Faculty of Computer Science and Information Technology

Tunnel Flood Detection System with Barrier Control and Notification Mechanism

NAKSATRA PREMANAND

Faculty of Computer Science and Information Technology

UNIVERSITY MALAYSIA SARAWAK

2025

Tunnel Flood Detection System with Barrier Control and Notification Mechanism

NAKSATRA PREMANAND

This project is submitted in partial fulfilment of the requirements for the degree of Bachelor of Computational Science with Honors

Faculty of Computer Science and Information Technology

UNIVERSITY MALAYSIA SARAWAK

2025

DECLARATION

I hereby declare that this report is my original work and has been prepared solely for academic purposes. All references, sources, and contributions by others have been duly acknowledged.

Signed,

……..

NAKSATRA PREMANAND 22 June 2025

Faculty of Computer Science and Information Technology

University of Malaysia Sarawak

ACKNOWLEDGEMENT

I wish to convey my sincere appreciation to all those who have assisted me during the progression of this project. Primarily, I would like to acknowledge my profound gratitude to my supervisor, Dr Sze San Nah, for their essential direction, insightful critiques, and unwavering support, which have played a crucial role in the achievement of this project.

I want to express my deepest gratitude to my family and friends for their support, patience, and encouragement when things get tough during this project. Their support and encouragement mean a lot to me; without that, this project wouldn't have been the same.

TABLE OF CONTENTS

DECLARATION 4
ACKNOWLEDGEMENT 5
TABLE OF CONTENTS 6
LIST OF FIGURES 9
LIST OF TABLES 10
LIST OF ABBREVIATIONS 11
ABSTRACT 12
ABSTRAK 13
CHAPTER 1: INTRODUCTION 14
1.1 Introduction 14
1.2 Problem Statement 15
1.3 Scope 16
1.4 Brief Methodology 19
1.5 Significance of Project 21
1.6 Project Schedule 22
1.7 Expected Outcome 23
1.8 Summary 24
CHAPTER 2: LITERATURE REVIEW 25
2.1 Introduction 25
2.2 Review on Similar Existing Systems 26
2.2.1 Flood Monitoring and Alerting System 26
2.2.2 Automated flood water level sensor and alarm system using Arduino Uno 29
2.2.3 Iot Based Early Flood Detection System with Arduino and Ultrasonic Sensors in Flood-Prone Areas. 30
2.3 Table of Comparison between Projects Systems. 33
2.4 Review on Tools and Technology 36
2.4.1 Hardware Component 36
2.4.2 Software Component 37
2.5 Summary 38
CHAPTER 3: REQUIREMENT ANALYSIS & DESIGN 39
3.1 Introduction 39
3.2 Project Methodology 39
3.2.1 Planning and Analysis 41
3.2.1.2 Analysis 41
3.2.1.3 Analysis of current methods 41

3.2.1.4 Analysis of Proposed System 42
3.2.1.5 Software Requirements 43
3.2.1.6 Hardware Requirements 45
3.2.3 Design 47
3.2.3.1 System Architecture 48
3.2.3.2 Flowchart 49
3.2.3.3 Context diagram 50
3.2.3.4 DFD level O 51
3.2.3.5 Entity Relationship Diagram (ERD) 52
3.2.3.6 Data Dictionary 53
3.2.4 Development Phase 56
3.2.5 Testing 57
3.2.6 Deploy and Maintenance 58
3.3 Summary 59
Chapter 4: Development and Implementation 60
4.1 Introduction 60
4.2 System architecture implementation 60
4.3 Hardware Implementation 62
4.4 Software Implementation 65
4.4.1 Arduino IDE Installation 65
4.4.2 NodeMCU Board Setup 67
4.4.3 XAMPP Installation and Configuration 68
4.4.4 Database Setup using phpMyAdmin 69
4.4.5 Serial Monitor Debugging 70
4.4.6 Website (HTML, CSS, JavaScript) Implementation 71
4.4.7 Email Notification Setup 72
4.5 Prototype Coding 73
4.5.1 NodeMCU ESP8266 73
4.5.2 Arduino Uno 75
4.6 System Modules Application 77
4.6.1 Home Module 77
4.6.2 Real-Time Data Module 78
4.6.3 Precautions Module 78
4.6.4 Contacts Module 79
4.6.5 Rainfall Module 80
4.7 Conclusion 81
CHAPTER 5: CONCLUSION AND FUTURE WORK 82

5.1 Introduction 82
5.2 Achievements 83
5.3 Problems Encountered 84
5.4 Future Work 85
5.5 Conclusion 86
REFERENCES 87

LIST OF FIGURES

Figure 1.1: Gantt Chart for the System Development Timeline FYP 1 22
Figure 1.2: Gantt Chart for the System Development Timeline FYP 2 22
Figure 2.1: Working Model (Dhebe et al., 2023) 27
Figure 2.2: Realtime data showing on ThingSpeak Web App (Dhebe et al., 2023) 27
Figure 2.3: Flowchart of the system (Dhebe et al., 2023) 28
Figure 2.4: Actual Product of Automated flood water level sensor and alarm system using Arduino Uno (Magnaye et al., 2024) 29
Figure 2.5: Application block diagram (Darwis et al., 2023) 30
Figure 2.6: Application Flowchart (Darwis et al., 2023) 31
Figure 2.7: Schematic circuit of device and component (Darwis et al., 2023) 32
Figure 2.8: Display of water level notification via email 32
Figure 3.1: System Development Life Cycle (SDLC) (Sinha & Sinha, 2021) 40
Figure 3.2: System architecture of the proposed system 48
Figure 3.3: Flowchart of the system 49
Figure 3.4: Context Diagram of the system 50
Figure 3.5: DFD level 0 of the system 51
Figure 3.6: Entity Relationship Diagram of the system 52
Figure 4.1 Implemented System Architecture 60
Figure 4.2 Complete circuit wiring setup 63
Figure 4.3 Miniature town model 64
Figure 4.4 Arduino IDE download page 65
Figure 4.5 Selecting Arduino Uno COM port 66
Figure 4.6 Adding ESP8266 board URL 67
Figure 4.7 Selecting NodeMCU board 67
Figure 4.8 XAMPP download page 68
Figure 4.9 XAMPP control panel 69
Figure 4.10 phpMyAdmin homepage 69
Figure 4.11 Readings table structure 70
Figure 4.12 Serial motor readings 70
Figure 4.13 Web dashboard interface 71
Figure 4.14 Web dashboard interface 72
Figure 4.15 NodeMCU ESP8266 Program Code (Part 1) 73
Figure 4.16 NodeMCU ESP8266 Program Code (Part 2) 73
Figure 4.17 NodeMCU ESP8266 Program Code (Part 3) 74
Figure 4.18 NodeMCU ESP8266 Program Code (Part 4) 74
Figure 4.19 Arduino Uno Program Code (Part 1) 75
Figure 4.20 Arduino Uno Program Code (Part 2) 75
Figure 4.21 Arduino Uno Program Code (Part 3) 76
Figure 4.22 Flood Alert Home Page Interface 77
Figure 4.23 Real-Time Water Level and Status Display 78
Figure 4.24 Flood Safety Tips Page 79
Figure 4.25 State Water Department Contact Numbers 79
Figure 4.26 Rainfall Intensity Image and Water Level Table 80

LIST OF TABLES

Table 2.1: The comparison of functionalities between similar existing systems and the proposed project 32
Table 3.1: Data dictionary for sensor 52
Table 3.2: Data dictionary for alert..... 53
Table 3.3: Data dictionary for authority 54
Table 3.4: Data dictionary for traffic control..... 54

LIST OF ABBREVIATIONS

IoT	Internet of Things
MCU	Microcontroller Unit
LoRa	Long Range Communication Module
IDE	Integrated Development Environment
XAMPP	Cross-Platform Apache, MySQL, PHP, Perl
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
PHP	Hypertext Preprocessor
JS	JavaScript
RM	Ringgit Malaysia
SDLC	Software Development Life Cycle

ABSTRACT

This report discusses the development of a Tunnel Flood Detection System with Barrier Control and Notification Mechanism designed specifically to address the challenges of flooding in Malaysia, especially in tunnels and low-lying areas. The system uses ultrasonic sensors for water level detection, an Arduino Uno as the microcontroller, and NodeMCU with LoRa modules for wireless communication. A web-based platform allows for the visualization of real-time data and the generation of alerts. The system integrates automatic barriers, traffic lights, and a buzzer that work together to control vehicular access in view of public safety with regard to flood scenarios. The use of XAMPP as a local server and MySQL for data storage ensures reliable and efficient backend operations. The project has the objective to improve flood management, raise safety, and shorten response times by integrating hardware and software components. A physical model was developed to simulate real-life situations, thus showing the effectiveness of the system.

ABSTRAK

Laporan ini mengulas pembangunan Sistem Pengesanan Banjir Terowong dengan Kawalan Halangan dan Mekanisme Pemberitahuan, yang direka untuk menangani cabaran banjir di Malaysia, terutamanya di kawasan rendah dan terowong. Sistem ini mengintegrasikan penderia ultrasonik untuk pengesanan paras air, Arduino Uno sebagai mikropengawal, serta modul NodeMCU untuk komunikasi. Platform berasaskan web yang dibangunkan menyediakan visualisasi data masa nyata serta pemberitahuan. Sistem ini juga dilengkapi dengan halangan automatik, lampu isyarat, dan buzzer untuk menguruskan akses kenderaan serta memastikan keselamatan awam semasa banjir. Penggunaan XAMPP sebagai pelayan tempatan dan MySQL untuk penyimpanan data memastikan operasi backend yang boleh dipercayai dan cekap. Dengan menggabungkan komponen perkakasan dan perisian, projek ini bertujuan untuk meningkatkan pengurusan banjir, meningkatkan keselamatan, dan mempercepatkan masa tindak balas. Model fizikal dibina untuk mensimulasikan aplikasi dunia sebenar, membuktikan keberkesanan sistem ini.

CHAPTER 1: INTRODUCTION

1.1 Introduction

Flooding is nevertheless a continuous and growing problem in Malaysia, particularly in metropolitan areas vulnerable to flash floods during monsoon seasons. Floods have become more severe and frequent due to climate change and urbanization, endangering infrastructure and public safety. As critical components of urban transportation, tunnels are particularly vulnerable due to their underground nature, which may readily amass water and imprison vehicles and people during flash floods.

Traditional flood management solutions, such as the SMART Tunnel in Kuala Lumpur, employ large-scale water diversion and control strategies; nevertheless, coverage is restricted and does not avoid flooding in other urban tunnels that lack specific flood monitoring systems. This project aims to develop a real-time flood detection and response system specifically for tunnels, aiming to reduce hazards by rapid barrier control and automatic alerts to relevant authorities, hence improving public safety and response time.

1.2 Problem Statement

Flooding in Malaysian tunnels, exacerbated by severe rainfall and climate instability, poses a serious infrastructure concern. Floods have gotten more severe in recent years, with the 2021-2022 monsoon season costing an estimated RM6.1 billion in damage across Malaysia. These instances have resulted in road closures, infrastructure failures, and even fatalities, underscoring the urgent need for improved flood detection systems in vital infrastructures such as tunnels.

Tunnels in metropolitan areas frequently lack dedicated flood warning systems, resulting in delayed reactions and serious safety hazards. Tunnel environments remain vulnerable to sudden flooding due to the lack of an automated system to monitor and respond to rising water levels, resulting in accidents, damage, and costly emergency responses. This project tackles these issues by providing a sensor-based flood detection and barrier control system that monitors water levels in tunnels, restricts access when necessary, and sends timely notifications to local authorities, reducing risk and enhancing flood response.

1.3 Scope

The scope of the project is as follows:

i) Real-time Water Level Monitoring: To continuously check water levels, the system employs ultrasonic sensors positioned thoughtfully throughout the tunnel. These sensors enable prompt identification of possible flood conditions by providing precise, immediate information on rising water levels. To ensure timely detection and response to flooding incidents, the collected data is routed to a central system for processing and visualization.

ii) Barrier Control Mechanism: The system automatically closes tunnel barriers to limit access when water levels surpass a preset safety threshold. This automated system reduces the chance of accidents and guarantees public safety during flood events by keeping cars and pedestrians out of hazardous, wet areas.

iii) Notification System: As soon as flooding conditions are identified, local authorities are notified via an integrated real-time alert mechanism. An online interface that provides comprehensive information about the water levels and barrier status is used to send alerts. This lowers the possibility of delays and increases the effectiveness of flood management overall by guaranteeing prompt and coordinated emergency response actions.

Physical Model Development: To simulate actual conditions, a smaller physical model of a tunnel will be built. To illustrate how the flood detection and response mechanism works, this model will include water level sensors, barriers, and a notification system. It functions as a physical prototype for system testing and improvement.

iv) Website for Data Visualization: Developing an intuitive, user-friendly website that displays real-time data on water levels, barrier status, and alarms for authorities to track remotely.

Aims and Objectives

i. Water Level Detection: Create a sensor system that can precisely measure water levels in real-time and transmit data to a central monitoring system.

ii. Automated Barrier Activation: Install a barrier control system that automatically responds when water levels rise over acceptable levels, prohibiting vehicles from entering flooded tunnels.

iii. Notification Alerts: Implement a web-based alert system to notify authorities as soon as flooding conditions are recognized, allowing for faster, coordinated responses.

iv. Physical Model and Demonstration: Create a workable model of the tunnel flood detection system to demonstrate functionality and facilitate testing.

v. Traffic Light Control System: Create a traffic light guidance system that signals vehicles about tunnel accessibility based on water level.

1.4 Brief Methodology

System Design and Development.

i) Programming: Using Arduino and C/C++, create sensors that measure water levels within the tunnel model. Set the barrier control mechanism to respond at certain threshold values.

ii) Sensor Integration: Sensors will be inserted at strategic spots throughout the tunnel model to ensure accurate monitoring of water levels. When these levels exceed a specified threshold, the barrier control system activates automatically.

Physical Model Construction

i) Create a physical model of a tunnel with embedded sensors and barrier gates. This model will be used as a prototype to replicate flood conditions and illustrate the system's response.

