DEVELOPING A REAL-TIME OBJECT DETECTION SYSTEM ON FPGA

Master thesis of Paris-Saclay University

Specialization: M2 Communication and Data Engineering

Master student: NGUYEN Trung Kien

Committee

Arnaud BOURNEL	Chairman
Université Paris-Saclay	
Pierre DUHAMEL	Reporter
CNRS, Laboratory of Signals and Systems	
NGUYEN Linh Trung	Examiner
VNU University of Engineering and Technology	
BUI Duy Hieu	Examiner
VNU Information Technology Institute	
Erwan LIBESSART	Examiner
CentraleSupélec	

Thesis supervision

BUI Duy Hieu	Thesis supervisor
VNU Information Technology Institute	
TRAN Thi Thuy Quynh	Co-supervisor of the thesis
VNU University of Engineering and Technology	
Erwan LIBESSART	Co-supervisor of the thesis
CentraleSupélec	

ABSTRACT

AUTHORSHIP

Hanoi, January 22nd, 2024

Author

ACKNOWLEDGEMENT

Contents

List of Abbreviations

Abbreviation	Definition
IoT	Internet of Things
ADAS	Advanced Driver Assistance Systems
AI	Artificial Intelligence
AMD	Advanced Micro Devices
mAP	Mean Average Precision
ASIC	Application Specific Integrated Circuit
AVC	Advanced Video Coding
CBAM	Convolutional Block Attention Module
CNN	Convolutional Neural Network
COCO	Common Objects in Context
CPU	Central Processing Unit
DMA	Direct Memory Access
FPGA	Field Programmable Gate Array
FPS	Frames Per Second
GB	Gigabyte
GE	Gate equivalents
GPU	Graphics Processing Unit
HD	High Definition
HOG	Histogram of Oriented Gradients
LTS	Long Term Support
MJPEG	Motion JPEG
MOT15	Multiple Object Tracking Benchmark 2015
MOTC	Multiple Object Tracking Challenge
MPSoC	MultiProcessor System On Chip
NMS	Non-Maximum Suppression
OS	Operating System
PASCAL	Pattern Analysis Statistical Modelling and Computational Learning
PETS09	Performance Evaluation of Tracking and Surveillance 2009
PTZ	Pan-Tilt-Zoom
RAM	Random Access Memory
RCNN	Regional Convolutional Neural Network
RGB	Red Green Blue
SIFT	Scale-Invariant Feature Transform
SPP	Spatial Pyramid Pooling
SRAM	Static Random Access Memory
SSD	Single Shot Detector
SVM	Support Vector Machine

TSMC	Taiwan Semiconductor Manufacturing Company
UHD	Ultra High Definition
VNU	Vietnam National University
VOC	Visual Object Classes
YOLO	You Only Look Once
ZCU106	Zynq UltraScale+ ZCU106 Evaluation Kit

List of Figures

List of Tables

Introduction

Chapter 1. Real-time Object Detection System

This chapter provides an overview of the application of Internet of Things (IoT) technology in monitoring and alerting fire detection systems, primarily in the context of increasingly serious fire problems that require critical attention. It begins with an introduction to IoT and the importance of its role in improving monitoring skills and its value in fire situations. In addition, this chapter also focuses on the main objective of the thesis: to explore and employ the full potential of IoT-connected hardware devices in both IoT applications and real-world scenarios. This chapter is organized into three sections. Section 1.1 provides an overview of the IoT and its applications in fire monitoring systems. *Section 1.2* discusses the current situation and limitations of existing fire monitoring systems. *Section 1.3* presents modern fire detection methods using current technologies, along with their limitations.

1.1. Introduction to the Internet of Things (IoT) and Its Applications for Fire Monitoring Alert Systems

In recent years, fire accidents have become increasingly dangerous, causing critical damage to both human life and property. Traditional fire monitoring systems, which often rely on manual checks or local alerts, are limited in their ability to respond quickly and effectively, especially in complex or remote environments. To handle this issue, we need smart and effective solutions to improve it in time.

The IoT is one exciting technical development that resolves these issues. As its name suggests, IoT refers to a network of smart devices that can communicate and connect with each other, exchanging data. These devices can collect, transmit, and process data in real-time, thereby enabling automation and making personal decisions.

Unlike standalone devices that work independently, a true IoT system (Figure 1) is a scalable and integrated system where multiple devices work together for a common goal. In the context of fire detection, this goal is always monitoring environmental conditions, identifying early signs of fire, and sending real-time alerts, even when no one is present at that location.

By providing features such as remote monitoring, automated notifications, and log data, IoT-based systems significantly enhance responsiveness and overall safety. Therefore, using IoT for fire monitoring systems is both useful and essential for enhancing early detection and minimizing damage.

