SentryLeaf: Rescue Network and IoT Devices for Flood Affected Area

by

Md. Ragib Rawnak Rohan
ID: CSE2101022071

Md. Rakibur Rahman
ID: CSE2101022070

Md. Sazzad Hossain Jibon
ID: CSE2002020025

Naimul Islam Omit
ID: CSE2101022116

Supervised by
Md. Ashfakur Rahman

Submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SONARGAON UNIVERSITY (SU)

January 2025

APPROVAL

The project titled "**SentryLeaf: Rescue Network And IoT Devices For Flood Affected Area.**" submitted by Md. Ragib Rawnak Rohan (CSE2101022071), Md. Rakibur Rahman (CSE2101022070), Md. Sazzad Hossain Jibon (CSE2002020025) and Naimul Islam Omit (CSE2101022116) to the Department of Computer Science and Engineering, Sonargaon University (SU), has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering and approved as to its style and contents.

Board of Examiners

Md. Ashfakur Rahman **Supervisor**
Lecturer
Department of Computer Science and Engineering
Sonargaon University (SU)

(Examiner Name and Signature) **Examiner 1**
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 2
(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

(Examiner Name and Signature) **Examiner 3**
Department of Computer Science and Engineering
Sonargaon University (SU)

DECLARATION

We, hereby, declare that the work presented in this report is the outcome of the investigation performed by us under the supervision of **Md. Ashfakur Rahman**, Lecturer**,** Department of Computer Science and Engineering, Sonargaon University, Dhaka, Bangladesh. We reaffirm that no part of this project has been or is being submitted elsewhere for the award of any degree or diploma.

Countersigned

(Md. Ashfakur Rahman)
Supervisor

Signature

Md. Ragib Rawnak Rohan
ID: CSE2101022071

Md. Rakibur Rahman
ID: CSE2101022070

Md. Sazzad Hossain Jibon
ID: CSE2002020025

Naimul Islam Omit
ID: CSE2101022116

ABSTRACT

Floods are among the most devastating natural disasters, causing widespread destruction to property, infrastructure, and human lives. Rapid response and efficient resource allocation are critical for mitigating their impact. **SentryLeaf** proposes an innovative rescue network that leverages Internet of Things (IoT) technology to enhance disaster management in flood-affected areas.

The system integrates a network of IoT-enabled devices, including sensors, and communication modules, to monitor flood conditions in real time. These devices collect data on water levels, flow rates, and environmental parameters, which are transmitted to a centralized control system via low-power, high-range communication protocols. This information is used to generate accurate flood models, predict the spread of water, and identify high-risk zones.

SentryLeaf also facilitates coordination between rescue teams through a mesh network of connected devices, ensuring seamless communication even in areas where traditional networks are disrupted. Additionally, IoT-enabled rescue nodes provide real-time updates, enabling targeted aid delivery, efficient evacuation planning, and resource optimization. The system is designed to be scalable, cost-effective, and deployable in both urban and remote regions.

The adoption of SentryLeaf's IoT-based approach has the potential to revolutionize flood response efforts, saving lives and minimizing damage by providing actionable insights and ensuring timely intervention.

ACKNOWLEDGMENT

At the very beginning, we would like to express my deepest gratitude to the Almighty Allah for giving us the ability and the strength to finish the task successfully within the schedule time.

We are auspicious that we had the kind association as well as supervision of **Md. Ashfakur Rahman**, Lecturer, Department of Computer Science and Engineering, Sonargaon University whose hearted and valuable support with best concern and direction acted as necessary recourse to carry out our project.

We would like to convey our special gratitude to **Prof. Bulbul Ahamed,** Head of the Department, Computer Science and Engineering and **Brig. Gen. (Retd) Prof. Habibur Rahman Kamal, NDC, PSC,** Dean, Faculty of Science and Engineering, Sonargaon University, for their kind concern and precious suggestions.

We are also thankful to all our teachers during our whole education, for exposing us to the beauty of learning.

Finally, our deepest gratitude and love to my parents for their support, encouragement, and endless love.

LIST OF ABBREVIATIONS

ACL	Access Control Lists
AP	Access Point
CAN	Campus Area Network
DHCP	Dynamic Host Configuration Protocol
ESP	Espressif Systems (ESP8266MOD)
HTTP	Hyper Text Transfer Protocol
IoT	Internet of Things
IP	Internet Protocol
ISP	Internet Service Provider
LAN	Local Area Network
MAC	Media Access Controller
MAN	Metropolitan Area Network
NAT	Network Address Translation
P2P	Peer-to-Peer Communication
PAN	Personal Area Network
PHP	Hypertext Preprocessor
QoS	Quality of Service
SSID	Service Set Identifier
VLAN	Virtual Local Area Network
WAN	Wide Area Network
Wi-Fi	Wireless Fidelity PHP
WLAN	Wireless Local Area Network
WMAN	Wireless Metropolitan Area Network
WPAN	Wireless Personal Area Network
WWAN	Wireless Wide Area Network
XAMPP	Cross-Platform, Apache, MySQL, PHP, and Perl

TABLE OF CONTENTS

Title	Page No
DECLARATION ……	i
ABSTRACT ……	ii
ACKNOWLEDGEMENT ……	iii
LIST OF ABBREVIATION ……	vi
Chapter 1: Introduction	1-2
1.1 Overview of the project…….	1
1.2 Problem statement……	1
1.3 Objectives and scope of the project……...	1
1.4 Significance of the study……..	2
1.5 Research methodology……..	2
Chapter 2: Literature Review	3-5
2.1 Existing flood monitoring and rescue systems……	3
2.2 IoT applications in disaster management……	3
2.3 Wireless communication technologies for disaster recovery……...	3-4
2.4 Challenges in flood response and communication in affected areas..….	4
2.5 Previous research and related works……	4-5
Chapter 3: Architecture of SentryLeaf	6-10
3.1 Overview of the system design……	6
3.2 Components and their functions……	7-8
3.3 Data flow and communication between components……..	8-9
3.4 Network topology and IoT device integration…….	9
3.5 User interface design and accessibility……	9-10
Chapter 4: System Implementation	11-14
4.1 Hardware setup and configuration……	11-12
4.2 Software design and development……...	12-13
4.3 Testing and troubleshooting……..	13-14
4.4 System integration……	14
Chapter 5: Results and Discussion	15-18
5.1 Performance of the network and IoT devices……	15-16
5.2 User experience and feedback……...	16-17
5.3 Comparison with traditional flood monitoring systems……...	17-18
5.4 Limitations and challenges encountered……...	18
Chapter 6: Conclusion and Future Work	19-21
6.1 Summary of findings…….	19
6.2 Contributions of the project…….	19-20
6.3 Potential for future improvements and scalability……...	20
6.4 Recommendations for further research and development……	20-21
REFERENCES …….....	22
APPENDIX …….	23-38

LIST OF TABLES

Figure No.	Title	Page No.
Table 3.1	Hardware Components Table ……..	08

LIST OF FIGURES

Figure No.	Title	Page No.
Figure 3.1	Architecture diagram of SentryLeaf …….	06
Figure 3.2	Use Case Diagram…….	07
Figure 3.3	WEB Interface for Rescue Team……	09
Figure 3.5	WEB Login Page for Rescue Team……	10
Figure 4.1	IoT Device 1(Water Level Monitoring)……	11
Figure 4.2	IoT Device 2(Rescue Request Sender) …….	12
Figure 4.3	Data flow of SentryLeaf…….	14
Figure 5.1	Water-Level Sensors and ESP8266MOD Connections..	16
Figure 5.2	Voice sender.....	16
Figure 5.3	Real-Time Water level monitoring Data…….	18

CHAPTER 1

INTRODUCTION

1.1 Overview of the project

Floods are among the most devastating natural disasters, causing widespread damage to infrastructure, property, and most importantly, human lives. In response to this, the **SentryLeaf:** Rescue Network and IoT Devices for Flood-Affected Areas project aims to provide a reliable and scalable solution for flood monitoring and communication in flood-prone regions. The system integrates IoT devices with a decentralized communication network to deliver real-time data on water levels, environmental conditions, and other critical information. The system leverages **Mikrotik RB750r2 hEX** lite routers, **Range Extender APs**, and **XAMPP servers** to form a robust infrastructure that ensures connectivity and information accessibility even when traditional communication channels are down. Through this system, emergency responders, local authorities, and affected individuals will be able to access real-time information, enabling faster and more informed decision-making during floods.

1.2 Problem Statement

In many flood-prone areas, the lack of reliable communication and flood monitoring systems exacerbates the impact of disasters. Conventional infrastructure often becomes inoperable during floods, leaving communities isolated and without timely warnings or rescue updates. There is a critical need for a decentralized and self-sustaining communication system that can deliver real-time data on water levels, weather conditions, and emergency alerts. The **SentryLeaf** project addresses this gap by combining **IoT technology** with **peer-to-peer communication** networks, enabling effective monitoring, coordination, and support in flood-affected regions.

1.3 Objectives and Scope of the Project

The primary objectives of the **SentryLeaf** project are:

i. To design and implement a **rescue network** that ensures seamless communication in flood-affected areas.
ii. To deploy **IoT devices** that can monitor environmental conditions such as temperature, humidity, and water levels, sending real-time data to a central server for processing and display.
iii. To create a **web-based platform** where users, including residents and responders, can access real-time flood data and emergency alerts.
iv. To ensure scalability, enabling the system to expand and cover larger areas as needed.