Web Development

i) Create a website using HTML, CSS, and JavaScript to display sensor data, barrier status, and live alarms. The website will serve as a platform for local authorities to monitor flood conditions in real-time.

Testing and Calibration

i) Test the system at a variety of simulated water levels to confirm accurate detection, quick barrier reaction, and adequate alert functioning. Calibration will ensure that the device can accurately distinguish between safe and unsafe water levels.

Documentation and Reporting

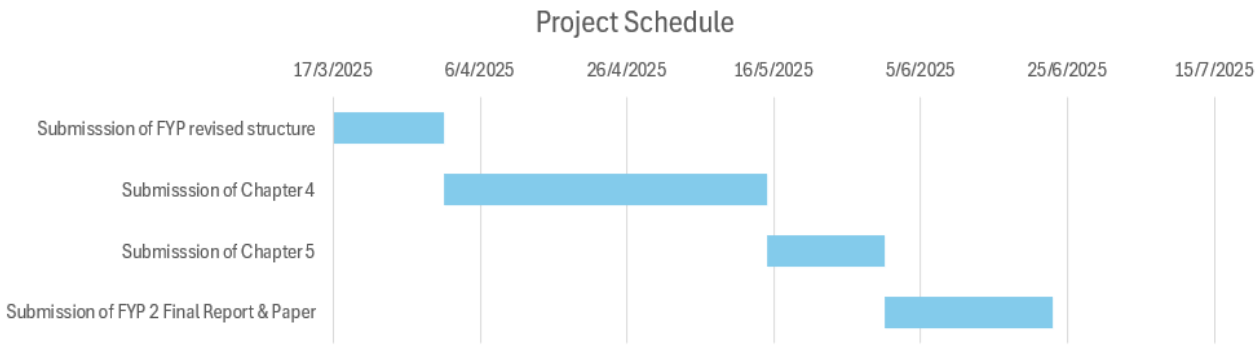
i) Document all stages of system development, including design specifications, coding techniques, and testing findings, to guarantee that the flood detection system is thoroughly analyzed and reproducible.

1.5 Significance of Project

This project is extremely important in developing tunnel safety and flood resistance in Malaysia. Incorporating automatic flood detection and control techniques, lowers the need for manual monitoring and intervention, improving response times. The system's notification component provides authorities with timely information, allowing for quick, coordinated actions to flood incidents. In a larger sense, this project helps to strengthen Malaysia's flood management strategy and infrastructure resilience, indicating a proactive approach to addressing climate-related hazards.

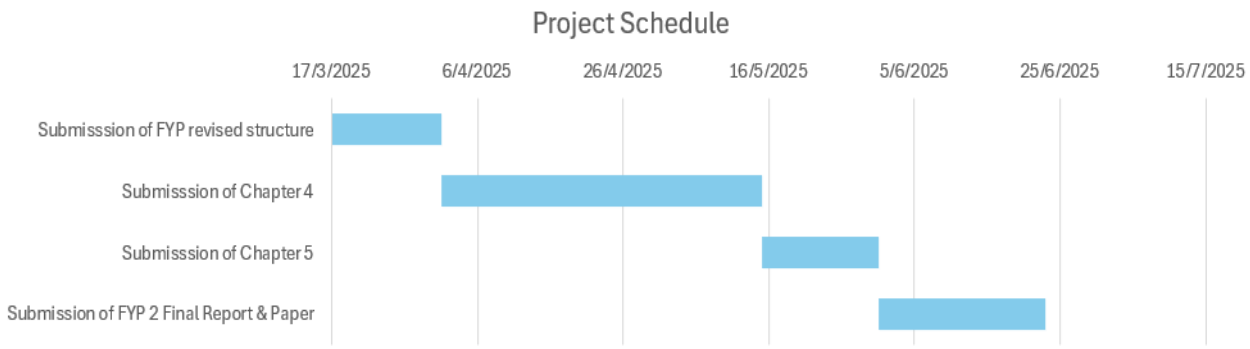
1.6 Project Schedule

The Gantt charts below show the project's major checkpoints and due dates, including submitting proposals, finishing chapters, and final evaluations, all planned for October 2024 until June 2025.



	Submission of FYP 2 Final Report & Paper	Submisssion of Chapter 5	Submisssion of Chapter 4	Submisssion of FYP revised structure
Start Date	31/5/2025	15/5/2025	1/4/2025	17/3/2025
Duration (Days)	23	16	44	15

Figure 1.1: Gantt Chart for the System Development Timeline FYP 1



	Submission of FYP 2 Final Report & Paper	Submisssion of Chapter 5	Submisssion of Chapter 4	Submisssion of FYP revised structure
Start Date	31/5/2025	15/5/2025	1/4/2025	17/3/2025
Duration (Days)	23	16	44	15

Figure 1.2: Gantt Chart for the System Development Timeline FYP 2

1.7 Expected Outcome

i) A functional prototype demonstrating the system's flood detection, barrier management, and real-time monitoring capabilities.

ii) A user-friendly website that displays real-time tunnel parameters such as water levels, barrier activation status, and notifications for authorities.

iii) A physical model that illustrates the flood detection system's real-time operation and effectiveness, allowing for a clear display of its components and actions.

iv) Tunnel infrastructure safety and resilience have been increased by automated flood management, as well as faster emergency service response times.

1.8 Summary

This chapter presents the creation of a Tunnel Flood Detection System with a Barrier Control and Notification Mechanism to solve the growing problem of tunnel flooding in Malaysia, especially during intense monsoon rains. Since underground tunnels are particularly vulnerable to water buildup, flooding seriously threatens infrastructure and public safety. Existing flood control measures are insufficient in their reach and do not safeguard additional susceptible tunnels. Through a web-based interface, this concept uses ultrasonic sensors to measure the water level in real time, activate barriers automatically, and instantly notify authorities. A physical prototype and traffic light guidance system will be developed to demonstrate functionality, enhance tunnel safety, improve emergency response times, and contribute to Malaysia's efforts in mitigating climate-induced flood risks.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

In the literature review, we will examine the background of existing flood detection systems, their features, and their advantages and disadvantages to enable a comparison with the proposed system. This evaluation is vital for identifying features and technologies that could be integrated into the proposed tunnel flood monitoring and alert system. Understanding the strengths and limitations of current systems allows for informed decisions to enhance the efficiency and effectiveness of the new system. Additionally, by fixing flaws found in current systems, we hope to steer clear of similar problems and produce a stronger solution.

To conduct a thorough analysis, this review will consult several sources, such as journal articles, conference papers, and system case studies. Key references include IoT-based flood monitoring systems, Arduino-driven flood detectors, and commercial flood alert mechanisms, which will help differentiate the proposed system and guide its development.

2.2 Review on Similar Existing Systems

2.2.1 Flood Monitoring and Alerting System

The "Flood Monitoring and Alerting System using LoRaWAN Technology" is an innovative solution aimed at mitigating the impact of flooding through real-time monitoring and early warning mechanisms. The system employs sensors to measure water levels and rainfall intensity, transmitting data via LoRaWAN, a long-range, low-power wireless communication protocol. This data is sent to a cloud-based platform, where it is analyzed to generate alerts and notifications for authorities and residents.

Key features include scalability, cost-effectiveness, and real-time data transmission, making it suitable for urban and rural areas prone to flooding. By integrating IoT devices, cloud computing, and efficient communication technology, this system demonstrates a robust approach to enhancing flood resilience and preparedness.

The project's hardware design integrates various components to establish a robust system for monitoring and managing water flow, environmental conditions, and communication. As illustrated in Figure 1, the working model features an Arduino UNO as the primary control unit, a Node MCU for wireless communication, and sensors such as a water flow sensor, an ultrasonic sensor, and a DHT11 sensor for precise data collection. Additionally, the inclusion of a LoRa module provides reliable long-range connectivity.

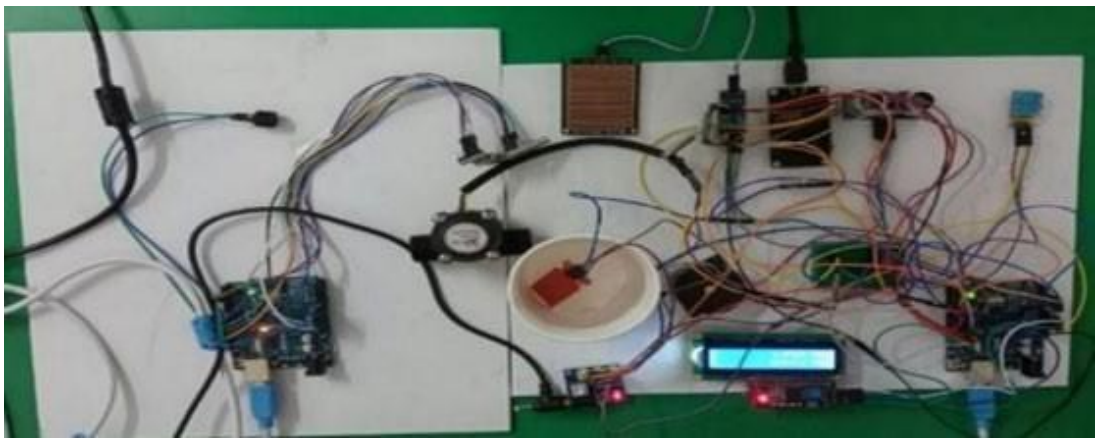


Figure 2.1: Working Model (Dhebe et al., 2023)

The real-time sensor data is displayed on the ThingSpeak web application, clearly indicating the absence of any flood conditions. ThingSpeak serves as a platform for visualizing and analyzing sensor data in real-time, allowing users to monitor and track various parameters. Figure 2 presents the real-time data on the web app, confirming that the current water level readings do not point to any potential flooding.

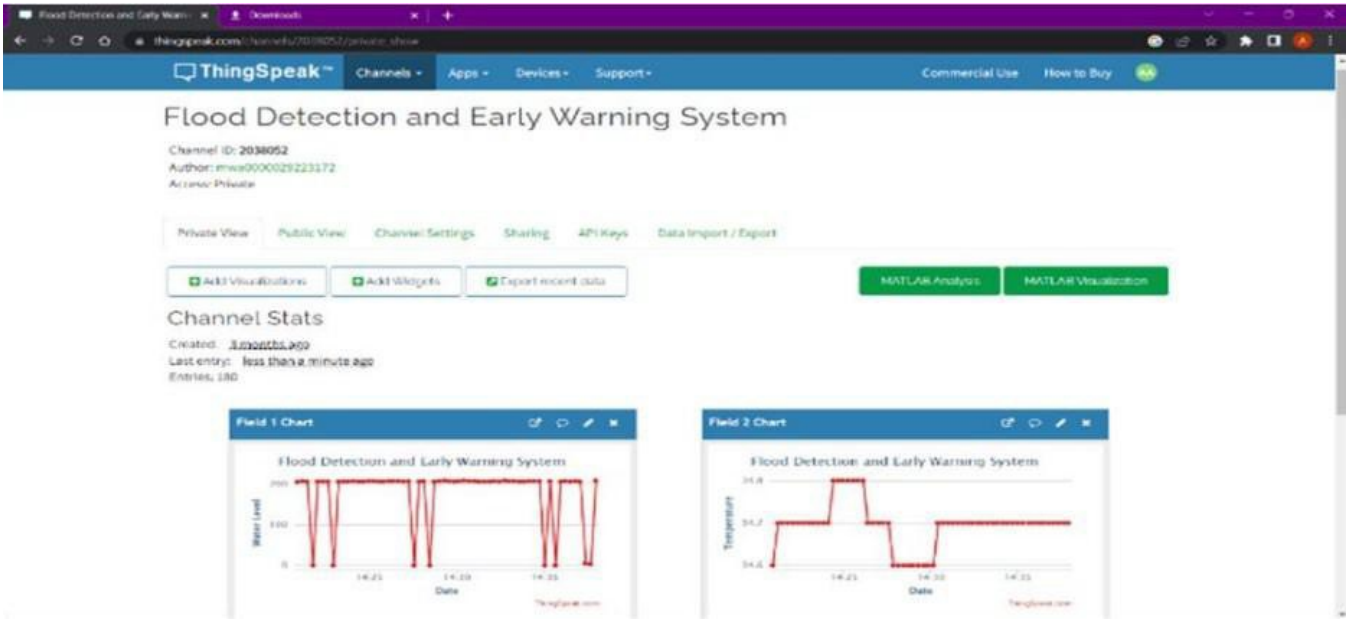


Figure 2.2: Realtime data showing on ThingSpeak Web App (Dhebe et al., 2023)

Starting with sensor initialization and data collection, the system workflow records variables such as weather, rainfall intensity, and water level. The Node MCU and LoRaWAN network are used to send the data to a base station or gateway for processing and analysis. Process restarts if transmission fails. In addition to being shown in real time on the Virtuino 6 app, processed data is transferred to the ThingSpeak cloud platform for management and storage. This app's user -friendly interface allows it to visualize important metrics like humidity, water level, and temperature. When thresholds are crossed, such as dangerous water levels, alerts are set off, but safe conditions are shown graphically without any further information. The process guarantees effective data processing, allowing users to keep an eye on environmental conditions and make defensible decisions to reduce risks.