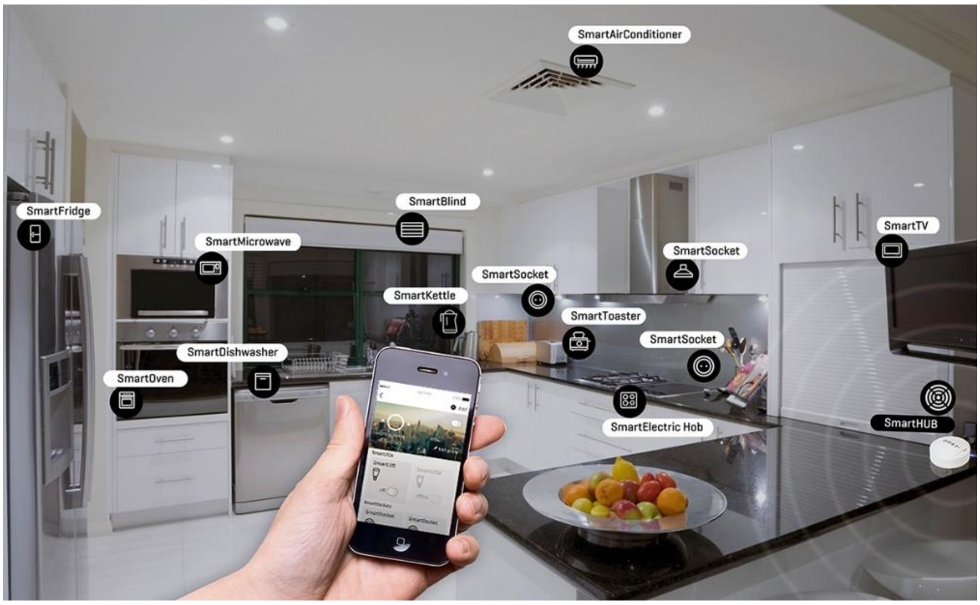


Figure 1. An IoT system with interconnected smart devices.[gắn link]

1.2. Current situation and Limitations of the monitoring fire alert system

Fire accidents are now common and can occur in various types of environments, ranging from crowded residential areas to industrial zones, forests, etc,.. In an industrial environment, the use of flammable materials, high electricity consumption, and poor ventilation are common reasons that increase the risk of fire. In a residential area, the complex and overloaded electrical network increases the chance of short circuits and electrical fires. In a forest environment, rising global temperatures make it more flammable, and also human activities, such as camping or purposeful burning of the forest, increase the risk of fire. As mentioned, the fire can occur anywhere, any time, whether in crowded urban areas or isolated natural areas.

Therefore, deploying fire monitoring systems with sensors and data-collecting modules in large areas is expensive and challenging. Traditional systems, which rely on wired connections or require continuous human monitoring, are not realistic, especially in large areas or remote and harsh environments. These systems will face problems such as processing complex data and false fire alerts. This is really a big challenge for the fire warning system. The solution using IoT with sensor data can help provide more accurate and useful results.

Currently, traditional fire monitoring and alert systems are still widely used and have not been fully replaced by IoT-based systems. Although some improvements have been made, many issues and limitations still exist, such as detection delays, limited coverage range, and high maintenance requirements. As a result, developing a fire detection and monitoring system that uses fire, temperature, and humidity sensors combined with wireless communication technologies is a creative, possible, worthy solution to consider.

1.3. Related Works

The development of fire monitoring and alert systems based on IoT has gained increasing attention in recent years. This section presents several existing works and discusses their limitations, highlighting the gap that this project aims to address.

Currently, IoT systems typically integrate various sensors and microcontrollers, ranging from simple to advanced designs. Moreover, with the growing popularity of artificial intelligence (AI), many modern IoT applications are incorporating machine learning to improve detection accuracy. Although this thesis does not apply AI, it is planned as a direction for future work.

Regarding related works, one notable example is presented in [1], where the authors proposed an IoT-based fire detection and monitoring system using temperature, flame, and smoke sensors, paired with an ESP32 microcontroller. Their system employed a LoRa network to detect and alert about forest and farm fires. Similarly, in [2] , the authors developed an Arduino-based home fire alarm system that uses a GSM module and temperature sensor to send SMS alerts, thereby enhancing user safety and protecting property. In another study [3], the researchers designed a smart IoT-based system that also used temperature, flame, and smoke sensors like in [1],, but combined them with an Arduino UNO and a gas sensor.

Their system was implemented on the Blynk platform and used GSM communication for alerting. Meanwhile, in [4], an STM32-based wireless fire detection system was introduced, utilizing multiple sensors and embedded wireless technologies such as Wi-Fi to detect fires in real-time. A more advanced system is shown in [5], which focuses on the use of robotics in fire monitoring. It integrates STM32 and STC89C51 microcontrollers, along with drones, to create a combined air and ground monitoring system, with communication facilitated via ZigBee. On the other hand, [6] emphasizes the alerting aspect. This system uses an ESP32 with PIR sensors to detect both fire and movement and sends alerts via a Telegram bot to smartphones, including visual and temperature-based alarms.