The scope of the project includes the design, implementation, and testing of the **SentryLeaf** system in a controlled environment, followed by real-world testing in a flood-prone area. The system will be evaluated based on performance, reliability, ease of use, and effectiveness in emergency response scenarios.

1.4 Significance of the Study

The **SentryLeaf** project holds significant value for disaster management, particularly in flood-prone areas where communication infrastructure is often disrupted. The project aims to bridge the gap in emergency communication, providing a system that operates independently of conventional internet and telecom networks. By offering **real-time flood monitoring** and reliable communication, the system can help save lives, reduce property damage, and support timely rescue and relief operations. Additionally, the system's **scalability** and **user-friendliness** make it adaptable for use in other disaster scenarios, such as hurricanes or wildfires, providing a foundation for broader applications in emergency management.

1.5 Research Methodology

The research methodology for this project follows a **design-based approach**, combining system development, testing, and evaluation. The project begins with a **literature review** to understand existing technologies and solutions for flood monitoring and communication. The system is then designed using **IoT-based** sensors for environmental monitoring, a **Mikrotik router** for communication, and a **XAMPP server** for data processing. The implementation phase involves setting up the hardware, configuring the network, and developing the **web-based platform**. Afterward, the system is tested in controlled environments to assess its performance, data accuracy, and overall functionality. Finally, the system is evaluated through user feedback, performance analysis, and comparison with traditional flood monitoring methods.

CHAPTER 2

LITERATURE REVIEW

2.1 Existing Flood Monitoring and Rescue Systems

Flood monitoring and rescue systems have become critical in reducing the damage caused by floods. Traditional flood monitoring systems often rely on weather stations, satellite data, and ground-based sensors. These systems, however, face challenges such as high costs, lack of coverage in remote areas, and delayed data reporting. Many countries have developed national flood early warning systems that use river gauges, weather data, and forecasting models. For example, the **Global Flood Monitoring System (GFMS)** provides real-time flood monitoring globally, and local initiatives like the **Flood Warning Information System (FWIS)** in the United States utilize river gauges and data analytics to predict and monitor flooding. Rescue systems typically involve emergency response teams, vehicles, and helplines. However, these systems are often limited by poor communication networks, especially in flood-affected regions where infrastructure is damaged or non-existent.

2.2 IoT Applications in Disaster Management

The Internet of Things (IoT) has shown immense potential in disaster management, particularly in flood monitoring and response. IoT devices, such as sensors and connected devices, can continuously monitor environmental parameters like water levels, humidity, and rainfall, providing real-time data to authorities and the public. In the context of flood monitoring, IoT-based systems, such as the **FloodNet project** and **SmartFlood**, use IoT-enabled sensors to gather data on water levels, temperature, and other critical factors. These systems transmit data to a centralized platform where it is processed and analyzed to issue early warnings or inform evacuation plans. IoT applications in disaster management also include mobile apps and automated systems that enable first responders to track real-time conditions, locate people in distress, and coordinate rescue operations more effectively.

Additionally, IoT-enabled flood detection devices can be placed in rivers, dams, or urban areas to measure water levels, predict flooding, and trigger warnings in case of emergencies. These devices are particularly useful in areas with limited infrastructure or where communication networks may be compromised due to the disaster. IoT offers scalability, flexibility, and a cost-effective means of managing flood monitoring in remote or affected regions.

2.3 Wireless Communication Technologies for Disaster Recovery

In disaster recovery situations, reliable communication is essential. Wireless communication technologies, such as **Wi-Fi**, **LoRaWAN**, **5G**, and **satellite communication**, are commonly used in disaster recovery for enabling network connectivity in affected areas. **Wi-Fi** and **mesh networking** provide short-range communication, allowing devices to communicate even if the main infrastructure is down. **LoRaWAN** (Long Range Wide Area Network) is a low-power, long-range wireless protocol that is particularly useful in rural and remote areas where cellular networks may not be available. **5G** networks offer high-speed data transfer, low latency, and can be utilized for real-time communication during emergencies, enabling better coordination for disaster management.

Satellite communication is a reliable option when local networks are damaged.[2] Technologies like **Iridium** and **Global star** provide satellite phones and communication devices that can operate in remote and disaster-stricken areas. These systems allow emergency responders to maintain communication and relay critical data to disaster management authorities.

The combination of these technologies ensures that communication remains operational even during and after floods, helping to coordinate rescue operations, issue public alerts, and provide updates on the situation.

2.4 Challenges in Flood Response and Communication in Affected Areas

Despite advances in flood monitoring and rescue systems, several challenges still hinder effective flood response. One of the primary challenges is the **disruption of infrastructure**, such as electricity, telecommunication lines, and roads, which makes it difficult for both responders and the affected population to access critical resources and communicate. Communication breakdowns are common in flooded areas due to the destruction of communication towers and power outages, which can delay emergency responses and complicate rescue operations.

Another challenge is the **lack of real-time data**, which makes it difficult to assess the full scale of the disaster and allocate resources efficiently. Without real-time monitoring, decisions may be based on outdated information, leading to delays or incorrect actions. **Coordination among multiple agencies** is also a challenge, as different organizations may have different communication systems, priorities, and information sources, making it harder to share critical data in a timely and efficient manner.

Finally, there is the issue of **lack of public awareness and access to information** during floods. Even if a flood monitoring system is in place, affected communities may not have access to the information in a usable format, or they may not know how to respond effectively. The **user-friendliness** of flood monitoring systems is crucial for their effectiveness, especially in disaster-prone regions where many people may have limited technical knowledge.

2.5 Previous Research and Related Works

Numerous studies and projects have focused on flood monitoring and disaster management using IoT technologies. For instance, **Chien et al. (2017)** developed a **flood monitoring system** using wireless sensor networks (WSN) to measure water levels and send real-time data to a remote monitoring station. Similarly, **Zhu et al. (2020)** proposed a **smart flood monitoring system** that integrates IoT sensors and cloud computing to predict and manage flood risks. These systems have shown the potential of IoT in improving the accuracy and timeliness of flood predictions, but many of them are limited by the infrastructure required for data transmission and power supply.

Other works, such as **Smart Cities for Disaster Management** by **Smith et al. (2019)**, focus on how IoT, combined with smart cities technologies, can improve emergency response by integrating various systems into a centralized platform. While these studies have demonstrated the efficacy of IoT in disaster scenarios, they often do not address the challenges of **rural or remote areas**, where communication and power sources may be scarce.

Some studies, like **The Development of a Smart Flooding System** by **Xu et al. (2018)**, emphasize the importance of low-cost IoT devices and battery-powered solutions to create scalable and sustainable flood monitoring systems. These findings are directly applicable to the **SentryLeaf** project, where cost-effective, low-power IoT devices play a central role in ensuring the system's scalability and resilience in flood-prone areas.

CHAPTER 3

ARCHITECTURE OF SENTRYLEAF

3.1 Overview of the System Design

The **SentryLeaf** project is designed to provide a decentralized rescue network for flood-affected areas by leveraging **IoT devices** and **wireless communication technologies**. The system architecture consists of several components working in unison to enable real-time flood monitoring, communication, and emergency response.[8] The network is designed to operate in areas where traditional infrastructure, such as cellular networks or power grids, may be compromised by flood conditions. By utilizing **Mikrotik RB750r2 hEX** lite routers, **Range Extender APs**, **XAMPP servers**, and **IoT sensors**, **SentryLeaf** provides a scalable, robust, and efficient solution for flood management.

The system is intended to be deployed in areas where flood monitoring is most critical, using **peer-to-peer** (P2P) communication between devices, enabling connectivity without the need for external network infrastructure. A web-based user interface allows emergency responders, local authorities, and affected residents to access real-time data about water levels and other environmental parameters.