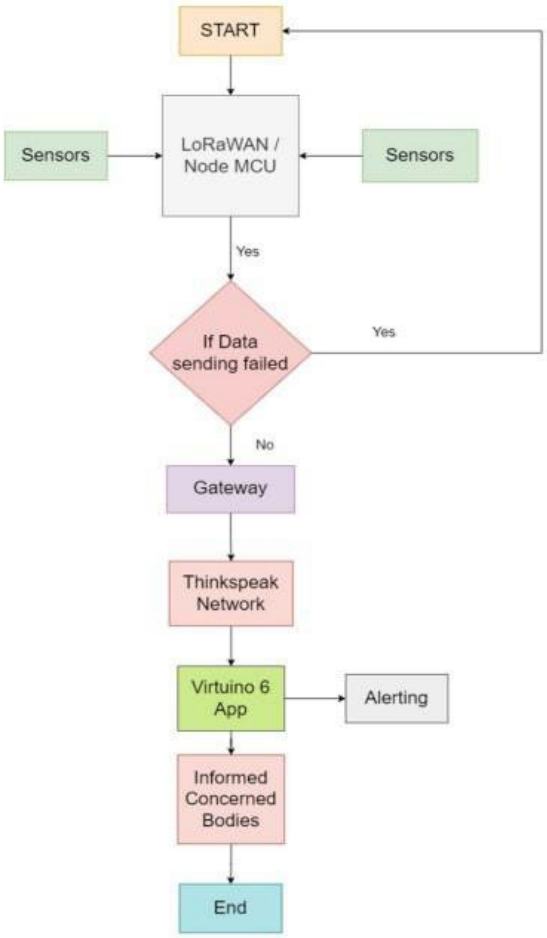


Figure 2.3: Flowchart of the system (Dhebe et al., 2023)

2.2.2 Automated flood water level sensor and alarm system using Arduino Uno

This system is a cost-effective and useful way to improve flood monitoring and early warning systems. The system detects rising water levels and sets off alarms to provide timely alerts by using Arduino Uno microcontrollers in conjunction with ultrasonic and capacitive water level sensors. It gives communities the ability to react swiftly and lessen the effects of floods by providing real-time flood data.

The system, which was created to address the Philippines frequent flood problems, has proven successful in keeping an eye on rivers, urban drainage systems, and other high-risk locations. Customization and scalability are made possible by its open-source and user-friendly design, and features like buzzers and LED indicators enhance alert capabilities, guaranteeing operation even in remote areas. The system is ideal for widespread use in areas that are prone to flooding because it is small, inexpensive, and simple to install. Its potential IoT integration and compatibility with data visualization tools also improve resilience and preparedness for disasters. See the study that was published in the International Journal of Research Studies in Educational Technology for more information.

Figure 2.4: Actual Product of Automated flood water level sensor and alarm system using Arduino Uno (Magnaye et al., 2024)

2.2.3 Iot Based Early Flood Detection System with Arduino and Ultrasonic Sensors in Flood-Prone Areas.

This system is intended to solve the recurring problem of flooding, which is a serious risk to property, public safety, and infrastructure. This system uses Arduino microcontrollers and Internet of Things technology to provide a dependable, real-time flood monitoring and warning system. To detect rising water levels, water level sensors are placed throughout the area. They send vital data to a centralized platform for analysis and alert distribution to authorities and residents.

This system is specially designed for areas that are vulnerable to flooding, allowing for early detection and response to reduce damage and save lives. The systems use of IoT for connectivity guarantee smooth data transfer to cloud-based platforms, enabling users to keep an eye on conditions via mobile applications or web interfaces. Because of its modular design, it is very scalable and adaptable to the requirements of residential areas, rivers, and urban drainage systems.

The IoT-based system is a viable and sustainable flood risk management solution because of its emphasis on affordability and deployment simplicity. By offering an effective, automated method of flood detection and early warning, this project seeks to increase community resilience and support improved disaster preparedness and mitigation initiatives.

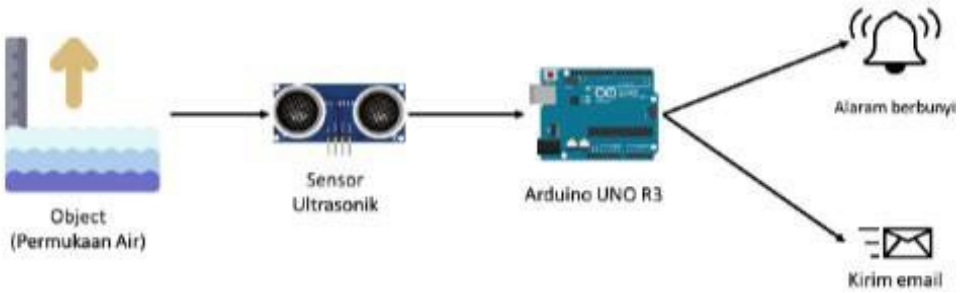


Figure 2.5: Application block diagram (Darwis et al., 2023)

The distance between the ultrasonic sensor and the water surface is measured by the Arduino-connected sensor, as seen in Figure 5. It sends an email with the water level status and sounds like the buzzer when the water level rises and gets close to the sensor, indicating that it is at or below 10.

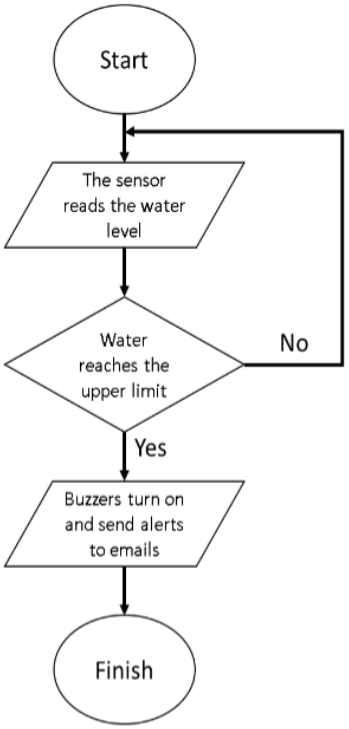


Figure 2.6: Application Flowchart (Darwis et al., 2023)

The procedure that users go through when interacting with the IoT-based flood early warning application is depicted in the flowchart in Figure 6. The device is placed in flood-prone areas, especially in places like rivers or ditches where water first flows.

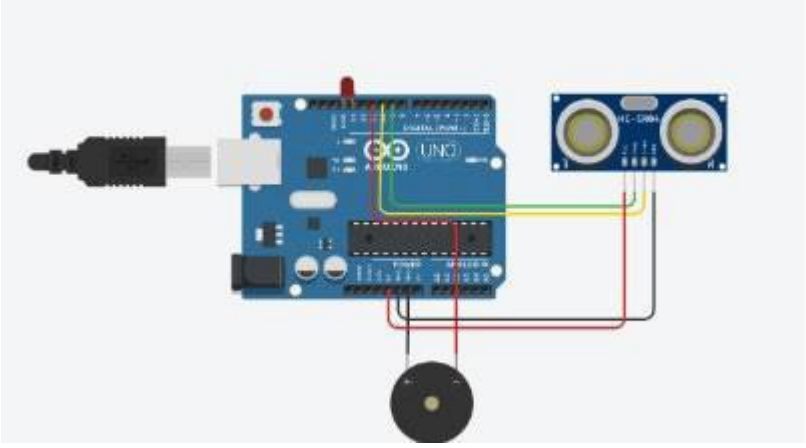


Figure 2.7: Schematic circuit of device and component (Darwis et al., 2023)

Figure 2.8: Display of water level notification via email

2.3 Table of Comparison between Projects Systems.

Table 2.1: The comparison of functionalities between similar existing systems and the proposed project

Functionality	Project 1: Flood Monitoring and Alerting System	Project 2: Automated flood water level sensor and alarm system using Arduino Uno	Project 3: IoT-Based Early Flood Detection System with Arduino and Ultrasonic Sensors in Flood-Prone Areas.	Proposed Project
Real-Time Monitoring	✓	✓	✓	✓
Barrier Control Mechanism	✗	✗	✓	✓
Notification System	✓	✓	✓	✓
Web Interface	✗	✓	✗	✓
Traffic Management	✗	✗	✗	✓
Physical Model	✓	✗	✓	✓
Targeted Area	✓	✓	✓	✓

The comparison between the existing systems and the proposed Tunnel Flood Detection System with Barrier Control and Notification Mechanism reveals significant advancements in functionality and design. All systems share the essential feature of real-time monitoring, utilizing sensors to detect rising water levels, ensuring timely flood detection. However, the proposed system extends beyond basic monitoring by introducing several critical innovations that address specific challenges in tunnel flood management.

One of the most notable differences is the inclusion of an automated barrier control mechanism in the proposed system. While the existing systems monitor water levels, they cannot actively restrict access to dangerous areas during flooding. The proposed system mitigates this gap by employing automated barriers that activate when water levels exceed a predetermined threshold, effectively preventing vehicles and pedestrians from entering flooded tunnels and reducing the risk of accidents.

Additionally, although all systems provide some form of notification to authorities, the proposed system enhances this functionality with a comprehensive web-based interface. Unlike the existing systems, which rely on basic notifications through SMS or emails, the proposed solution offers a real-time, user-friendly platform for monitoring water levels, barrier status, and alerts. This interface improves situational awareness and allows for coordinated emergency responses by providing actionable insights directly to authorities.

The proposed system also incorporates a traffic management feature, a critical addition that the other systems lack. With a traffic light control mechanism, the system can signal the accessibility of tunnels to drivers, further enhancing public safety during flood events. Moreover, the inclusion of a physical model in the proposed system demonstrates a practical approach to design validation and testing. This model allows for simulated flood conditions, enabling thorough evaluation and refinement of the systems functionalities.

Lastly, while the existing systems target general flood-prone areas, the proposed system is specifically tailored to tunnels, addressing their unique vulnerability to water accumulation during flash floods. By focusing on this specific infrastructure, the proposed system provides a dedicated solution to an overlooked yet critical safety issue, particularly in urban areas prone to severe monsoons. These combined features make the proposed Tunnel Flood Detection System a comprehensive, innovative, and effective solution for modern flood management challenges.

2.4 Review on Tools and Technology

2.4.1 Hardware Component

i) Ultrasonic Sensors

The main piece of hardware used to measure water levels is an ultrasonic sensor. By sending out sound waves and timing the return of the echo, these sensors determine the distance to the water's surface. They are perfect for tunnel flood monitoring because of their dependability in challenging environmental circumstances.

ii) The Arduino microcontroller

The Arduino acts as the system's central processing unit, gathering sensor data and carrying out preprogrammed actions. It interprets sensor data to operate traffic lights, barriers, and alerts.

iii) Modules for connectivity and power supply

The system includes a reliable power source and communication module like Wi-Fi to guarantee continuous operation. These modules are essential for keeping the system operating and sending alerts.

2.4.2 Software Component

i) C/C++ Programming for Arduino

C/C++ programming is used to process sensor data, manage hardware, and carry out responses on the Arduino microcontroller. The code specifies the water level thresholds as well as the logic for managing traffic lights, activating barriers, and setting alerts.

ii) Development of Web Interfaces

The real-time web-based interface is a significant software innovation in the suggested system. The interface, which was created with HTML, CSS, and JavaScript, provides important information in an easy-to-use format, including traffic light signals, barrier statuses, and water levels. Authorities can keep an eye on the system from a distance and react quickly.

iii) Mechanism of Alerting

Web-based alerts are used by the system to alert the appropriate authorities. Using quicker and more flexible communication channels, this feature takes the place of conventional SMS or email notifications found in current systems.

2.5 Summary

In conclusion, this chapter examines current flood detection systems, evaluating their advantages, disadvantages, and features to guide the creation of a suggested tunnel flood detection system. Important systems include Arduino-powered solutions, LoRaWAN-enabled alert systems, and Internet of Things-based flood monitoring. The suggested system offers novel features like automated barrier control, traffic management, and a web-based interface designed specifically for tunnels, even though these systems are excellent at real-time monitoring and alerting. The suggested solution seeks to improve safety, effectiveness, and preparation for disasters by tackling the difficulties associated with tunnel flooding.

CHAPTER 3: REQUIREMENT ANALYSIS & DESIGN

3.1 Introduction

This chapter describes the methodological approach pursued in the development of the Tunnel Flood Detection System with Barrier Control and Notification Mechanism. The aim is to devise a system that is reliable, efficient, and can easily be scaled up to meet the peculiar challenges of flood management in tunnels. The methodology will be segregated into clearly distinguishable phases, each with specific objectives ranging from the collection of requirements of the system. The structured approach will provide an all-around systematic development that meets the objectives of the project while delivering a functional prototype.

3.2 Project Methodology

The SDLC framework, which offers an organized process to efficiently manage the systems development, is used in the project methodology. To guarantee a complete comprehension of the phases involved with the systems development, each SDLC phase is explained in detail. This project will yield a resilient, dependable, and scalable flood detection system by following SDLC. This chapter discusses SDLC phases in detail, namely: Planning and Analysis, Design, Development, Testing and Maintenance and how each phase relates to the project. The following Figure 3.1 illustrates the five primary stages of the System Development Life Creator (SDLC) :

Figure 3.1: System Development Life Cycle (SDLC) (Sinha & Sinha, 2021)

3.2.1 Planning and Analysis

3.2.1.1 Planning

The project's planning phase is its foundation, defining its goals, parameters, resources, and viability. To improve public safety, the main goal is to create a real-time flood detection system for tunnels that incorporates automated barrier control, a notification system, and a web-based interface for real-time monitoring and alerts. The scope includes using ultrasonic sensors to monitor tunnel water levels, automatically activating barriers during flooding, notifying authorities, and developing a physical prototype and web-based dashboard. Resource planning determines the necessary hardware and software, making sure that everything is in line before moving on to later stages.

3.2.1.2 Analysis

The analysis stage holds a thorough assessment of the requirements relevant to the suggested system, together with the assessment of current flood detection methodologies in the literature review conducted in Chapter 2. Such a phase guarantees that a proper comparison of the existing and proposed alternative systems will be made while highlighting the distinct requirements needed in the development process.

3.2.1.3 Analysis of current methods

The contemporary flood monitoring and detection systems, which were discussed in Chapter 2, are realized with the support of various technologies, Arduino-driven solutions, and IoT platforms. These technologies efficiently carry out their designed functions: observing the water level and warning users or authorities in charge. Still, they fall short of offering such advanced features as automated barrier control and traffic

management with full real-time visualization interfaces. Most existing systems are limited to the functions of monitoring and notification without addressing safety measures such as access restriction to hazardous areas during floods. Other constraints of these systems include dependence on specific communication technologies, limited scalability, and high maintenance requirements.

3.2.1.4 Analysis of Proposed System

The proposed Tunnel Flood Detection System overcomes the deficiencies of existing methods by incorporating additional features. It has automated barrier control systems that restrict access to tunnels in cases of flooding, ensuring the safety of the public. It also features a traffic management system with traffic light indicators that greatly enhance functionality by indicating the accessibility of the tunnels to motorists. The system further provides a simple web interface allowing real-time data visualization and alert monitoring. By using IoT technology, the system provides efficient data transmission and scalability in deployment for many tunnel locations. The use of low-cost and long-life components makes it economically feasible for long-term operation under different environmental conditions.