As AI continues to develop, some recent papers have begun integrating it with IoT for fire detection. For instance, [7] presents a system that combines IoT devices with the YOLOv5 model to enable early and real-time forest fire detection, aiming to reduce false alarms and enhance safety during dry seasons. Building upon that, [8] proposes an improved system using IoT and machine learning, integrating various sensor types to boost detection accuracy and support remote monitoring. Although recent works reflect technological advancements, the core concept of IoT-based fire detection was explored much earlier. For example, [9] describes a real-time fire alarm system developed using Raspberry Pi and Arduino Uno. This system included smoke detection, room image capture, and alerting via SMS and a web interface. Notably, it featured a user confirmation step before notifying firefighters, effectively reducing false alarms. Despite having a basic web interface, the implementation was comprehensive and forward-thinking, even including features like user login.

As stated earlier, while this thesis does not yet include AI integration, it aims to build upon these foundational ideas. Notably, many existing systems use Wi-Fi or GSM for communication. In contrast, this thesis proposes an enhanced approach by adopting **radio communication** as the core of an IoT-based fire detection and alert system. Ultimately, this work is made possible thanks to the pioneering efforts of these earlier researchers, upon whose contributions this thesis continues to build.

1.4. Conclusions

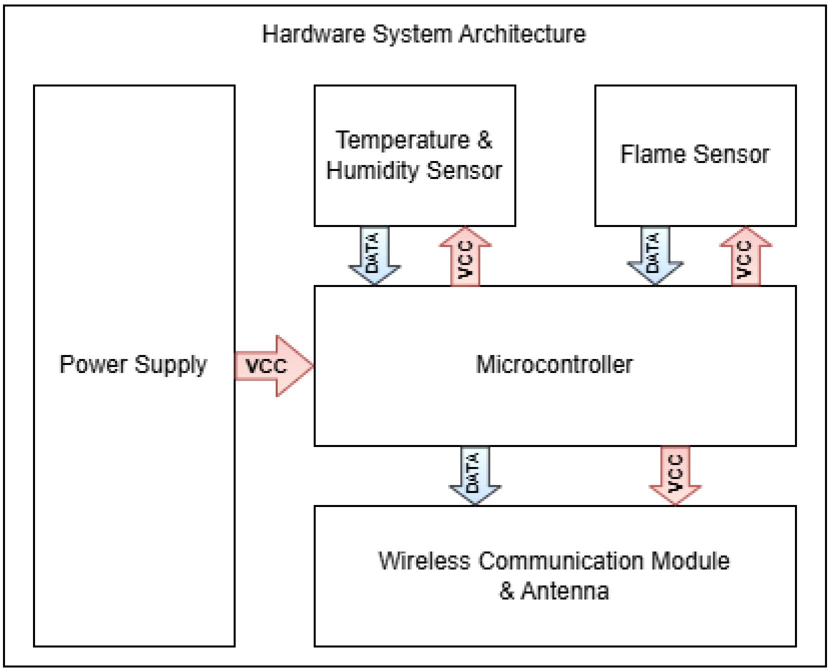
An IoT system is an essential technology with applications in monitoring and fire detection. Achieving real-time monitoring, high accuracy, processing all scenarios, and overcoming hardware limitations creates serious difficulties. The use of IoT brings a promising solution, enabling real-time monitoring, remote access, and automated alert notifications. This project aims to inherit and integrate the ideas from those previous works, while also contributing further improvements to develop a more responsive and reliable fire alert system. It also introduces new innovations to improve the system's flexibility and performance in real-life situations. In Chapter 2 *System Architecture Design*, a proposal will be presented, focusing on the integration of hardware and software components, including sensor modules, communication systems, and control logic. All of which will be implemented through software to build the proposed IoT-based fire monitoring solution.

Chapter 2. System Architecture Design

The first chapter of this thesis provided an overview of IoT technology and its applications in fire detection and monitoring. It also discussed the current state of fire accidents occurring in many places today, highlighting the need for more efficient solutions. Furthermore, with the current state of fire, the thesis proposed a new direction by enhancing existing ideas with the use of radio communication. While many previous systems deploy on Wi-Fi or GSM, this thesis focuses on using *low-cost, low-power* wireless communication modules. The main goal is to develop an IoT-based system that monitors, detects fires, and alerts in real-time.