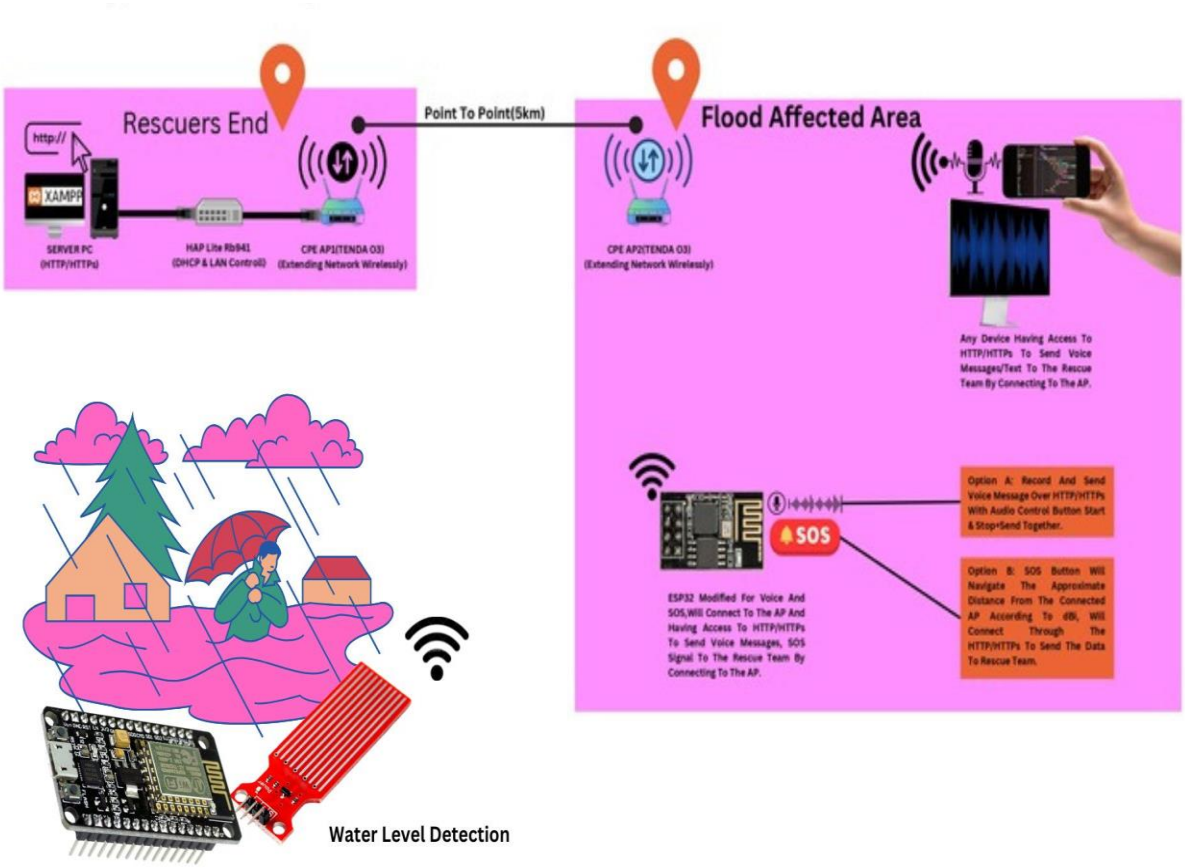


Figure 3.1 : Architecture diagram of SentryLeaf

3.2 Components and Their Functions

1. **Mikrotik RB750r2 hEX** lite Router The **Mikrotik RB750r2 hEX** lite router acts as the central hub for the rescue network. This router is equipped with Ethernet ports to connect devices and **Wi-Fi capabilities** to manage communication between different network nodes. It establishes **peer-to-peer communication** between the central network and **Range Extender APs**, allowing the network to extend even in flooded or remote areas where traditional infrastructure may fail. Its role is crucial in ensuring **stable communication**, enabling devices and users to stay connected throughout the flood-affected region.
2. **Range Extender APs** The **Range Extender APs** (Access Points) are critical in ensuring that the network can extend over a wider area, particularly in environments where the main infrastructure is damaged. These devices communicate with the Mikrotik router to relay the network signal, making sure the flood-affected area remains connected. The **Range Extender APs** also serve to distribute internet access to **IoT devices** and the **XAMPP server**, ensuring all components are part of a seamless network, even in areas where signal coverage might otherwise be limited. This component significantly enhances the **resilience** of the system and ensures **communication continuity** during flood emergencies.
3. **XAMPP Server** The **XAMPP server**, hosted on a laptop or a small server unit, serves as the central data processing and management system. It hosts a **local web server** that provides a platform for collecting, storing, and processing data from the IoT devices. The XAMPP server connects to the router, receives data transmitted from the IoT sensors, and processes it in real-time. This data can then be displayed on a **web interface**, where users can access important information, such as **water level readings**, **environmental conditions**, and **flood alerts**. The server allows for remote monitoring and is essential for data-driven decision-making during flood management.

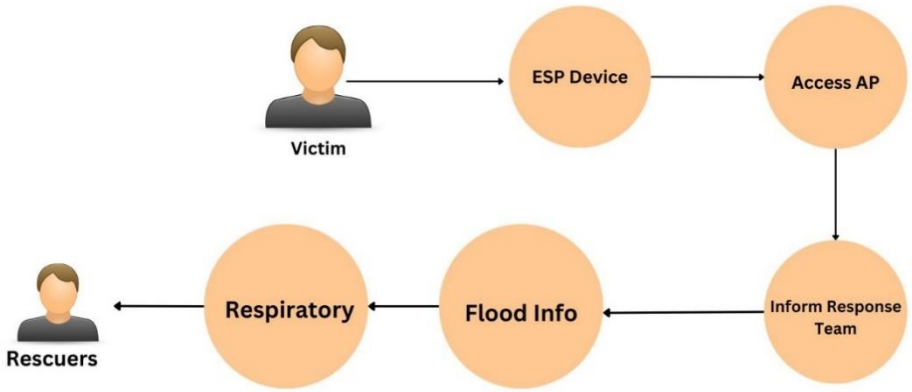


Figure 3.2: Use Case Diagram

4. **IoT Devices (ESP8266MOD, Water-Level Monitoring)** The **IoT devices** play a critical role in collecting real-time data from the environment. There are two primary IoT devices in the system:
    i. **Water-Level Monitoring Sensor**: This device is placed in key locations in the house floors, to monitor rising water levels. It is equipped with a hydrostatic pressure sensor that measures the water level and transmits this information to the XAMPP server. This sensor helps track the progress of floods in real-time and triggers **emergency alerts** if water levels exceed safe thresholds.

Component	Specification	Purpose	Remarks
Router	Mikrotik RB750r2 hEX lite	Acts as the main network hub for AP communication	Configured for DHCP communication
Range Extender AP	Point To Point 5KM AP Router	Extends the network range to reach IoT devices	Placed strategically for coverage
IoT Device 1 (Water Level Monitoring)	ESP8266MOD and Water Level Monitoring Sensor.	Measures environmental data like temperature and humidity	Sends data to the XAMPP server, Installed on house floors.
IoT Device 2 (Rescue Request Sender)	ESP8266MOD	Personal Device To Send Notification To Rescue Team	Installed on house floors to detect floods
Laptop/PC	Any Type, That CanWirelessly/LAN Connect To The Wifi.	Hosts the XAMPP server for data processing	Acts as the central data repository

Table 3.1: Hardware Components Table

3.3 Data Flow and Communication Between Components

The data flow within the **SentryLeaf** system follows a straightforward path:

1. **Data Collection**: The IoT devices continuously collect data, such as **water levels** and user input and transmit this information to the **Range Extender APs** via Wi-Fi.
2. **Data Transmission**: The **Range Extender APs** relay the data to the **Mikrotik router**, which ensures connectivity across the network and routes the data to the **XAMPP server** for processing.
3. **Data Processing**: The **XAMPP server** processes the received data, stores it in a local database, and prepares it for display on the **web-based user interface**. The server is also responsible for issuing **alerts** when water levels exceed predefined thresholds.
4. **User Access**: Users (emergency responders, local authorities, residents) can access the **web interface** hosted on the **XAMPP server** via any device with a browser. This interface provides real-time data visualization, allowing users to view **flood warnings**, **water levels**, and **environmental readings**.

The entire communication process is designed to be **low-latency** and **highly reliable**, ensuring that even in emergency situations, data can be transmitted and accessed in real-time. The decentralized architecture of the system further enhances its resilience, as it does not rely on a single point of failure and can continue operating even if parts of the network are compromised by the flood.

3.4 Network Topology and IoT Device Integration

The network topology of **SentryLeaf** is designed to be flexible, scalable, and robust. At the core of the system is the **Mikrotik RB750r2 hEXlite router**, which serves as the central communication point. The **Range Extender APs** are deployed throughout the flood-affected region, extending the network's coverage and ensuring communication in remote or compromised areas.[9]

The **IoT devices** are strategically placed to monitor critical environmental data. They are integrated into the network through the **Range Extender APs**, which communicate directly with the **Mikrotik router**. This network topology ensures that data is routed efficiently from the IoT sensors to the central server and ultimately to the user interface. The system's **modular design** allows for easy expansion, enabling the addition of more **IoT devices**or **Range Extender Aps** to cover a larger area as needed.

3.5 User Interface Design and Accessibility

This work will be scaled up to accommodate a web based Flood Information System (FIS). [1]
The **user interface** is a crucial element of the **SentryLeaf** system, providing users with a clear, intuitive way to interact with the data and manage flood-related information. Hosted on the **XAMPP server**, the interface is designed to be **accessible via any web browser**, enabling users to access the system from smartphones, tablets, or laptops.
The interface displays critical data, including:

i. Real-time water levels from the monitoring sensors.
ii. Flood alerts and emergency notifications.

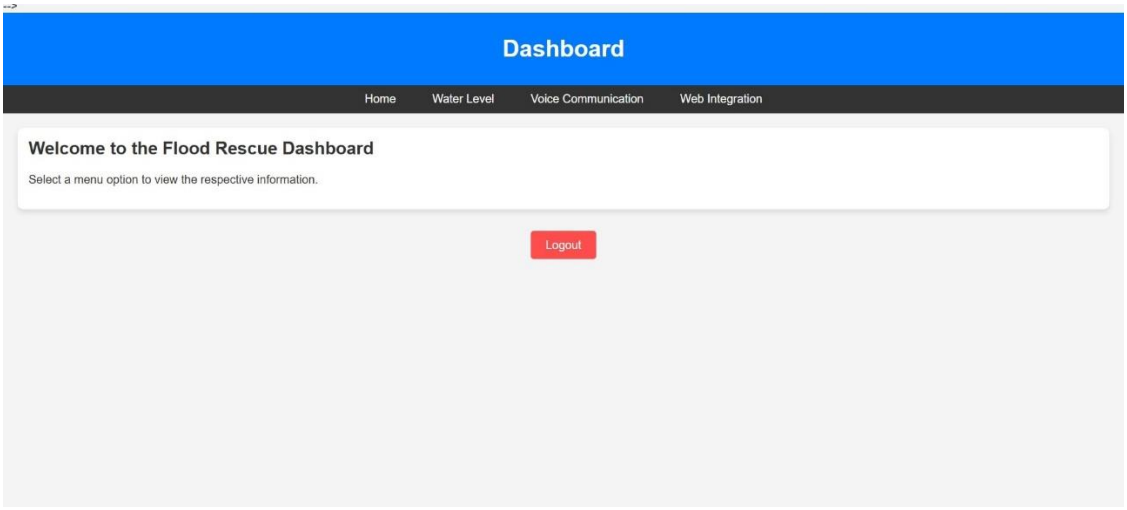


Figure 3.3 : WEB Interface for Rescue Team.