3.2.1.5 Software Requirements

The Arduino IDE is an open-source platform that provides a complete development environment for writing, compiling, and uploading code to the Arduino microcontroller. It supports programming in C/C++; hence, it is very good at developing hardware projects based on Arduino. Moreover, the IDE has a serial monitor that allows real-time testing and debugging of sensor data to ensure proper communication between the hardware and software parts.

The C/C++ programming language forms the basis of the Arduino framework, which offers strength and efficiency in defining the operational parameters of the hardware elements. This includes the monitoring of flood detection sensors, the triggering of the buzzer, and the transmission of data to the database. With the help of libraries like `Wire.h` and `WiFi.h`, advanced features like real-time sensor readings, notifications, and seamless data transmission can be achieved.

The front-end of the flood detection system is built using HTML, CSS, and JavaScript. HTML structures the content of the website, CSS styles it for a user-friendly appearance, while JavaScript adds dynamic interactivity. Technologies that allow the website to show live sensor data and update water level statuses in real time make it easy for users to access and interpret critical information related to floods through an intuitive interface.

For the backend, a server-side language like PHP has to be in place. They perform tasks involving setting up connections to a database, processing real-time data from sensors, and enabling communication between the hardware system and the website. This framework ensures effective management of the flood alert and notifications system, hence increasing responsiveness and reliability. To host the website, eXAMPP is a local server package that makes website testing on a personal computer much more convenient by bundling essential

tools like Apache, MySQL, and PHP. On the other hand, Firebase Hosting is an online platform that is scalable, fast, and secure, which makes it especially good for deploying real-time applications.

Finally, Visual Studio Code serves as the main development environment for the project. This powerful text editor, with its support for different programming languages and extensions, is suitable for both Arduino programming and front-end and back-end web development. The debugging features, integration of version control, and support for multiple languages in Visual Studio Code ensure a productive and development experience.

3.2.1.6 Hardware Requirements

The flood detection system will include a few hardware components to ensure the system works well and simulates real-world applications. At the core of the system is the Arduino Uno, the microcontroller that acts as the brain or the main processing unit of the entire system. This will read data from sensors, control the operation of actuators like barriers and buzzers, and communicate with other parts of the system. The Arduino Uno is very versatile and reliable for prototyping and real-time applications.

The system will also use ultrasonic sensors for water-level detection. It will emit sound waves and, based on the time of the echo return, give real-time data required for flood detection. The Node MCU, a Wi-Fi-enabled microcontroller, will make the system capable of sending the data wirelessly to an IoT platform and a website to disseminate real-time updated information and manage water levels along with flood alerts remotely. These modules will allow the system to scale and adapt to regions without direct Wi-Fi connectivity by enabling long-distance communication.

Servo motors will be used for automated barrier control, opening and closing barriers according to the water level readings. The precise nature of these motors ensures a correct response to changed flood conditions. A traffic light system with LEDs will indicate the accessibility of the tunnel. An auditory alert via a buzzer will also warn when the water level surpasses the safe limit, ensuring immediate attention even if visual indicators are not observed.

The system will be powered by 9V batteries to assure a stable supply of power to the Arduino and related components. This will ensure reliable operation when performing test procedures or in cases of power disruptions. A voltage regulator will also be included, ensuring stable power for the sensible components and being protected from potential damage due to fluctuations in the power supply.

The project requires a model tunnel structure to be made from cardboard, acrylic, or PVC to simulate real-world conditions. The water reservoir will then be placed below the model. It is in this structure that rising levels of water can be replicated so that the system under scrutiny performs well.

Jumper wires are used to allow the assembly and interconnection of the components by providing reliable electrical connections among sensors and Arduino, by using a breadboard. These two platforms are essential in keeping the circuit well-organized and less complex during prototyping. Finally, mounting components such as adhesive, screws, or tape will be attached to securely fasten the sensors, traffic lights, and barriers to the model for stability and durability purposes during testing and demonstration.

3.2.3 Design

The design stage is where the specifications and analytical findings are transformed into a detailed blueprint of the Tunnel Flood Detection System. It includes developing the architecture of the system and integrating various components. Software design involves programming Arduino to collect data, initiate barriers, and send notifications, while hardware design involves the wiring of sensors with communication modules and control systems. A flowchart outlines the workflow, while a context diagram focuses on the interactions between system components and external entities, thus ensuring adherence to the intended functionality.

3.2.3.1 System Architecture

System architecture relates to the conceptual design and structure of a system, outlining its individual components and their interrelations. It incorporates the hardware, software, and network components of the system, along with the methods of communication and procedures for exchanging information between them. Additionally, system architecture represents a reference schema for the composition of several subsystems, modules, and services with the ultimate view of achieving a desired functionality. Figure 3.2 shows the flood detection system architecture.

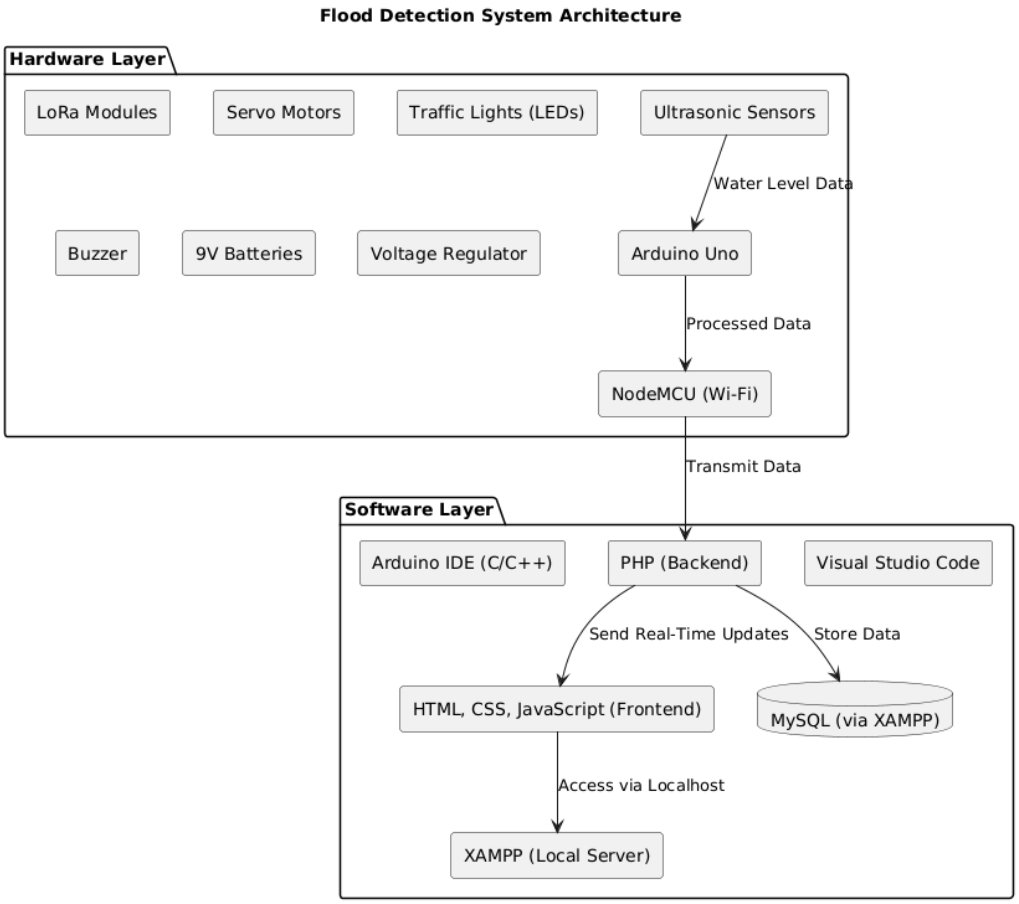


Figure 3.2: System architecture of the proposed system

3.2.3.2 Flowchart

A flowchart is a graphical tool used to represent the step-by-step development of processes in a specific system. It involves different symbols to represent actions, decisions, and the overall sequence of operations. By providing a graphical view of the process, a flowchart makes it easier to understand the workflow of the system and highlights important decision points and activities. The flowchart of the proposed system is shown below in Figure 3.3.

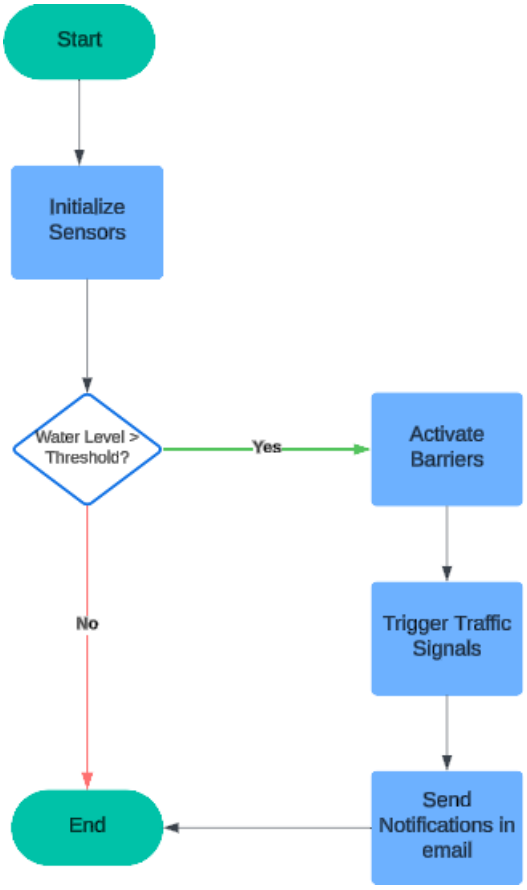


Figure 3.3: Flowchart of the system

3.2.3.3 Context diagram

A context diagram is a kind of visual representation that shows how a system interacts with external entities. It provides an overview of the main components of the system and their interrelations, showing the flow of information between the system and its users or related devices. This enables one to understand better the boundaries of the system and its core functions regarding external inputs and outputs. The context diagram is designed in Figure 3.4.

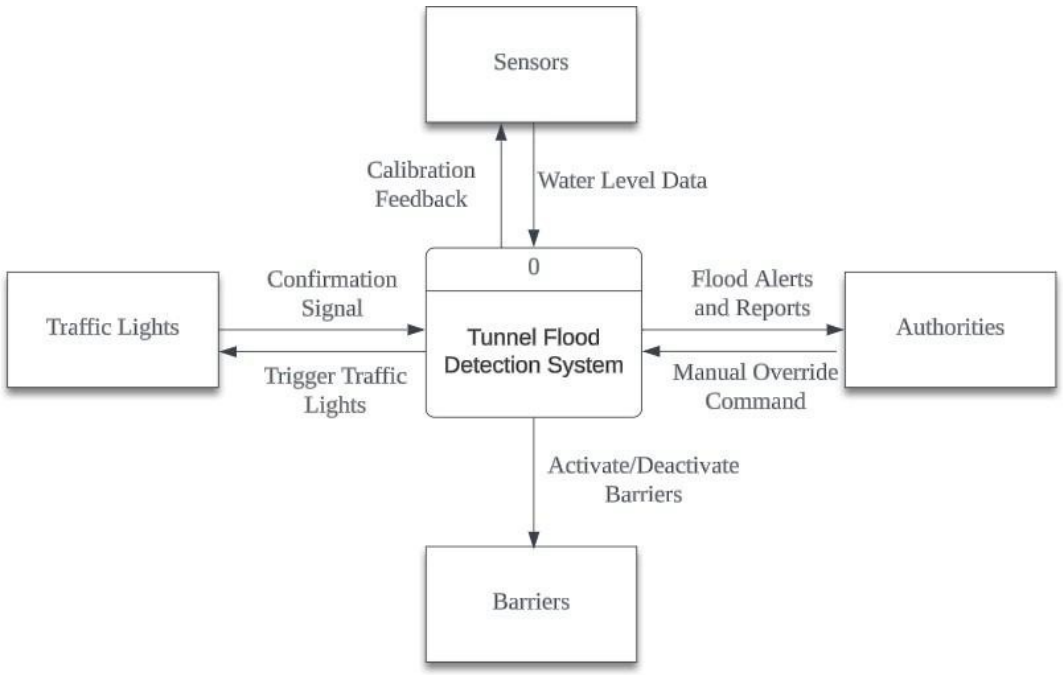


Figure 3.4: Context Diagram of the system

3.2.3.4 DFD level O

DFD Level 0 is a general view of the system, representing the main processes, external entities, data stores, and data flows within the system. This level stresses how information is received and disseminated in the system by encapsulating core functionality without delving into the detailed intricacies of processes; hence, this is an elementary and high-level presentation of the system. Figure 3.5 displays the Data Flow Diagram Level 0.

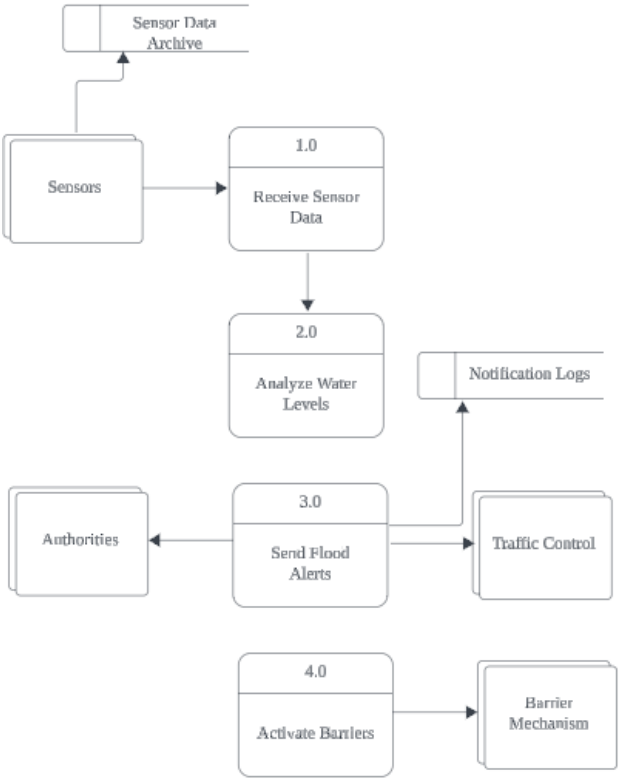


Figure 3.5: DFD level 0 of the system

3.2.3.5 Entity Relationship Diagram (ERD)

An Entity-Relationship Diagram (ERD) is a graphical representation of the data model of a system. It is a representation of the entities, objects, people, or ideas that constitute the system, their attributes, and the relationships between them. An ERD is very instrumental in understanding the structure of a database for a system and plays a very critical role in the design of a consistent database schema that supports efficient data storage and retrieval. The following diagram shows the data interconnectivity of the system in a pictorial form. The ERD of the proposed system is showed in Figure 3.6.

