science* (Vol. 479, No. 1, p. 012016). IOP Publishing.

- Iyekekpolo, U. B., Idachaba, F. E., & Popoola, S. I. (2018). Early flood detection and monitoring system based on Wireless Sensor Network. In *Proc. Int. Conf. Ind. Eng. Oper. Manag* (Vol. 2018, pp. 1381-1394).

- Siddique M, Ahmed T, Husain MS. Flood Monitoring and Early Warning Systems–An IoT Based Perspective. EAI Endorsed Transactions on Internet of Things. 2023 Jul 24;9(2).