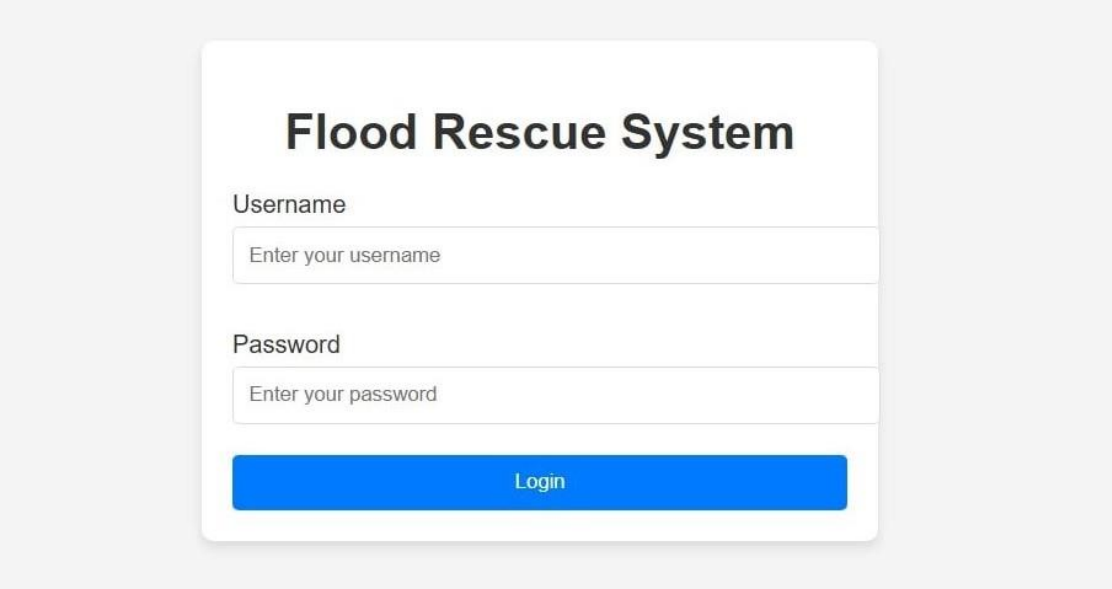


Figure 3.4 : WEB Login Page for Rescue Team.

The design prioritizes user-friendliness, ensuring that even individuals with minimal technical knowledge can navigate the platform efficiently. The interface also features **color-coded alerts** for immediate identification of dangerous conditions, and **graphs and charts** for easy understanding of water-level trends and environmental changes.[10]

CHAPTER 4
SYSTEM IMPLEMENTATION

4.1 Hardware Setup and Configuration

The successful implementation of the **SentryLeaf** system relies on the appropriate hardware setup and configuration. The key hardware components of the system include:

1. **Mikrotik RB750r2 hEX lite Router**: The **Mikrotik RB750r2** router serves as the central hub for communication in the **SentryLeaf** network. It is configured to handle **peer-to-peer (P2P) communication** between the **Range Extender APs** and ensure connectivity across the flood-affected region. The router is connected to the **XAMPP server** via Ethernet for stable and fast communication.
2. **Range Extender APs**: The **Range Extender Access Points** are strategically placed throughout the disaster zone to extend the network's coverage. These devices are configured to connect wirelessly with the Mikrotik router and ensure seamless communication between the central server, IoT devices, and end-users. Each **Range Extender APs** is configured to operate on a specific Wi-Fi channel to minimize interference.
3. **IoT Devices (ESP8266MOD, Water-Level Monitoring)**: The **ESP8266MOD** is used to collect environmental data and send it to the **XAMPP server** via the wireless network. The **water-level monitoring sensors** are connected to the IoT devices to measure the water levels in real time and transmit the data to the server. These sensors are configured to measure water levels using hydrostatic pressure or ultrasonic methods, and they send data wirelessly to the server via the **Range Extender APs**.
    i. **IoT Device 1 (Water Level Monitoring):** This device purpose is to send water level detection message to the rescue center.

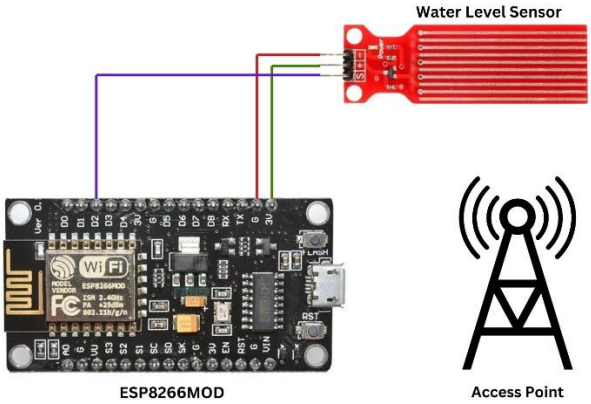


Figure 4.1 : IoT Device 1 (Water Level Monitoring)

ii. **IoT Device 2 (Rescue Request Sender):** This device triggers data transmission while the button is press and it is sent to the server using ap, tracking device id, mac address etc.

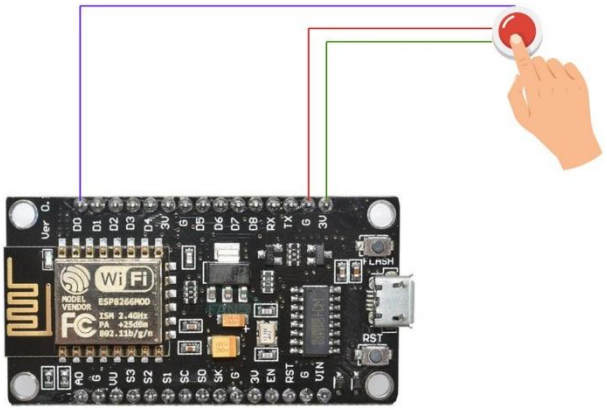


Figure 4.2: IoT Device 2 (Rescue Request Sender)

4. **XAMPP Server**: The **XAMPP server** is configured on a **laptop** or small **server** and acts as the main data repository and processing unit for the system. The server hosts the web interface, stores data, and processes requests from users and IoT devices. The server is connected to the Mikrotik router and is configured to communicate with the network components over **Wi-Fi**.

All devices are carefully configured to ensure that they can communicate with one another effectively, even in challenging environments such as flooded areas where traditional networks may not be available.

4.2 Software Design and Development

1. **Programming and Configuration of IoT Devices** The **IoT devices** are programmed using the **Arduino IDE** and the **ESP8266MOD** Wi-Fi module. The main tasks for the IoT devices include:
    i. **Sensor Data Collection**: The **ESP8266MOD** is connected to the **water-level sensor** to continuously monitor the water levels in real-time.
    ii. **Data Transmission**: The device uses the **Wi-Fi protocol** to connect to the **Range Extender APs**, which act as the communication bridge to the **XAMPP server**.
    iii. **Power Management**: The IoT devices are programmed to minimize power consumption by utilizing deep-sleep modes and sending data at regular intervals, ensuring the system can function for extended periods on battery power.

    The sensor readings are sent in **JSON format** via **HTTP requests** to the XAMPP server, allowing the server to process and store the data in a structured format.

2. **Server-Side Setup (XAMPP and Web Interface)** The **XAMPP server** is installed on a **laptop** or small computer running **Windows** or **Linux** and includes the following components:
    i. **Apache Web Server**: Hosts the web interface for users to view real-time data.
    ii. **MySQL Database**: Stores sensor data, including **water levels**, **environmental conditions**, and **alert statuses**. This database is configured to receive data from the IoT devices via **HTTP POST requests**.
    iii. **PHP Scripts**: Handles the server-side logic, processing incoming data from IoT devices, managing user requests, and generating alerts for abnormal water levels or flood conditions.

    The **web interface** is designed to be accessible through any browser and provides real-time data visualization for users. The interface includes:

    i. **Graphs and charts** to display water levels over time.

    ii. **Emergency alerts** in case of abnormal water levels.

    iii. **Interactive maps** showing the status of different monitoring locations.

    The server-side setup ensures that data is processed in real-time and made available to users with minimal latency.