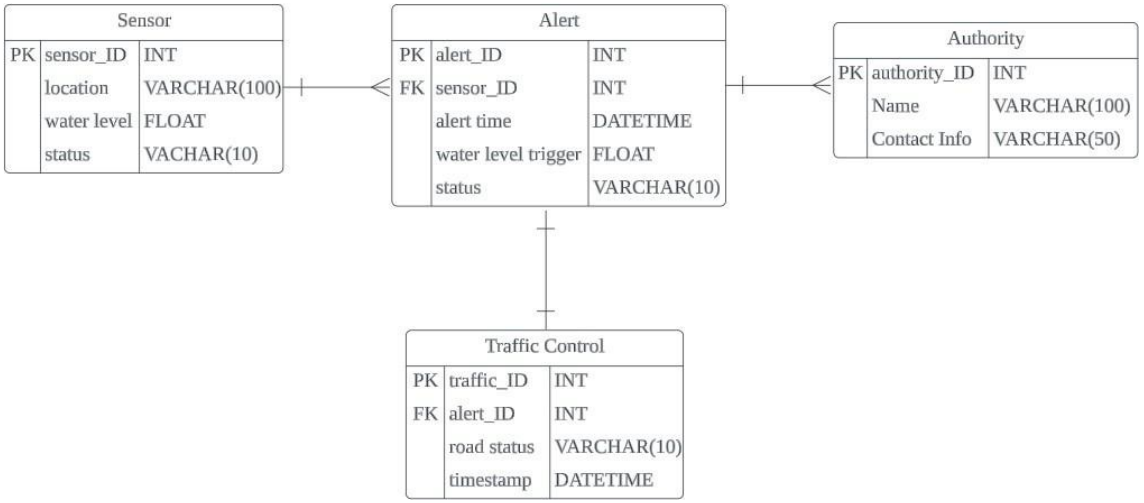


Figure 3.6: Entity Relationship Diagram of the system

3.2.3.6 Data Dictionary

A Data Dictionary serves as a centralized storage that defines every data element in use within the system. It defines the definitions, data types, acceptable values, formats, constraints, and interrelationships between each data element. By maintaining a centralized data dictionary, the system enforces consistency, accuracy, and standardization in data application throughout the framework. It becomes an easy reference for users to understand how data is structured and what its purpose is. The data dictionary for the proposed system is below:

Sensor Table

Table 3.1: Data dictionary for sensor

Field Name	Field Type	Constraints
Sensor_ID	INT	Primary key, auto increments, Not null
Location	VARCHAR (100)	Not null
Water_level	FLOAT	Not null
Status	VARCHAR (10)	Not null

Alert Table

Table 3.2: Data dictionary for alert

Field Name	Field Type	Constraints
Alert_ID	INT	Primary key, auto increments, Not null
Sensor_ID	INT	Foreign key references Sensor Not null
Alert_time	DATETIME	Not null
Water_level_trigger	FLOAT	Not null
Status	VARCHAR (10)	Not null

Authority Table

Table 3.3: Data dictionary for authority

Field Name	Field Type	Constraints
Authority_ID	INT	Primary key, auto increments, Not null
Name	VARCHAR (100)	Not null
Contact_info	VARCHAR (50)	Not null

Traffic Control Table

Table 3.4: Data dictionary for traffic control

Field Name	Field Type	Constraints
Traffic_ID	INT	Primary key, auto increments, Not null
Alert_ID	INT	Foreign key references Alert Not null
Road_status	VARCHAR (10)	Not null
Timestamp	DATETIME	Not null

3.2.4 Development Phase

The development phase involves translating the design specification into a working system. This involves hardware assembly, software development, and the integration of different components. Sensors are installed in a scaled model of a physical tunnel to represent real-world conditions, and all elements such as Arduino, Node MCU, barriers, and traffic lights are integrated and set up. The Arduino microcontroller is programmed to process sensor data and execute predetermined actions, including barrier actuation and notification sending. The web-based interface is deployed for real-time monitoring. In this integration phase, smooth communication among hardware and software components is ensured to ensure that a complete system is ready for testing and deployment.

3.2.5 Testing

The testing phase is very important because it allows for the determination of whether the system behaves as anticipated under all possible scenarios. In this respect, unit testing of individual components such as sensors, barriers, and notification systems is carried out. Integration testing ensures proper interaction between hardware and software components, guaranteeing that data is freely circulated in the system. System testing evaluates the overall functionality by simulating different flood scenarios to test the responsiveness and reliability of real-time monitoring, barrier activation, and alert notifications. Validation testing ensures that the system meets all the functional and non-functional requirements specified during the analysis stage. This test process identifies any potential problems or limitations, which are then addressed before the final deployment.

3.2.6 Deploy and Maintenance

The maintenance phase guarantees the enduring functionality and reliability of the system following its deployment. Regular calibration of sensors is conducted to uphold accuracy, while the system performance undergoes continuous monitoring to promptly identify and resolve any emerging issues. Hardware components may be upgraded as necessary to improve the functionality or scalability of the system. Additionally, user support and training are offered to ensure that administrators can operate the system effectively. Maintenance activities include updating software regularly, tuning for performance and adding new functions to ensure the system continues to maintain its resilience and agility in meeting the dynamically changing demands.

3.3 Summary

This chapter clarifies the development of the system by applying the SDLC methodology, highlighting the design of a reliable, scalable, and efficient solution to control flooding in tunnels. The proposed system faces the shortcomings of existing methods by including features like real-time flood detection, automated barrier control, IoT-based notifications, and an interactive web interface. The hardware components, such as Arduino and ultrasonic sensors, are designed with the help of software applications in data processing and visualization to achieve uninterrupted functionality. This chapter also focuses on using flowcharts and context diagrams to define workflows and interactions for better transparency in system design. Much attention is given to testing and maintenance to ensure long-lasting reliability and adaptability to real-life situations.

Chapter 4: Development and Implementation

4.1 Introduction

This chapter describes how the Tunnel Flood Detection System with Barrier Control and Notification Mechanism was developed and implemented. It provides in-depth descriptions of how the hardware and software components were integrated. The chapter elaborates on the real-world implementation according to the system design, the establishment of the prototype model, and the operational capacities of separate modules, ensuring that the system meets the requirements of immediate flood detection, autonomous barrier control, and notification sending.

4.2 System architecture implementation

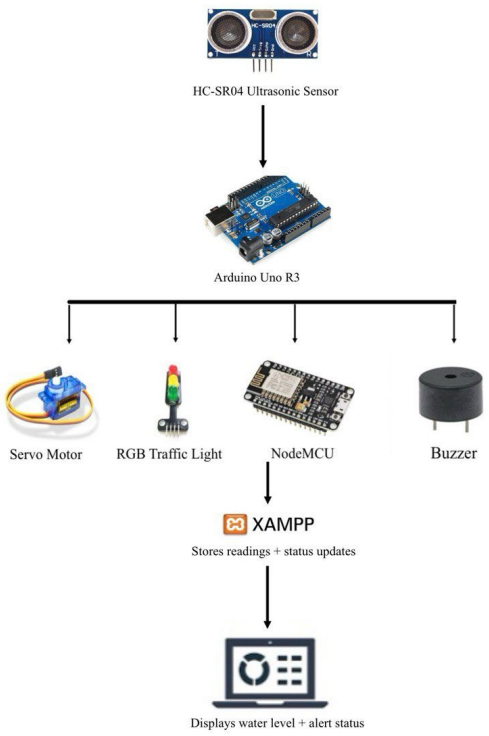


Figure 4.1 Implemented System Architecture

System architecture has key components to detect tunnel flooding, activate response, and undertake real-time monitoring. Foremost is the Ultrasonic Sensor, which tracks water levels in the tunnel and sends the information to the Arduino Uno R3, the main controller. The Arduino uses the water level to trigger different components. It turns on the Servo Motor to open or close the barrier. It uses Traffic Light LEDs to signal the status of the tunnel. It uses a Buzzer for audible alerts. NodeMCU receives data regarding water level and sends email alerts to the authorities.

At the same time, data is transmitted to the NodeMCU ESP8266, which is Wi-Fi capable and sends data to a Web Server and Database. Data is visualized on a Real-Time Web Dashboard so that users and authorities can view tunnel status remotely. The whole system runs on a USB connection, with the help of a voltage regulator, so that all components can run safely and uninterruptedly.

4.3 Hardware Implementation

Various hardware components were assembled and integrated into a functional model to develop the Tunnel Flood Detection System with Barrier Control and Notification Mechanism. The model was constructed as a miniature town that includes a tunnel, a road network, and traffic lights to simulate real-life flood situations in a city. The hardware setup was crafted to illustrate how the system responds to rising water levels by triggering responses like barrier closure and notification sending and presenting real-time data to users.

The following is a list of hardware elements employed:

- Arduino Uno R3
- NodeMCU ESP8266 (Wi-Fi Microcontroller)
- Ultrasonic Sensor (HC-SR04)
- Servo Motor (Barrier Control)
- Traffic Light System (LEDs: Red, Yellow, Green)
- Buzzer (Audio Alert)
- Voltage Regulator
- USB Power Supply
- Breadboard and Jumper Wires
- Miniature Model

The automation flood monitoring, detection, and alerting system was accomplished by piecing together wiring for each component to build the hardware circuitry. The sonar HC-SR04 sensor was interfaced with an Arduino Uno R3 microcontroller which acted as the

primary control unit of the system. Based on the sensor measurements, an attempt on closing the barrier gate using a servo motor at critical water levels above a preset borderline. For surpassing threshold levels, a traffic light system of red, yellow, and green LEDs was also mounted on an Arduino, together with a tunnel visual indication for vehicle access. In conditions of risk, a buzzer was added as an audible alarm. For sensor reading, the NodeMCU was powered separately with its battery, whilst having the Arduino UNO together with the NodeMCU powered by a USB cable. The NodeMCU oversaw connecting to Wi-Fi and sending HTTP post requests to online web servers. To maintain voltage between each of the modules, a voltage regulator needed to be present.

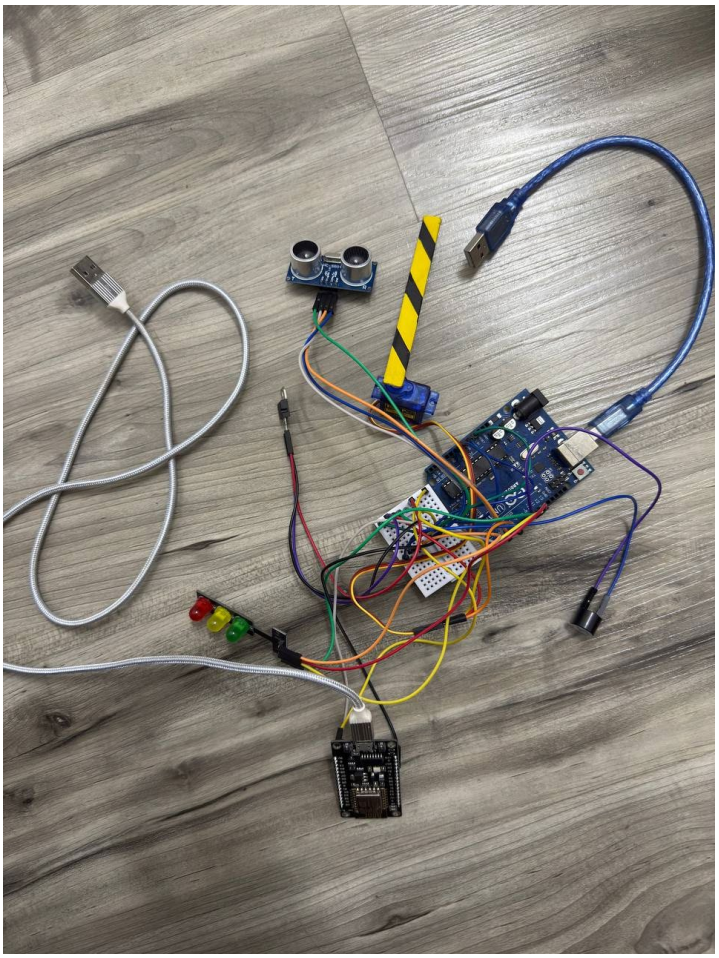


Figure 4.2 Complete circuit wiring setup

The integrated circuit was incorporated in the miniature model town to showcase how a realistic system would work. The model also had the cybernetic features of a real city, such as a tunnel, barrier gate, a road system, traffic lights, and buildings. An ultrasonic sensor was located at the entrance of the tunnel to determine the changing water levels, and a servo-operated barrier blocked access wherever a flood was found. Traffic lights were also placed at the side of the roads and the entrance of the tunnel for safe entrance indication. The buzzer acted as a primary notice for emergency response. This configuration enabled users to witness how the system identifies flood detection, initiates actions, and send alerts on its own within a powerful automated workflow.