    3. **Communication Protocols Used (P2P, Wi-Fi)** The communication in the **SentryLeaf** system relies on **peer-to-peer** (P2P) and **Wi-Fi** protocols:

    ii. **P2P Communication**: The system uses **P2P communication** to ensure devices can directly communicate with each other without needing a centralized internet connection. This allows the system to function even when traditional internet or network infrastructure is down.
    iii. **Wi-Fi Communication**: The **ESP8266MOD** devices use **Wi-Fi** to communicate with the **Range Extender APs**, which in turn relay data to the **Mikrotik router** and **XAMPP server**. Wi-Fi was chosen for its flexibility and widespread availability, making it ideal for both local and remote communication in the flood-affected areas.

4.3 Testing and Troubleshooting

The implementation phase of the system included thorough testing to ensure that each component works as expected. The testing process involved:

1. **Unit Testing**: Each hardware component (Mikrotik router, Range Extender APs, IoT devices, and XAMPP server) was tested individually to ensure they functioned correctly. For example, IoT devices were tested to verify accurate data collection and successful transmission of data to the XAMPP server.
2. **Network Testing**: The entire network was tested for **connectivity** and **data flow**. The communication between the IoT devices, Range Extender APs, Mikrotik router, and XAMPP server was tested in various scenarios, including:
    i. Normal conditions with all devices functioning correctly.

ii. Simulated failures, such as an AP losing connectivity or the XAMPP server going offline, to assess the system's resilience and **failover capabilities**.

3. **User Interface Testing**: The web interface was tested to ensure that users could access and interpret the data without issues. Tests were conducted on various devices (smartphones, tablets, and laptops) to ensure compatibility and accessibility.
4. **Troubleshooting**: During the testing phase, several issues were identified and addressed:
    i. **Signal interference**: In some areas, the Wi-Fi signals from the **Range Extender Aps** were weak. This was resolved by adjusting the placement of the APs and configuring them to operate on different channels.
    ii. **Power issues**: Some **IoT devices** faced power issues in remote locations. Battery-saving measures, such as **deep sleep mode** in the **ESP8266MOD**, were implemented to extend battery life.

4.4 System Integration

The final stage of the implementation process involved integrating all components into a cohesive system. This included:

i. **Connecting the Mikrotik router** to the **Range Extender APs**, ensuring seamless communication between the network nodes.
ii. **Linking the XAMPP server** to the IoT devices, enabling the server to receive real-time sensor data and process it accordingly.
iii. **Configuring the user interface** to pull data from the server and display it in a user-friendly manner.

The system was thoroughly tested in a **simulated flood scenario**, with real-time water-level data and environmental information displayed on the web interface.[5] The integration of hardware, software, and communication protocols ensured that the system was fully functional and could be deployed in a real-world flood situation.

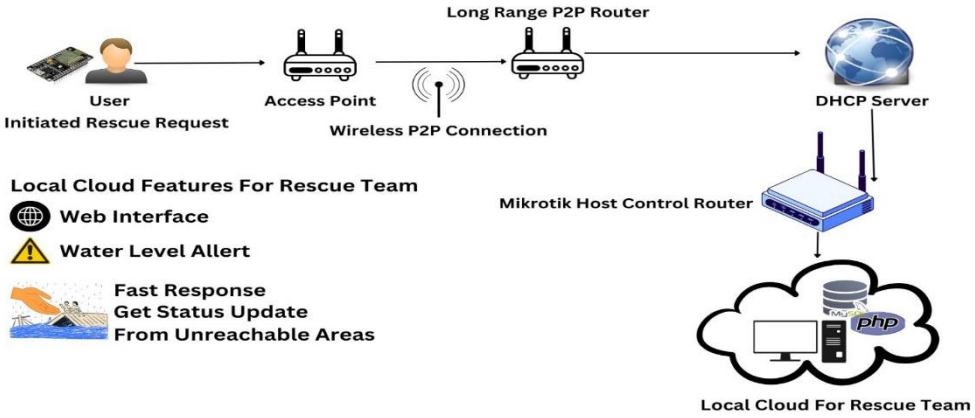


Figure 4.3 : Data flow of SentryLeaf.

CHAPTER 5
RESULTS AND DISCUSSION

5.1 Performance of the Network and IoT Devices

The performance of the **SentryLeaf** system was evaluated in terms of the **range**, **stability of communication**, and the **accuracy** of the data collected by the IoT devices. These factors were critical to ensure the system could provide reliable flood monitoring and support rescue operations in real-time.

1. **Range and Stability of Communication** One of the primary challenges of deploying a flood monitoring system is ensuring that communication remains stable across wide and potentially obstructed areas.[7] The performance of the communication network, including the **Mikrotik RB750r2 hEX lite router** and **Range Extender APs**, was evaluated under different conditions, including:
    i. **Distance**: The network was able to maintain reliable communication over distances of up to 200 meters between devices (depending on environmental factors such as obstacles and interference). The **Range Extender APs** played a crucial role in ensuring that the coverage area was extended beyond the normal range of the router.[12]
    ii. **Obstructions**: In areas with heavy **obstructions** (such as buildings or trees), communication was still stable, though signal strength decreased in certain locations. This was mitigated by optimizing the placement of the Range Extender APs to minimize signal degradation and ensure consistent coverage.
    iii. **Stability**: The system performed well in maintaining stable communication during the tests, with minimal data packet loss. The **P2P communication protocol** proved to be robust, even in challenging environments where traditional cellular networks or infrastructure were unavailable. In instances where communication was lost due to environmental conditions, the system successfully re-established connections once the line-of-sight was restored or the interference subsided.
2. **Accuracy of Data Collected (Water Level)** The **accuracy** of the data collected by the **IoT devices** was tested to ensure that the **water-level sensors** provided reliable information. The results of the tests showed:
    i. **Water-Level Sensors**: The water-level sensors showed high accuracy, with the sensors being able to measure water levels within a range of ±2 cm under controlled conditions. The data was collected continuously, and the sensors accurately captured water level changes over time, providing useful information for flood monitoring and early warning systems.

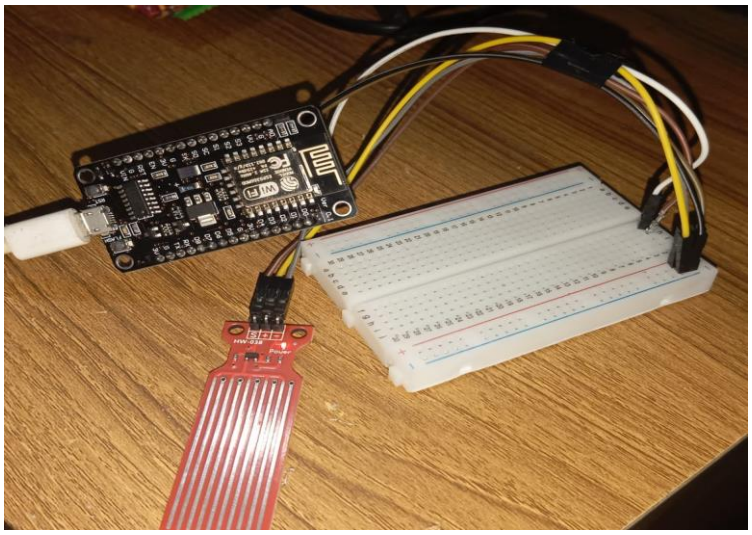


Figure 5.1: Water-Level Sensors and ESP8266MOD Connections.

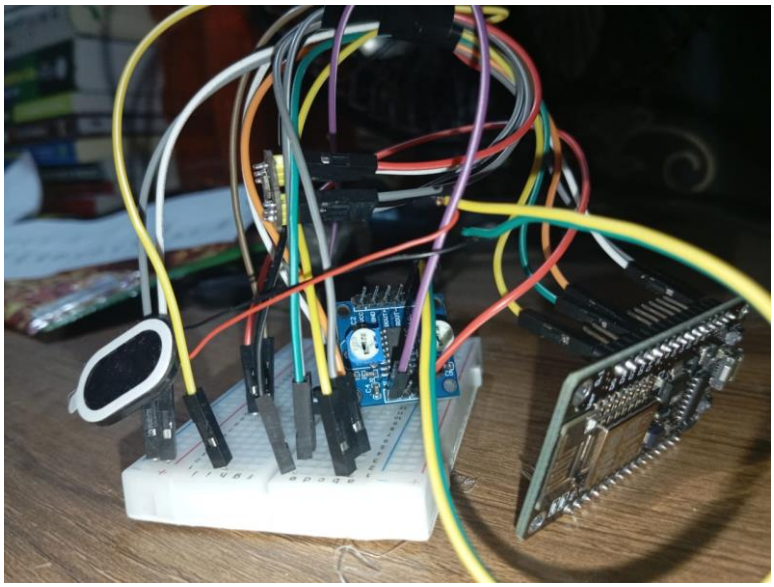


Figure 5.2: Voice sender.

ii. **Environmental Data (Temperature, Humidity, Pressure)**: The environmental sensors, based on the **ESP8266MOD**, collected data such as temperature, humidity, and atmospheric pressure. The accuracy of these readings was found to be within acceptable limits, with discrepancies typically falling within ±3%. [3] The data provided insights into the local conditions and helped verify the overall environmental context of the flood.