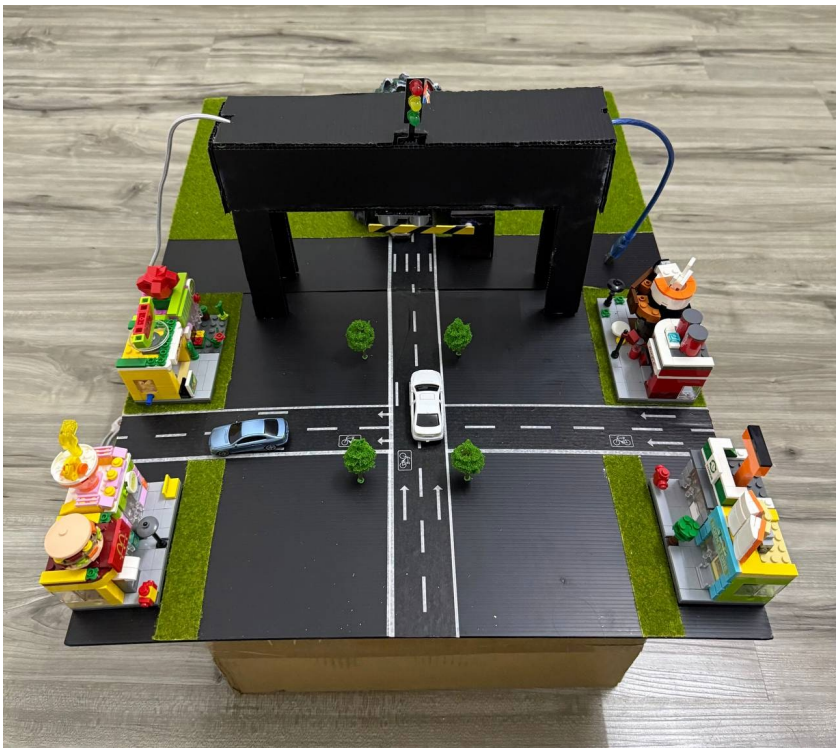


Figure 4.3 Miniature town model

4.4 Software Implementation

This part outlines the installation and configuration of the software instruments necessary to build and maintain the Tunnel Flood Detection System. This encompasses the installation of Arduino IDE along with the NodeMCU board, XAMPP server and phpMyAdmin, database, a serial monitor for debugging, and the website designed with HTML, CSS, and JavaScript.

4.4.1 Arduino IDE Installation

The Arduino IDE was utilized to write and load code to the Arduino Uno board. It was obtained from the official Arduino website (https://www.arduino.cc/en/software), which is depicted in Figure 4.4.

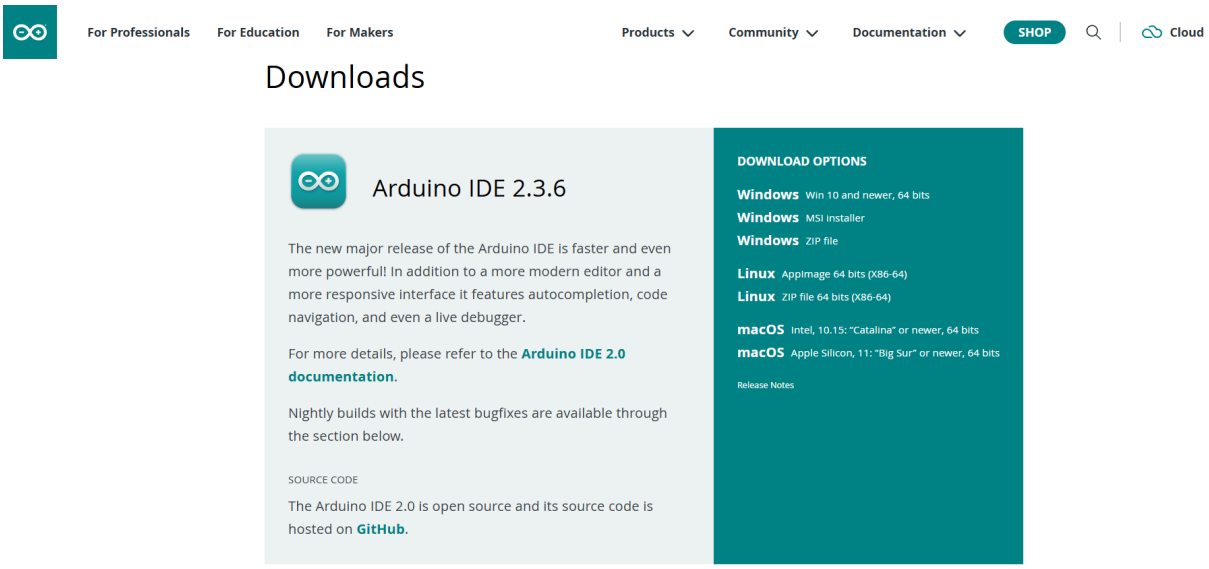


Figure 4.4 Arduino IDE download page

The software is then run after it is installed, and then a USB cable is used to connect the Arduino Uno board with a computer. Following this, the board that has been connected is selected manually by navigating to Tools → Board and choosing Arduino Uno. A corresponding COM port was chosen from Tools → Port in order to facilitate the communication. This configuration was the reason why the microcontroller was programmed directly from the IDE.

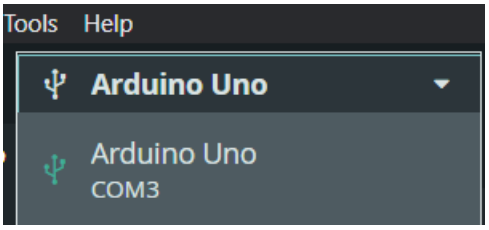


Figure 4.5 Selecting Arduino Uno COM port

4.4.2 NodeMCU Board Setup

Initially, not being in the Arduino IDE by default, NodeMCU (ESP8266) had to be put into the board manager manually. The link http://arduino.esp8266.com/stable/package_esp8266com_index.json was taken from there to the Additional Board Manager URLs field under File → Preferences. Then the board package was installed from the Board Manager via Tools → Board by searching for "ESP8266" and clicking install. Following the installation, NodeMCU board was chosen as NodeMCU 1.0 (ESP-12E Module) from the board list. This setup allowed NodeMCU to be the one programmed through the Arduino IDE for sending the sensor data to the web server.

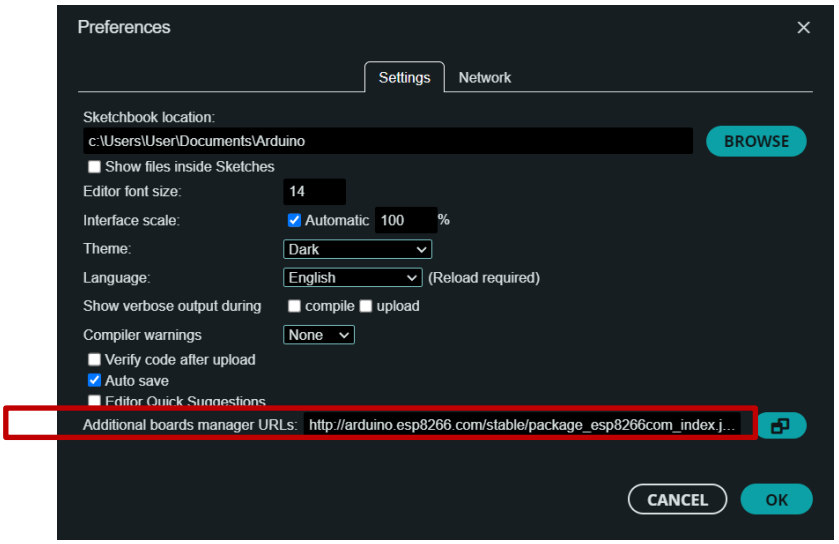


Figure 4.6 Adding ESP8266 board URL

Figure 4.7 Selecting NodeMCU board

4.4.3 XAMPP Installation and Configuration

XAMPP has been used for the local server environment to run the server-side PHP scripts and also to manage the database. The application was installed by getting it from https://www.apachefriends.org/index.html wherein Figure 4.8 illustrates it.

Figure 4.8 XAMPP download page

Once the installation was done, the XAMPP control panel was started and then the Apache and MySQL services were activated with clicking the Start buttons. After the two services had been turned on, the local server was accessible through the web address localhost in the browser while the database management interface (phpMyAdmin) could be further reached by going to the address localhost/phpmyadmin.

Figure 4.9 XAMPP control panel

4.4.4 Database Setup using phpMyAdmin

A fresh database named flood_system was created in phpMyAdmin to accommodate the information about flood detection. Therefore, in this database, a table called readings was created. The table has these fields: distance (storage of water level readings from the sensor), status (Safe, Warning, or Danger indication), and timestamp (that marks date and time of each reading). The effectiveness of this way of arranging the monitoring system made the data processing and retrieval easier. Figures 4.10 and 4.11 present the database view and table structure, respectively.

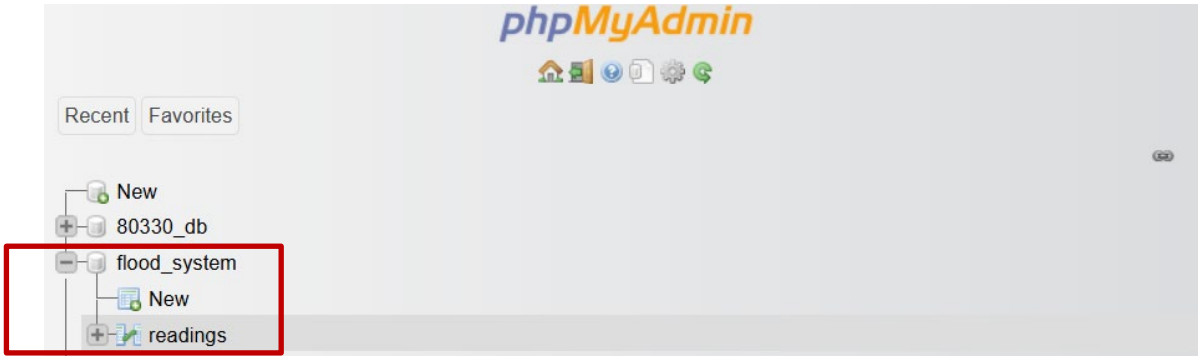


Figure 4.10 phpMyAdmin homepage

distance	timestamp	status
267	2025-04-27 04:59:53	open
267	2025-04-27 04:59:56	open
67	2025-04-27 04:59:57	open
7	2025-04-27 04:59:59	closed

Figure 4.11 Readings table structure

4.4.5 Serial Monitor Debugging

The Serial Monitor in the Arduino IDE was used for testing and debugging the system during development. After the code was uploaded to the NodeMCU, the Serial Monitor was opened by going to Tools → Serial Monitor. At the time the system was operational, the display exhibited the distance from the ultrasonic sensor, the IP address of the NodeMCU, the server response codes and received verification of the successful transmission of the data to the server. This action was necessary to confirm the right operation of the system before its application in real-time.

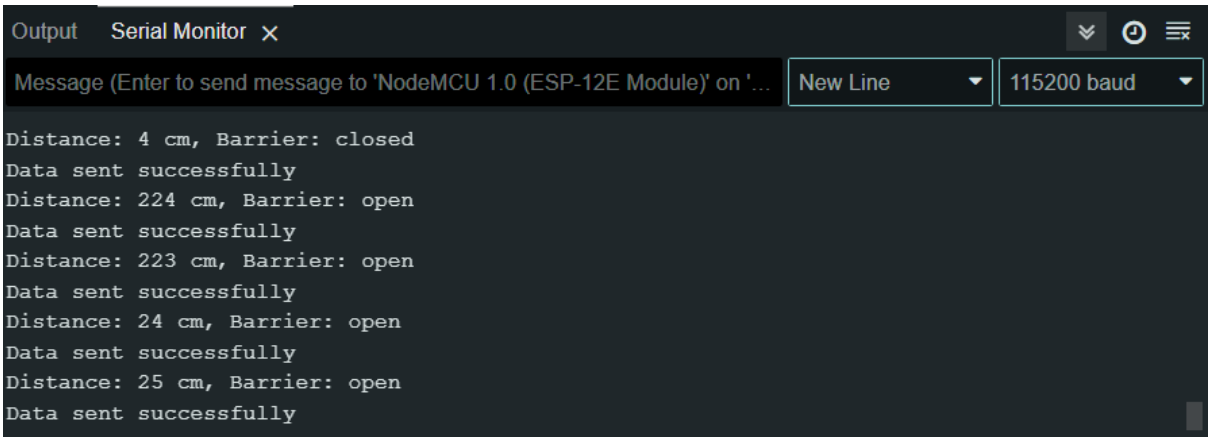


Figure 4.12 Serial motor readings

4.4.6 Website (HTML, CSS, JavaScript) Implementation

A monitor that shows only one local concurrent flooding situation from other areas in the Malaysia has been designed. HTML file is the original document including the dashboard overview containing Home, Real-Time Data, Precautions, Contacts and Rainfall parts. CSS was responsible for making visual improvements, setting colour-coded alerts, and creating a tunnel-themed background. JS provided the opportunity to request the latest data from the server and then periodically receive it using AJAX (Asynchronous JavaScript and XML) service without page reloading and, for the demonstration, the dashboard was further complemented by the animation of the rain and the cloud effects. This means that the current water level, the tunnel status (Open / Closed), and the traffic light signals were visible to both users and authorities using the dashboard.