5.2 User Experience and Feedback

The user experience and feedback were assessed by interacting with **local authorities**, **emergency responders**, and **test users** who used the **web interface** and engaged with the system in simulated flood scenarios. The findings include:

1. **Interface Usability**: The web interface was generally well-received for its **intuitive design** and ease of use. Users found it easy to access real-time data, view water level trends, and interpret emergency alerts. The interface's visualizations, including **graphs** and **charts**, provided valuable insights into the progress of the flood and potential risks.[11]
2. **Response Time**: The system was praised for its **quick response time** in displaying data. The **real-time updates** from IoT devices ensured that emergency responders could make informed decisions quickly. The data was delivered with minimal delay, enabling fast intervention.
3. **Feedback on Alerts**: Emergency responders appreciated the **alert system** that triggered notifications based on predefined water levels. The system's ability to send **early warnings** about potential flood risks allowed users to take proactive measures and minimize damage to affected areas.
4. **Mobile Compatibility**: Users were able to access the web interface on a variety of devices, including **smartphones**, **tablets**, and **laptops**. The system was fully responsive, which allowed users to monitor flood conditions even while on the move.

Overall, the **user feedback** was overwhelmingly positive, with users noting that the system was highly beneficial in providing real-time flood data and improving decision-making in disaster management.

5.3 Comparison with traditional flood monitoring systems

When compared to traditional flood monitoring systems, **SentryLeaf** demonstrated several advantages:

1. **Cost-Effectiveness**: Traditional flood monitoring systems often rely on **expensive infrastructure** and **sophisticated equipment**. In contrast, **SentryLeaf** utilizes low-cost **IoT devices** (such as **ESP8266MOD**) and **range extenders**, which significantly reduces the overall implementation and maintenance cost.
2. **Decentralized Nature**: Unlike conventional systems that depend on **centralized infrastructure** (such as communication towers or wired networks), **SentryLeaf** operates through **peer-to-peer communication**, making it more resilient and capable of functioning in areas where infrastructure has been destroyed or is otherwise unavailable due to flooding.
3. **Real-Time Data**: Traditional systems often provide data with significant **delays**, and updates may not be frequent enough to support immediate action. In contrast, **SentryLeaf** offers **real-time data transmission**, ensuring that stakeholders can access up-to-the-minute information on flood conditions.
4. **Mobility and Flexibility**: Traditional flood monitoring systems are usually fixed in specific locations, limiting their flexibility. **SentryLeaf**, however, is designed to be **mobile** and **scalable**, allowing it to be deployed quickly in different areas, even as floodwaters change location. This flexibility enables **emergency response teams** to adapt to the evolving situation more effectively.[5]

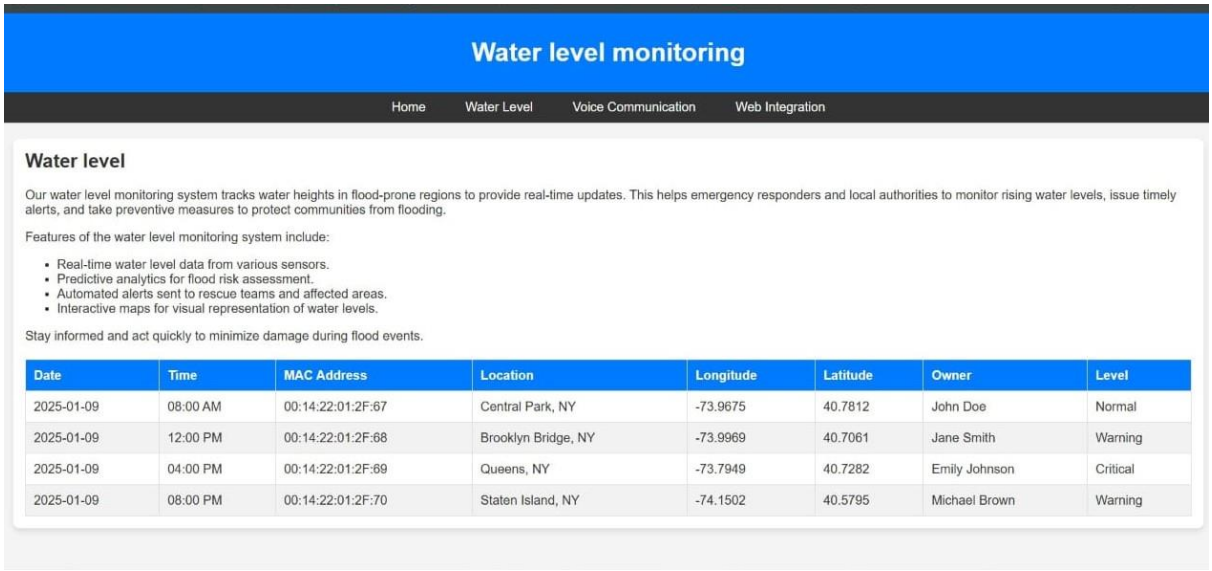


Figure 5.3: Real-Time Water level monitoring Data.

Despite these advantages, traditional systems may still have the edge in terms of **long-term durability** and **scalability**, especially when dealing with extremely large geographical areas or complex flood scenarios. However, **SentryLeaf** fills a significant gap in **emergency response capabilities**, particularly in areas with limited infrastructure.

5.4 Limitations and Challenges Encountered

While the **SentryLeaf** system proved to be effective in many areas, several limitations and challenges were encountered during its implementation:

1. **Power Supply Issues**: Some IoT devices experienced **power challenges** in remote locations where **battery replacement** was not feasible. Although power-saving features were implemented, ensuring the sustainability of the system in long-term, continuous operation remains a challenge. A potential solution is to incorporate **solar-powered charging systems** for the IoT devices.
2. **Signal Interference**: In densely built or heavily obstructed areas, the **Wi-Fi signal strength** from the **Range Extender APs** sometimes struggled to maintain a stable connection, especially in the presence of large amounts of **metal** or other materials that disrupt wireless signals. This issue was mitigated by adjusting the placement of the devices and selecting more optimal frequencies for transmission.
3. **Weather Conditions**: Extreme weather conditions, such as **heavy rain** and **high winds**, can cause **physical damage** to IoT devices and sensors. While the system is designed to withstand moderate environmental conditions, the ruggedness of the hardware in extreme weather conditions needs to be enhanced for improved **durability**.
4. **Scalability in Larger Areas**: While the system was effective in small to medium-sized flood zones, challenges may arise when scaling up to cover **larger regions**. The network's performance may degrade as more devices are added, and additional infrastructure (e.g., more Range Extender APs) may be required to ensure **consistent communication** across vast areas.

CHAPTER 6
CONCLUSION AND FUTURE WORK

6.1 Summary of Findings

The **SentryLeaf** project aimed to develop a **rescue network** using **IoT devices** and **wireless communication technologies** for flood-affected areas. The primary goal was to create a system that could provide real-time flood data, enhance disaster response efforts, and ensure connectivity in areas where traditional infrastructure is often unavailable or damaged. The key findings of the project include:

1. **System Design and Functionality**: The integration of **Mikrotik RB750r2** router, **Range Extender APs,** IoT devices **(ESP8266MOD),** and the **XAMPP server** resulted in a functional and reliable **peer-to-peer communication** system. The **Range Extender APs** successfully expanded the coverage area, ensuring that **IoT devices** were able to transmit real-time data even in challenging environments.
2. **Data Accuracy**: The **water-level sensors** and environmental sensors used in the project provided accurate readings, with water-level sensors achieving an accuracy of ±2 cm, and environmental data like temperature and humidity being collected with acceptable variance (±3%).[4]
3. **User Experience**: The system's **user interface** was well-received for its **intuitive design** and ability to display real-time data, including **water levels**, **environmental conditions**, and **emergency alerts**. This ensured that emergency responders could make timely decisions based on up-to-date information.
4. **Network Stability**: Despite challenges related to signal interference and environmental obstructions, the network was able to maintain stable communication over moderate to long distances. The **peer-to-peer** communication protocol proved effective, especially in flood-affected areas where traditional network infrastructure might be down.

Overall, the **SentryLeaf** system demonstrated its potential as an innovative solution for flood monitoring and disaster management, offering **real-time data**, **cost-effectiveness**, and **resilience** in areas with limited or no infrastructure.

6.2 Contributions of the Project

The **SentryLeaf** project made several important contributions to the field of **disaster management** and **flood monitoring**:

1. **Novel Approach to Communication**: The use of **peer-to-peer communication** via **Range Extender Aps** and **Mikrotik routers** provided a **resilient** and **flexible** method of ensuring network stability in flood-affected areas, where traditional communication infrastructures are often unreliable or nonexistent.
2. **Affordable IoT-based System**: By leveraging low-cost **IoT devices (ESP8266MOD)**, the system provided a **cost-effective** alternative to traditional flood monitoring systems. This makes it accessible for deployment in areas with limited financial resources.
3. **Real-Time Data Monitoring**: The ability to provide **real-time water level** and **environmental data** enabled early warnings for **flood risks**, improving the response time of emergency teams and reducing the impact of floods on affected communities.

4. **Scalable and Mobile Solution**: The **SentryLeaf** system is designed to be **mobile** and **scalable**, allowing it to be rapidly deployed in various regions based on the need for flood monitoring and disaster response. The system can be expanded as required by adding more **Range Extender APs** and **IoT devices** to cover larger areas.