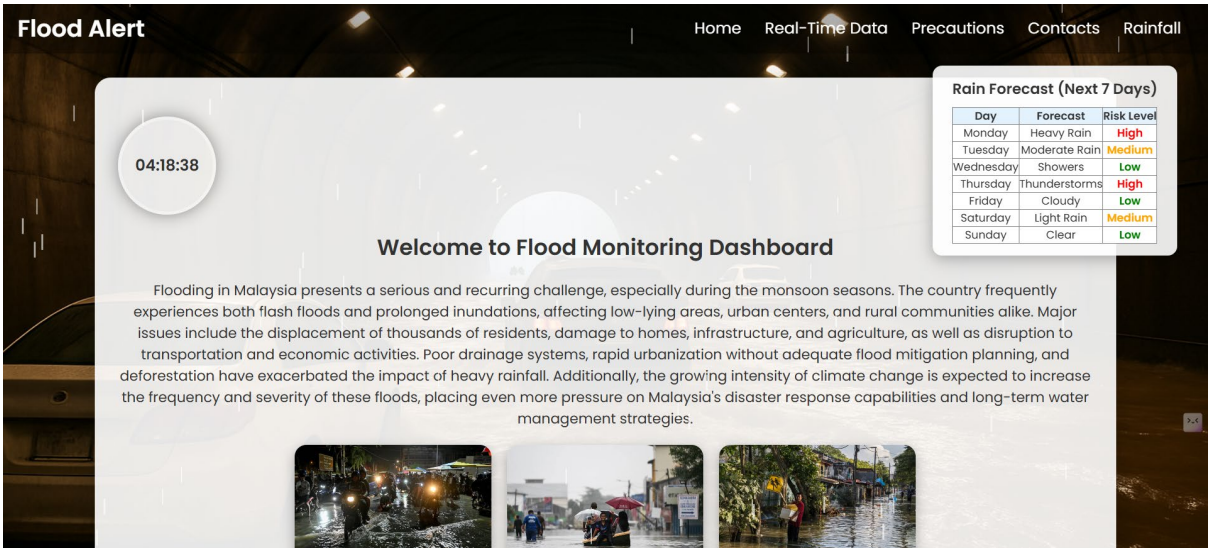


Figure 4.13 Web dashboard interface

4.4.7 Email Notification Setup

To notify authorities during flooding emergencies, an auto-email notification system was put in place utilizing PHPMailer and Gmail's SMTP server. PHPMailer is one of the most widely used PHP libraries for sending emails securely. Within the Gmail account, an app-specific password was generated for added security upon authentication. NodeMCU initiates a POST request to a PHP script on the server whenever the ultrasonic sensor detects a dangerous water level. The script below utilizes PHPMailer to generate and send the alert email with the current water level to the relevant authority. This setup offers real-time email alerting without reliance on third-party paid services. Figure 4.14 shows the email notification on the lock screen.

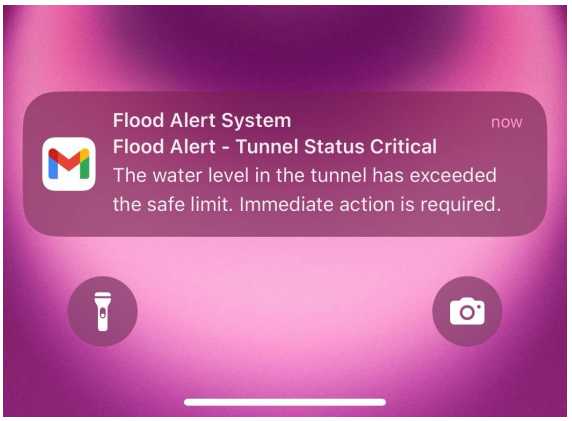


Figure 4.14 Web dashboard interface

4.5 Prototype Coding

Prototype coding is critical to this project as it undertakes vital tasks such as controlling hardware and data transfer. The C/C++ language was used to develop the program and upload it into the Arduino IDE. Two microcontrollers were dealt with: Arduino Uno R3, which is responsible for the flood detection and the triggering of the hardware responses, and NodeMCU (ESP8266), which transmits sensor information to the PHP server. The text and code are given in the next sections, respectively, and the figures that serve as supplementary material are shown in the upcoming sections.

4.5.1 NodeMCU ESP8266

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

// Wi-Fi credentials
const char* ssid = "stargirl";
const char* password = "iwontgive";
const char* serverName = "http://172.20.10.10/flood_monitoring_system/insert_data.php";
```

Figure 4.15 NodeMCU ESP8266 Program Code (Part 1)

These lines bring up the libraries which are needed for the Wi-Fi and HTTP connection to be established. The Wi-Fi password and the PHP server URL are set. Both of these will be used to connect and send data.

```
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);

  Serial.println("Connecting to WiFi...");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting...");
  }
  Serial.println("Connected to WiFi");
}
```

Figure 4.16 NodeMCU ESP8266 Program Code (Part 2)

With this block, we can communicate the NodeMCU with the serial and Wi-Fi. It displays the connection status every second until it connects.

```
21  void loop() {
22    String distanceValue = "";
23
24    // Check if Arduino sent something
25    if (Serial.available()) {
26      distanceValue = Serial.readStringUntil('\n'); // Read distance as string
27
28      distanceValue.trim(); // Clean any extra spaces/newlines
29
30      if (distanceValue.length() > 0) {
31        String status = "open";
32
33        int distance = distanceValue.toInt();
34
35        // Determine status
36        if (distance < 10) {
37          status = "closed";
38        }
39
40        Serial.print("Distance: ");
41        Serial.print(distance);
42        Serial.print(" cm, Barrier: ");
43        Serial.println(status);
```

Figure 4.17 NodeMCU ESP8266 Program Code (Part 3)

The part describes the sensor distance as read from Arduino through Serial. It is processed and verified that the value is within the range, then, it is identified if it is an open or closed flood based on the threshold. The status is displayed for confirmation.

```
45        // Send to server
46        if (WiFi.status() == WL_CONNECTED) {
47          HTTPClient http;
48          WiFiClient client;
49
50          http.begin(client, serverName);
51          http.addHeader("Content-Type", "application/x-www-form-urlencoded");
52
53          String httpRequestData = "distance=" + String(distance) + "&status=" + status;
54          int httpResponseCode = http.POST(httpRequestData);
55
56          if (httpResponseCode > 0) {
57            Serial.println("Data sent successfully");
58          } else {
59            Serial.print("Error sending data: ");
60            Serial.println(httpResponseCode);
61          }
62
63          http.end();
64        } else {
65          Serial.println("WiFi Disconnected!");
66        }
67      }
68    }
69
70    delay(1000);  // Read & send every second
71  }
```

Figure 4.18 NodeMCU ESP8266 Program Code (Part 4)

The following passage is communicating the distance together with the state of a connection to the PHP server using the POST method of HTTP. The response from the server is then printed on the screen, and Wi-Fi disconnection error is handled.

4.5.2 Arduino Uno

```
#include <Servo.h>

const int trigPin = 2;
const int echoPin = 3;
const int greenLight = 9;
const int yellowLight = 10;
const int redLight = 11;
const int buzzer = 8;
const int servoPin = 6;

Servo barrierServo;

void setup() {
    Serial.begin(115200);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(greenLight, OUTPUT);
    pinMode(yellowLight, OUTPUT);
    pinMode(redLight, OUTPUT);
    pinMode(buzzer, OUTPUT);
```

Figure 4.19 Arduino Uno Program Code (Part 1)

This part is where not only the Servo library is included but also the pin assignments are defined for each component (sensor, LEDs, buzzer, servo), and pin modes are set. In addition, serial communication is done to transfer the sensor data to NodeMCU.

```
    digitalWrite(greenLight, LOW);
    digitalWrite(yellowLight, LOW);
    digitalWrite(redLight, LOW);
    digitalWrite(buzzer, LOW);

    barrierServo.attach(servoPin);
    barrierServo.write(0); // Start at open position
}
```

Figure 4.20 Arduino Uno Program Code (Part 2)

All outputs are switched off at startup. The servo has been connected and set to 0° (barrier open).