6.3 Potential for Future Improvements and Scalability

While **SentryLeaf** has proven to be an effective solution for flood monitoring and disaster management, several potential improvements could further enhance the system's performance and scalability:

1. **Power Supply Solutions**: A major limitation of the system was the **battery life** of the **IoT devices**. Future iterations could incorporate **solar-powered charging systems** for **remote sensors** and **IoT devices** to ensure continuous operation in areas where access to electricity is limited.
2. **Network Optimization**: Although the system demonstrated strong performance, the network's communication range and signal strength could be further improved. This can be achieved by enhancing the **placement** and **configuration** of the **Range Extender Aps** and exploring the use of higher-capacity routers and advanced communication technologies such as **LoRa WAN** for **long-range communication** in challenging environments.
3. **Data Integration and Analytics**: The future development of the **SentryLeaf** system could involve the integration of advanced **data analytics** capabilities, such as **machine learning** algorithms, to predict flood patterns, identify potential flood risks, and enhance decision-making. This would provide a more proactive approach to disaster management.
4. **Improved Durability for Extreme Conditions**: The current system faced challenges in harsh environmental conditions (e.g., heavy rainfall and extreme temperatures). The use of **ruggedized IoT devices** and **weather-resistant enclosures** would improve the system’s durability in extreme weather scenarios.
5. **Integration with Government Systems**: In future iterations, the system could be integrated with local **government disaster response systems** and **early warning networks**. This would allow for **faster dissemination** of flood data and **coordinated emergency response** efforts across different sectors.

6.4 Recommendations for Further Research and Development

1. **Expanded Communication Technologies**: Further research should be conducted into exploring new wireless communication technologies like **LoRa**, **5G**, or **NB-IoT**, which offer **longer range** and **low power consumption**, making them ideal for flood monitoring systems in remote areas.
2. **Extended Environmental Monitoring**: Future developments could include the addition of more **environmental sensors**, such as **soil moisture** or air **quality sensors**, to provide more comprehensive data for flood prediction and overall environmental monitoring.
3. **Real-Time Decision Support Systems**: Research into **real-time decision support systems** using AI and machine learning could enhance the **SentryLeaf** system. These systems could automatically analyze sensor data, predict flood risks, and provide actionable insights to authorities and emergency teams.

4. **Collaborative Platforms for Disaster Management**: Further development of **collaborative platforms** that allow various stakeholders (e.g., local authorities, emergency responders, NGOs) to access the real-time data from **SentryLeaf** could improve coordination in response efforts during flood events.

REFERENCES

[1] Kreibich, H., Pech, I., Schröter, K., Müller, M. and Thieken, A. (2016, April). Floods in 2002 and 2013: comparing flood warnings and emergency measures from the perspective of affected parties. In EGU General Assembly Conference Abstracts (Vol. 18, p. 12611).

[2] H.L. Cloke *et al.* Ensemble flood forecasting: a review J. Hydrol. (2009)

[3] S. Drobot *et al.* Advances and challenges in flash flood warnings Environ. Hazard (2007)

[4] S. Kheradmand *et al.*Evaluation of adaptation options to flood risk in a probabilistic framework J. Hydrol. Reg. Stud. (2018)

[5] Report of the open-ended intergovernmental expert working group on indicators and terminology relating to disaster risk reduction United Nations General Assembly. Agenda Item (2017)

[6] J. Thielen-del Pozo *et al.*The Benefit of Continental Flood Early Warning Systems to Reduce the Impact of Flood Disasters (2015)

[7] The impact of flooding on food security across Africa 2022, Proceedings of the National Academy of Sciences of the United States of America

[8] Helal, Sumi, et al. "An architecture for wireless LAN/WAN integration." 2000 IEEE Wireless Communications and Networking Conference. Conference Record (Cat. No. 00TH8540). Vol. 3. IEEE, 2000.

[9] Vassis, Dimitris, et al. "The IEEE 802.11 g standard for high data rate WLANs." IEEE network 19.3 (2005): 21-26.

[10] Deshpande, Suyog D. "Software implementation of IEEE 802.11 b wireless LAN standard." Proceeding of the SDR 04 Technical Conference and Product Exposition. 2004.

[11] Environmental monitoring systems: A review,A Kumar, H Kim, GP Hancke - IEEE Sensors Journal, 2012

[12] Monitoring system for photovoltaic plants: A review SR Madeti, SN Singh - Renewable and Sustainable Energy Reviews, 2017 - Elsevier

APPENDIX

Access the Router

LAN ports (ports 3) > IP configuration to automatic (DHCP) > http://192.168.88.1 > Login admin.

#Assign IP Addresses to Ethernet Interfaces:

/ip address add address=192.168.88.1/24 interface=ether2

Configure DHCP Server

/ip pool add name=dhcp_pool ranges=192.168.88.2-192.168.88.254
/ip dhcp-server add name=dhcp1 interface=ether2 address-pool=dhcp_pool lease-time=30m disabled=no
/ip dhcp-server network add address=192.168.88.0/24 gateway=192.168.88.1 dns-server=192.168.88.1

Set Up Firewall Rules

Allow established and related connections
/ip firewall filter add chain=input connection-state=established,related action=accept

Drop invalid connections
/ip firewall filter add chain=input connection-state=invalid action=drop

Allow LAN access to the router
/ip firewall filter add chain=input in-interface=ether2 action=accept

Drop all other incoming traffic
/ip firewall filter add chain=input action=drop

NAT configuration for internet access
/ip firewall nat add chain=srcnat out-interface=ether1 action=masquerade

Save Configuration

/system backup save name=initial_config

#Water level ESP Source code

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>

const char* ssid = "leaf";
const char* password = "25802580";

const String serverURL = "https://192.168.2.100/water_level_handler.php";

#define SENSOR_PIN A0
const int waterLevelThreshold = 50;

const String deviceLocation = "H-78,Bijoynogor";
const String deviceLat = "24.018591";
const String deviceLong = "91.218145";
const String ownerName = "Rashedul Islam";

void setup() {
  Serial.begin(115200);
  pinMode(SENSOR_PIN, INPUT);
  Serial.println("Connecting to WiFi...");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nConnected to WiFi");
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP());
}

void loop() {
  int sensorValue = analogRead(SENSOR_PIN);
  float waterLevelPercentage = (sensorValue / 1023.0) * 100.0;

  Serial.print("Water Level: ");
  Serial.print(waterLevelPercentage);
  Serial.println("%");

  if (waterLevelPercentage > waterLevelThreshold) {
    sendAlert(waterLevelPercentage);
  }

  delay(5000);
}

void sendAlert(float waterLevel) {
```

```
  if (WiFi.status() == WL_CONNECTED) {
    WiFiClientSecure client;
    client.setInsecure();

    HTTPClient https;
    String macAddress = WiFi.macAddress();

    String encodedLocation = urlEncode(deviceLocation);
    String encodedOwner = urlEncode(ownerName);

    String url = serverURL + "?mac=" + macAddress + "&location=" + encodedLocation +
"&lat=" + deviceLat + "&long=" + deviceLong + "&owner=" + encodedOwner + "&level="
+ String(waterLevel);

    Serial.println("Sending GET request...");
    Serial.println("URL: " + url);

    if (https.begin(client, url)) {
      int httpCode = https.GET();

      if (httpCode > 0) {
        Serial.println("HTTP Response Code: " + String(httpCode));
        Serial.println("Response: " + https.getString());
      } else {
        Serial.println("Error in sending request: " + https.errorToString(httpCode));
      }

      https.end();
    } else {
      Serial.println("Unable to connect to server");
    }
  } else {
    Serial.println("WiFi not connected");
  }
}

String urlEncode(String str) {
  String encoded = "";
  for (int i = 0; i < str.length(); i++) {
    char c = str[i];
    if (isalnum(c)) {
      encoded += c;
    } else {
      encoded += '%' + String(c, HEX);
    }
  }
  return encoded;
}
```

WEB Interface All Source code

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="assets/css/style.css">
    <link rel="stylesheet" href="style.css">
    <title>Flood Rescue Login</title>
</head>
<body>
    <div class="login-container">
        <h1>Flood Rescue System</h1>
        <form onsubmit="return validateCredentials(event)">
            <label for="username">Username</label>
            <input type="text" id="username" placeholder="Enter your username" required>

            <label for="password">Password</label>
            <input type="password" id="password" placeholder="Enter your password"
required>

            <p id="error-message" class="error" style="display: none;">Invalid username or
password</p>

            <button type="submit">Login</button>
        </form>
    </div>

    <script>
        function validateCredentials(event) {
            event.preventDefault();

            const correctUsername = "flood2025";
            const correctPassword = "2025";

            const enteredUsername = document.getElementById('username').value;
            const enteredPassword = document.getElementById('password').value;

            if (enteredUsername === correctUsername && enteredPassword ===
correctPassword) {
                window.location.href = 'dashboard.html';
            } else {
```

```
            document.getElementById('error-message').style.display = 'block';
        }
    }
  </script>
</body>
</html>
```

 # Dashboard.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="style.css">
  <title>Dashboard - Flood Rescue System</title>
</head>
<body>
  <header>
    <h1>Dashboard</h1>
  </header>

  <nav>
    <ul>
      <li><a href="dashboard.html">Home</a></li>
      <li><a href="water.html">Water Level</a></li>
      <li><a href="voice.html">Voice Communication</a></li>
      <li><a href="web.html">Web Integration</a></li>
    </ul>
  </nav>

  <main>
    <section>
      <h2>Welcome to the Flood Rescue Dashboard</h2>
      <p>Select a menu option to view the respective information.</p>
    </section>