```
31  void loop() {
32      long duration;
33      int distance;
34
35      digitalWrite(trigPin, LOW);
36      delayMicroseconds(2);
37      digitalWrite(trigPin, HIGH);
38      delayMicroseconds(10);
39      digitalWrite(trigPin, LOW);
40
41      duration = pulseIn(echoPin, HIGH);
42      distance = duration * 0.034 / 2; // Convert to cm
43
44      // Send only the distance value to Serial for NodeMCU
45      Serial.println(distance);
46
47      if (distance > 25) { // Safe distance
48          digitalWrite(greenLight, HIGH);
49          digitalWrite(yellowLight, LOW);
50          digitalWrite(redLight, LOW);
51          digitalWrite(buzzer, LOW);
52          barrierServo.write(0); // Open barrier
53      } else if (distance > 15) { // Moderate distance
54          digitalWrite(greenLight, LOW);
55          digitalWrite(yellowLight, HIGH);
56          digitalWrite(redLight, LOW);
57          digitalWrite(buzzer, LOW);
58      } else { // Too close
59          digitalWrite(greenLight, LOW);
60          digitalWrite(yellowLight, LOW);
61          digitalWrite(redLight, HIGH);
62          digitalWrite(buzzer, HIGH); // Turn on buzzer
63          barrierServo.write(90); // Close barrier
64      }
65
66      delay(100); // Small delay to stabilize readings
67  }
```

Figure 4.21 Arduino Uno Program Code (Part 3)

This is the most important logic. The sensor detects the water level. The system uses the distance to switch the traffic lights, shut the barrier with the help of the servo, and activate the buzzer. The distance is further transmitted via the Serial to NodeMCU.

4.6 System Modules Application

This part describes the architecture and features of the Flood Alert web portal. The site serves as an easily accessible interface for users to view the latest flood information, get alerts, be informed, and have multiple emergency contacts. The system is comprised of various modules, each having its task to deliver the necessary information to the public and coordinate the actions with the authorities if the tunnel gets flooded in any potential event.

4.6.1 Home Module

The Home module is a page in a system that users get to see first, instantly. The page has a real-time clock (which is circular in shape and in the centre), a warm, welcoming message about the project, and a 7-day rainfall forecast table. The page shows users visually the whole system, and the current weather info is immediately available. The design is clear and informational to make usability better.

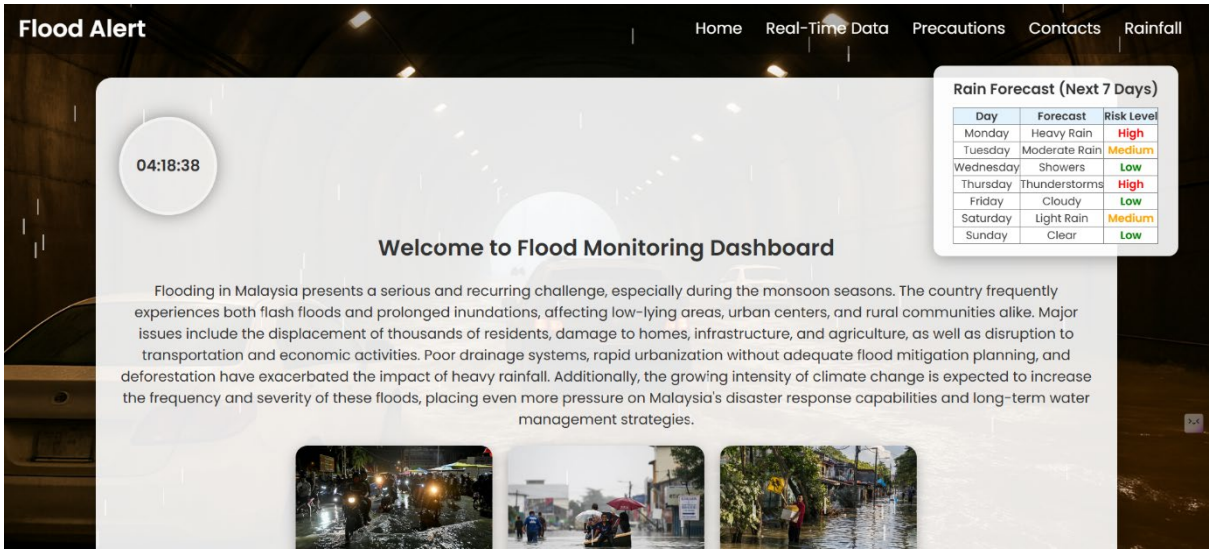


Figure 4.22 Flood Alert Home Page Interface

4.6.2 Real-Time Data Module

The Real-Time Data module is the heart of the whole system that keeps track of the water level, population and power stations in real time. The system also displays to the user the latest water level in centimeters, the current tunnel state and the exact time of the last sensor update. A line chart also gives a visual aid for the last week of measurements, which makes it easier for users to track changes and draw conclusions. For real-time updates, the module utilizes AJAX to pull data from a PHP script continuously every few seconds.

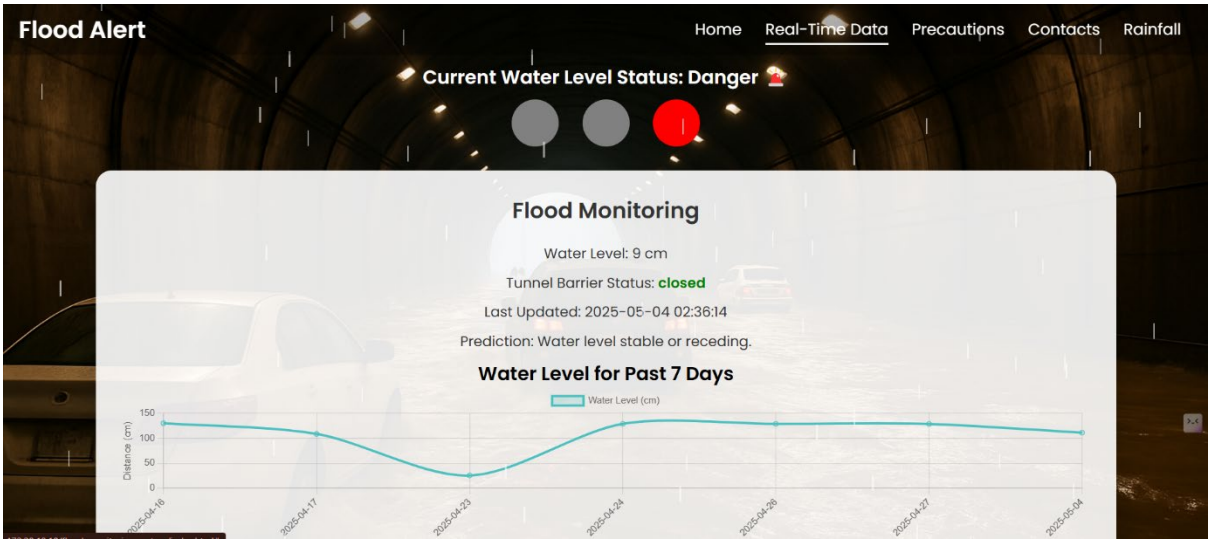


Figure 4.23 Real-Time Water Level and Status Display

4.6.3 Precautions Module

The Precautions module is a guide to the most crucial flood safety precautions to reach and inform the general public. It recommends the things people should do before and the measures they should undertake during the flood. This includes guidelines on what actions should be taken such as to keep away from low-lying areas, to unplug all electrical appliances, and to prevent the things you own from being scattered. The preventive measures presented here are broken down most clearly and directly possible using bullet points and are easily consumable for the reader.

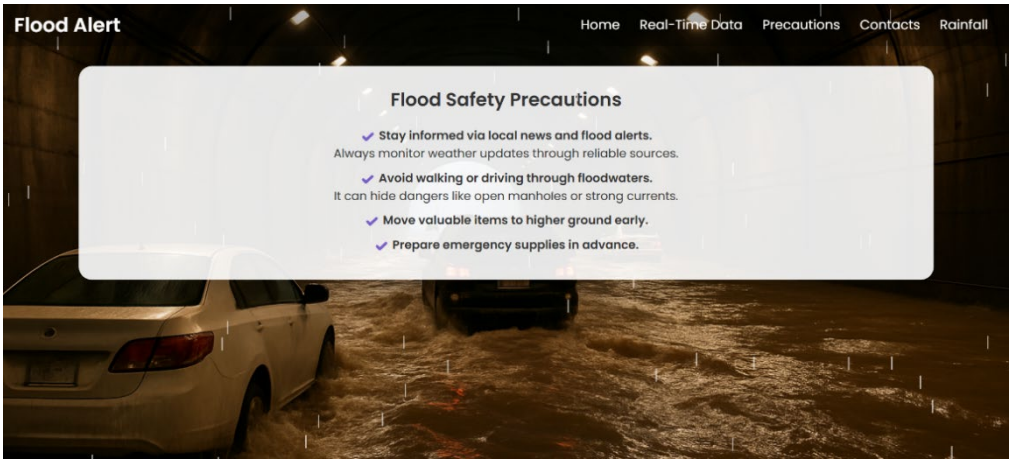
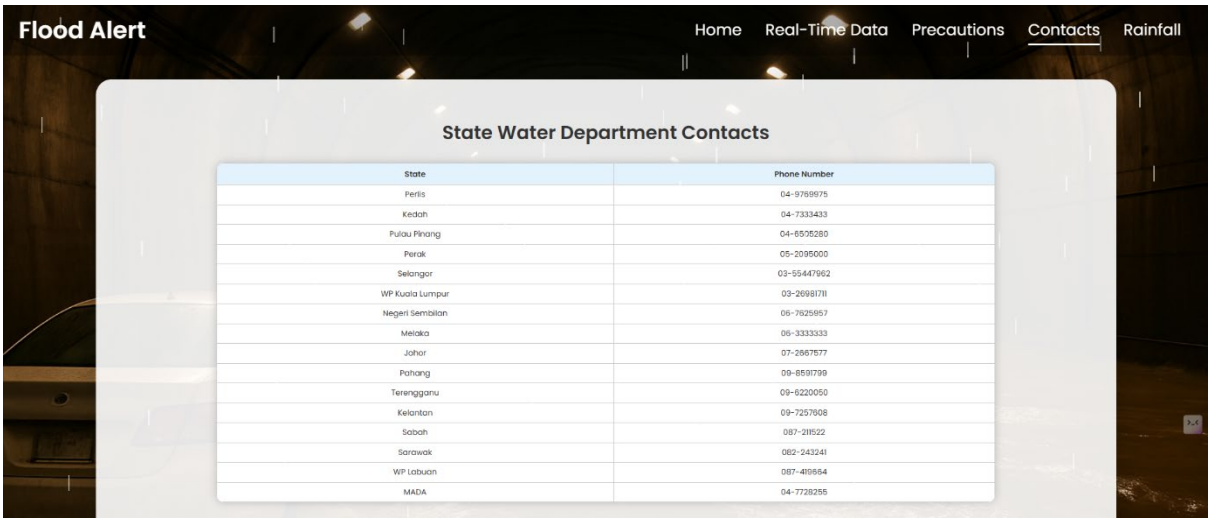


Figure 4.24 Flood Safety Tips Page

4.6.4 Contacts Module

The emergency phone numbers for water departments in several Malaysian states are provided in this module and organized into table cells. The contact details are available for immediate use in case of flooding, and easily reachable by both users and authorities, thanks to the table format organization.



State	Phone Number
Perlis	04-9769975
Kedah	04-7333433
Pulau Pinang	04-6505280
Perak	05-2095000
Selangor	03-55447962
WP Kuala Lumpur	03-26981711
Negeri Sembilan	06-7625957
Melaka	06-3333333
Johor	07-2667577
Pahang	09-8591799
Terengganu	09-6220050
Kelantan	09-7257608
Sabah	087-211522
Sarawak	082-243241
WP Labuan	087-419554
MADA	04-7726255

Figure 4.25 State Water Department Contact Numbers

4.6.5 Rainfall Module

This section is about the classification of underlying rainfall information based on specific graphic representations like light, moderate, heavy, and very heavy rainfall. The bottom part, on the other hand, is a chart that shows up-to-the-minute water gauge readings together with their timestamps, which are saved in the database. It gives people the opportunity to comprehend the relative magnitude of rainfall and the way it is associated with flood stages that rise.

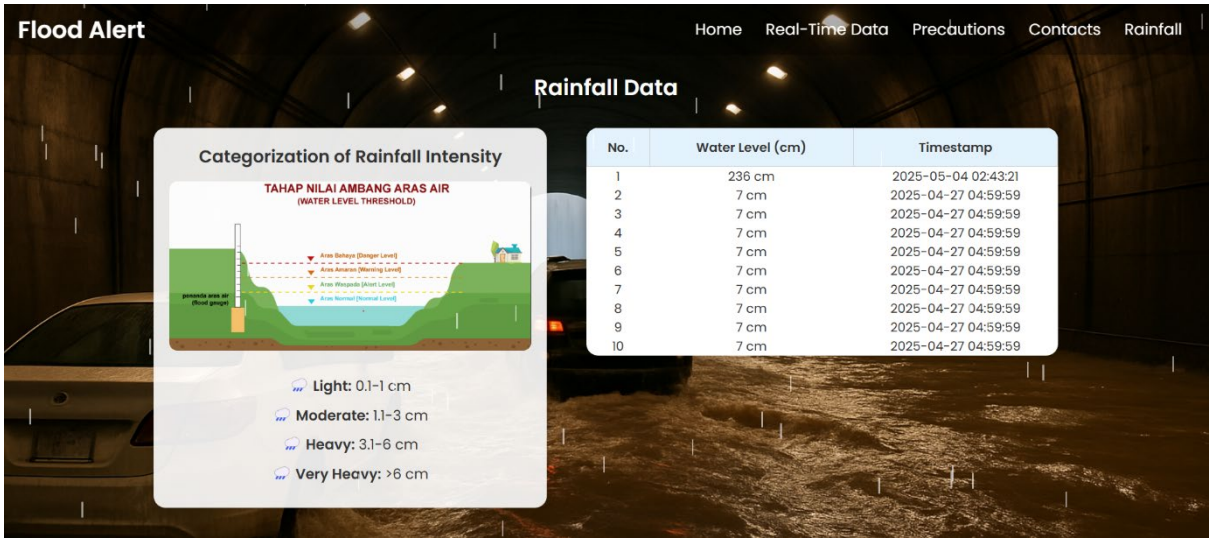


Figure 4.26 Rainfall Intensity Image and Water Level Table

4.7 Conclusion

In this chapter, we discussed the evolution and implementation of the Tunnel Flood Detection System. We provided an implementation of both the hardware and software components in which the Arduino Uno implemented sensor readings, control barriers, and sent emergency alerts via email, and the NodeMCU transmits data to our web server. Our web portal displays the real-time water levels, rainfall entered by the user, safety instructions, and emergency contact information through HTML, CSS, JavaScript, and PHP. The project accomplished three objectives: we successfully implemented a sensor system to detect flood waters, we engaged readiness responses through control barriers, and we displayed information clearly to end-users of the system, as well as authorities, for use, either in real time or as research information.

CHAPTER 5: CONCLUSION AND FUTURE WORK

5.1 Introduction

This chapter concludes the overall Flood Alert System project using a web-based monitoring system and physical flood detection via sensors in a tunnel. It identifies the success of the project, problems encountered, and suggestions for further development.

5.2 Achievements

It took some months, but finally, the prototype of the Flood Alert System was operated in the field. The principal results are as follows:

Table 5.1 Objectives and Achievements

Objectives	Achievements
Designing a flood warning system that can sense water levels in tunnels	A prototype was made for the proposed system with the help of Arduino Uno and ultrasonic sensor for measurement of the depth of water.
To develop a system for transmitting live data and alerts	Distance readings, from the NodeMCU, are transmitted to a web server, and when the water level is dangerously high/low, it will send an email via PHP.
A web interface to monitor the data gathered	A HTML, CSS, PHP, and MySQL-based responsive website was designed to show live water levels and system status.
For installing an automatic barrier device	A servo-driven gate was set to close in case of a flood.

5.3 Problems Encountered

Several difficulties were encountered during the development:

i. **Sensor instability:** The Ultrasonic sensor reading is full of interference and power supply noise at the beginning, which influences the water level detection reliability.

ii. **GSM module issues:** Our initial plan to use a GSM module (SIM900A) was abandoned as it only accepts 2G SIM cards, which are scarce or have been withdrawn by Malaysian mobile carriers.

iii. **Email senders:** We tested multiple SMTP senders, including Brevo, Mailjet, and MailSlurp, however, they all required business domains and or lengthy verification. After some attempts, we finally got Gmail SMTP to work via the PHPMailer library and App Passwords for our Google account.

iv. **Website connection issues:** The NodeMCU struggled with HTTP POST requests to the local server, with errors that had to be debugged several times until the connection was stable and data transfer accurate.

v. **Time limitations:**It took us a lot of time to integrate the hardware, debug the software and diagnose the email delivery issues that delayed parts of the implementation.

5.4 Future Work

Several limitations can be considered for the future work of this project:

i. The integration of GSM/4G Communication Modules: The existing system uses Wi-Fi for data and alert message communication. This is a limitation in areas where internet connection is non-existent or unreliable. Incorporating GSM or 4G modules (such as SIM900A or SIM7600) would allow the system to provide SMS notifications and initiate automated calls.

ii. Development of a Mobile Application Interface: To facilitate user convenience and monitoring ease, a separate mobile application could be developed for both iOS and Android operating systems. The app could comprise real-time tunnel status reports, push notifications, as well as access to important emergency information.

iii. Multi-Tunnel Monitoring Capability: The system is limited to monitoring only one tunnel. Development in the future could involve increasing the system to handle multiple locations of tunnels, each with its own sensor module and all reporting to a central monitoring system.

iv. Automated Pumping System Implementation: To change the system from passive observation to active flood management, future models can incorporate an automated pump system. Based on sensor readings, the system can activate pumps to drain the accumulated water from beneath the tunnel once a critical limit is met. Although not as widely applied yet in Malaysian tunnel engineering, such systems are applied in urban flood defense (such as Kuala Lumpur underpasses) and might be successfully utilized to assist in reducing flood effect in vulnerable tunnels.

5.5 Conclusion

The Tunnel Flood Detection System with Barrier Control and Notification Mechanism has been developed and met its target. The prototype can monitor flood levels, raise an automatic barrier and respond to synchronised flooding by sending real-time email alerts to relevant officials. The website dashboard makes the tunnel status clear and available to users. Despite several limitations that the implementation faced, the system stood as a solid basis for a real-world, scalable tunnel flood monitoring system.

REFERENCES

Darwis, M., Banna, H. a. A., Aji, S. R., Khoirunnisa, D., & Natassa, N. (2023). IoT Based Early Flood Detection System with Arduino and Ultrasonic Sensors in Flood-Prone Areas. *JURNAL TEKNIK INFORMATIKA*, *16*(2), 133–140. https://doi.org/10.15408/jti.v16i2.32161

Dewi, S. S., Satria, D., Yusibani, E., & Sugiyanto, D. (2018). Design of web based fire warning system using Ethernet Wiznet W5500. In *Emerald reach proceedings series* (pp. 437–442). https://doi.org/10.1108/978-1-78756-793-1-00073

Dhebe, S., Dhalge, H., Suryavanshi, V., & Shinde, H. (2023). Flood Monitoring and Alerting System. *International Journal for Research in Applied Science and Engineering Technology*, *11*(8), 223–230. https://doi.org/10.22214/ijraset.2023.55145

Kondaveeti, H. K., Kumaravelu, N. K., Vanambathina, S. D., Mathe, S. E., & Vappangi, S. (2021). A systematic literature review on prototyping with Arduino: Applications, challenges, advantages, and limitations. *Computer Science Review*, *40*, 100364. https://doi.org/10.1016/j.cosrev.2021.100364

Magnaye, A. a. B., Cruz, S. M. S., Reguyal, V. X. M., Ascaño, A. J. L., Quilit, M. a. A., & Limos-Galay, J. A. (2024). Automated flood water level sensor and alarm system using Arduino Uno. *International Journal of Research Studies in Educational Technology*, *8*(3). https://doi.org/10.5861/ijrset.2024.8022

Natividad, J. G., & Mendez, J. M. (2018). Flood monitoring and early warning system using ultrasonic sensor. *IOP Conference Series Materials Science and Engineering*, *325*, 012020. https://doi.org/10.1088/1757-899x/325/1/012020

Satria, D., Yana, S., Munadi, R., & Syahreza, S. (2018). Design of Information monitoring system Flood based Internet of things (IoT). In *Emerald reach proceedings series* (pp. 337–342). https://doi.org/10.1108/978-1-78756-793-1-00072

Zahir, S. B., Ehkan, P., Sabapathy, T., Jusoh, M., Osman, M. N., Yasin, M. N., Wahab, Y. A., Hambali, N., Ali, N., Bakhit, A., Husin, F., Kamil, M., & Jamaludin, R. (2019). Smart IoT Flood Monitoring System. *Journal of Physics Conference Series*, *1339*(1), 012043. https://doi.org/10.1088/1742-6596/1339/1/012043