    <div class="logout-container">
      <button onclick="logout()">Logout</button>
    </div>
```

```
        <script>
            function logout() {
                window.location.href = 'index.html';
            }
        </script>
    </main>
</body>
</html>
```

 # Voice.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Voice Communication</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <header>
        <h1>Voice Communication</h1>
    </header>

    <nav>
        <ul>
            <li><a href="dashboard.html">Home</a></li>
            <li><a href="water.html">Water Level</a></li>
            <li><a href="voice.html">Voice Communication</a></li>
            <li><a href="web.html">Web Integration</a></li>
        </ul>
    </nav>

    <main>
        <section id="voice">
            <h2>Voice Communication</h2>
            <p>Below is the received voice message:</p>
            <div id="voice-display">
                <p>No voice message received yet.</p>
            </div>
```

```
        <form id="voice-form">
            <label for="voice-input">Send a voice message:</label>
            <input type="file" id="voice-input" accept="audio/*">
            <button type="submit">Send</button>
        </form>
    </section>
</main>

<script>
    const voiceForm = document.getElementById('voice-form');
    const voiceInput = document.getElementById('voice-input');
    const voiceDisplay = document.getElementById('voice-display');

    voiceForm.addEventListener('submit', function (event) {
        event.preventDefault();

        if (voiceInput.files.length > 0) {
            const audioFile = voiceInput.files[0];
            const audioURL = URL.createObjectURL(audioFile);

            voiceDisplay.innerHTML = `
                <audio controls>
                    <source src="${audioURL}" type="${audioFile.type}">
                    Your browser does not support the audio element.
                </audio>
            `;
        } else {
            alert("Please select a voice file before sending!");
        }
    });
</script>
</body>
</html>
```

 # Water.html

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="style.css">
    <link rel="stylesheet" href="table.css">
    <link rel="stylesheet" href="dashboard.html">
</head>
<body>
    <header>
        <h1>Water level monitoring</h1>
    </header>

    <nav>
        <ul>
            <li><a href="dashboard.html">Home</a></li>
            <li><a href="water.html">Water Level</a></li>
            <li><a href="voice.html">Voice Communication</a></li>
            <li><a href="web.html">Web Integration</a></li>
        </ul>
    </nav>

    <main>
        <section>
            <h2>Water level </h2>
            <p>Our water level monitoring system tracks water heights in flood-prone regions to provide real-time updates. This helps emergency responders and local authorities to monitor rising water levels, issue timely alerts, and take preventive measures to protect communities from flooding.</p>
            <p>Features of the water level monitoring system include:</p>
            <ul>
                <li>Real-time water level data from various sensors.</li>
                <li>Predictive analytics for flood risk assessment.</li>
                <li>Automated alerts sent to rescue teams and affected areas.</li>
                <li>Interactive maps for visual representation of water levels.</li>
            </ul>
            <p>Stay informed and act quickly to minimize damage during flood events.</p>

            <table>
```

```
<thead>
  <tr>
    <th>Date</th>
    <th>Time</th>
    <th>MAC Address</th>
    <th>Location</th>
    <th>Longitude</th>
    <th>Latitude</th>
    <th>Owner</th>
    <th>Level</th>
  </tr>
</thead>
<tbody>
  <tr>
    <td>2025-01-09</td>
    <td>08:00 AM</td>
    <td>00:14:22:01:2F:67</td>
    <td>Central Park, NY</td>
    <td>-73.9675</td>
    <td>40.7812</td>
    <td>John Doe</td>
    <td>Normal</td>
  </tr>
  <tr>
    <td>2025-01-09</td>
    <td>12:00 PM</td>
    <td>00:14:22:01:2F:68</td>
    <td>Brooklyn Bridge, NY</td>
    <td>-73.9969</td>
    <td>40.7061</td>
    <td>Jane Smith</td>
    <td>Warning</td>
  </tr>
  <tr>
    <td>2025-01-09</td>
    <td>04:00 PM</td>
    <td>00:14:22:01:2F:69</td>
    <td>Queens, NY</td>
    <td>-73.7949</td>
```

```
                <td>40.7282</td>
                <td>Emily Johnson</td>
                <td>Critical</td>
            </tr>
            <tr>
                <td>2025-01-09</td>
                <td>08:00 PM</td>
                <td>00:14:22:01:2F:70</td>
                <td>Staten Island, NY</td>
                <td>-74.1502</td>
                <td>40.5795</td>
                <td>Michael Brown</td>
                <td>Warning</td>
            </tr>
        </tbody>
    </table>
</section>
</main>
</body>
</html>
```

 # Web.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <header>
        <h1>Web Integration</h1>
    </header>

    <nav>
        <ul>
            <li><a href="dashboard.html">Home</a></li>
            <li><a href="water.html">Water Level</a></li>
```

```
        <li><a href="voice.html">Voice Communication</a></li>
        <li><a href="web.html">Web Integration</a></li>
      </ul>
    </nav>

    <main>
      <section id="web">
        <h2>Web Integration</h2>
        <p>The Web Integration component allows the public, rescue teams, and authorities to access critical flood information through an intuitive online platform. It consolidates data from various sources, including water level sensors, weather forecasts, and live updates from rescue operations, making it easier for users to monitor and act accordingly.</p>
        <p>Features of the Web Integration platform include:</p>
        <ul>
          <li>Interactive maps showing real-time flood data and affected areas.</li>
          <li>Live updates and alerts related to water levels, weather conditions, and rescue operations.</li>
          <li>Data visualization tools for understanding flood patterns and risks.</li>
          <li>Public access to emergency preparedness information and resources.</li>
          <li>Integration with social media for sharing real-time information and updates.</li>
        </ul>
        <p>This web-based platform is essential for ensuring that everyone involved in the rescue process has access to up-to-date, accurate information to make informed decisions during a flood emergency.</p>
      </section>
    </main>
</body>
</html>
```

 # Style.css

```
/* General Styles */
body {
    font-family: Arial, sans-serif;
    margin: 0;
```

```
    padding: 0;
    background: #f4f4f4;
    color: #333;
}

/* Login Page */
.login-container {
    max-width: 400px;
    margin: 100px auto;
    padding: 20px;
    background: white;
    border-radius: 8px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}

.login-container h1 {
    text-align: center;
    margin-bottom: 20px;
}

.login-container label {
    display: block;
    margin: 10px 0 5px;
}

.login-container input {
    width: 100%;
    padding: 10px;
    margin-bottom: 20px;
    border: 1px solid #ddd;
    border-radius: 4px;
}

.login-container button {
    width: 100%;
    padding: 10px;
    background: #007BFF;
    color: white;
    border: none;
```

```
    border-radius: 4px;
    cursor: pointer;
}

.login-container button:hover {
    background: #0056b3;
}

/* Remember Me Checkbox Styling */
.remember-me {
    display: flex;
    align-items: center;
    margin-bottom: 20px;
}

.remember-me input[type="checkbox"] {
    margin-right: 10px;
}

/* Dashboard */
header {
    background: #007BFF;
    color: white;
    padding: 10px;
    text-align: center;
}

nav {
    background: #333;
    color: white;
}

nav ul {
    list-style: none;
    padding: 0;
    margin: 0;
    display: flex;
    justify-content: center;
}
```

```
nav ul li {
    margin: 0 10px;
}

nav ul li a {
    color: white;
    text-decoration: none;
    padding: 10px 15px;
    display: inline-block;
}

nav ul li a:hover {
    background: #555;
    border-radius: 4px;
}

main {
    padding: 20px;
}

main section {
    margin-bottom: 20px;
    padding: 15px;
    background: white;
    border-radius: 8px;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}

main section h2 {
    margin-top: 0;
}

/* Centered Logout Button */
.logout-container {
    display: flex;
    justify-content: center;
    margin-top: 30px;  /* Optional: Adds some space from the content */
}
```

```
.logout-container button {
    padding: 10px 20px;
    background-color: #ff4d4d;
    color: white;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-size: 16px;
}

.logout-container button:hover {
    background-color: #ff1a1a;
}
```

 # ESP8266MOD PHP Handler

```
<?php
header('Content-Type: application/json');

if (isset($_GET['mac']) && isset($_GET['location']) && isset($_GET['lat']) &&
isset($_GET['long']) && isset($_GET['owner']) && isset($_GET['level'])) {
    $mac = htmlspecialchars($_GET['mac']);
    $location = htmlspecialchars($_GET['location']);
    $latitude = htmlspecialchars($_GET['lat']);
    $longitude = htmlspecialchars($_GET['long']);
    $owner = htmlspecialchars($_GET['owner']);
    $waterLevel = htmlspecialchars($_GET['level']);

    $logFile = 'water_level_log.txt';
    $logEntry = date('Y-m-d H:i:s') . " - MAC: $mac, Location: $location, Latitude:
$latitude, Longitude: $longitude, Owner: $owner, Water Level:
$waterLevel%\n";

    file_put_contents($logFile, $logEntry, FILE_APPEND);

    echo json_encode([
        'status' => 'success',
        'message' => 'Data received successfully',
        'received_data' => [
```

```
            'mac' => $mac,
            'location' => $location,
            'latitude' => $latitude,
            'longitude' => $longitude,
            'owner' => $owner,
            'water_level' => $waterLevel
        ]
    ]);
} else {
    echo json_encode([
        'status' => 'error',
        'message' => 'Missing required parameters'
    ]);
}
?>
```