Building on this ideal, *Chapter 2* presents the proposed system architecture, which consists of both hardware and software components designed to work together for efficient fire monitoring. The system includes: Raspberry Pi and STM32 microcontrollers, HC-12 modules for communication, and a variety of sensors to collect fire-related data. This chapter has three main sections: *Section 2.1 Hardware System Architecture*: in this section, an overview of the hardware components used in this thesis will be presented, including sensors, microcontrollers, and the central processing unit. Each component will be explained with the reason why it is used and a summary of its advantages and disadvantages. *Section 2.2: Software & Firmware System Architecture*: This part details how the software & firmware work across the system. It begins with sensor-side firmware, which includes collecting data, encoding, and transmission. On the server side, it explains how data is received, decoded, stored, and visualized through a Flask-based web application. This section also includes user authentication, account management, and how software sends alerts (using Fuzzy Logic and Threshold Evaluation). *Section 2.3 Conclusions:* This final section summarizes all components introduced in *Chapter 2* and prepares for the system implementation and evaluation that will follow in later chapters, *Chapter 3: Implementation Results and Evaluation*.

2.1. Hardware System Architecture


Hardware System Architecture
Power Supply
VCC
Temperature & Humidity Sensor
Flame Sensor
DATA
VCC
DATA
VCC
Microcontroller
DATA
VCC
Wireless Communication Module & Antenna

thg thu đâu?

Figure 2: Hardware System Architecture

This section introduces the hardware components used at both the *sensor node* and *the central processing unit*. It explains what each component is, its role in the system, and the reasons behind choosing it for the demo. Subsections cover specific components, including fire sensors, temperature and humidity modules, STM32 for edge processing, and the Raspberry Pi for central control and web server hosting. The following *section 2.2* will discuss the data flow in depth, and how each part works together.

2.1.1. Sensor Node Architecture

(1) Flame Sensor using KY–026

Flame sensors are devices used to detect the presence of fire by sensing specific types of radiation emitted by flames, especially infrared light in the 760 – 1100 nm range. In this thesis, the KY-026 flame sensor is selected due to its low cost, small size, and easy to use. Despite being a basic sensor, it is well-suited for demo and educational purposes, making it a practical choice for the project's goals and budget.

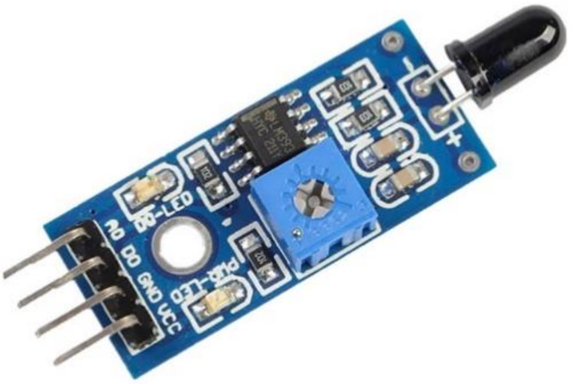


Figure 3. Flame sensor KY-026 [9].

(2) Temperature & Humidity Sensor using DHT11

A temperature and humidity sensor is used to collect environmental data, which is impotant in fire detection since fires typically cause a quick increase in temperature and quick decrease in humidity. Depend only on a flame sensor may cause inaccurate results, so combining it with temperature and humidity will improves detection reliability and reduces false alarms. Although there are many types of environmental sensors, this thesis uses the DHT11 due to its low cost, low power, and availability – making it ideal for the demo goals of the project and suitable for academic applications.

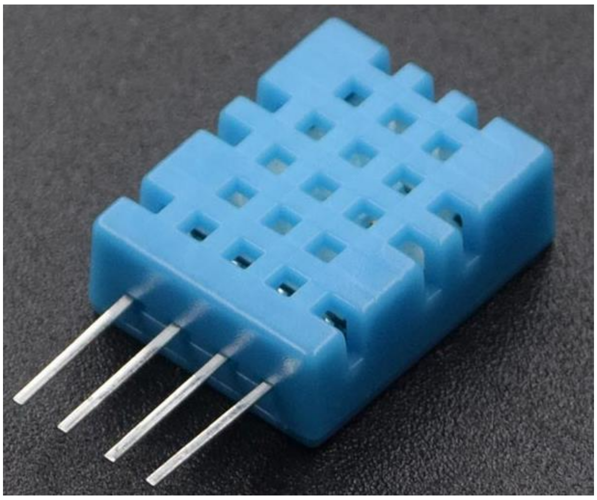


Figure 4: DHT11 - Temperatue & Humidity Sensor [10].

(3) Microprocessor using STM32F103C8T6

In an IoT system, the edge processor is a lightweight microcontroller responsible for collecting data from sensors such as flame, temperature, and humidity sensors.

This thesis uses the STM32F103C8T6 as the edge processor due to its suitability for the project's research and application goals. Its UART communication interface enables efficient serial data handling and transmission, making it a reliable choice for processing and transmitting sensor data in the proposed system.

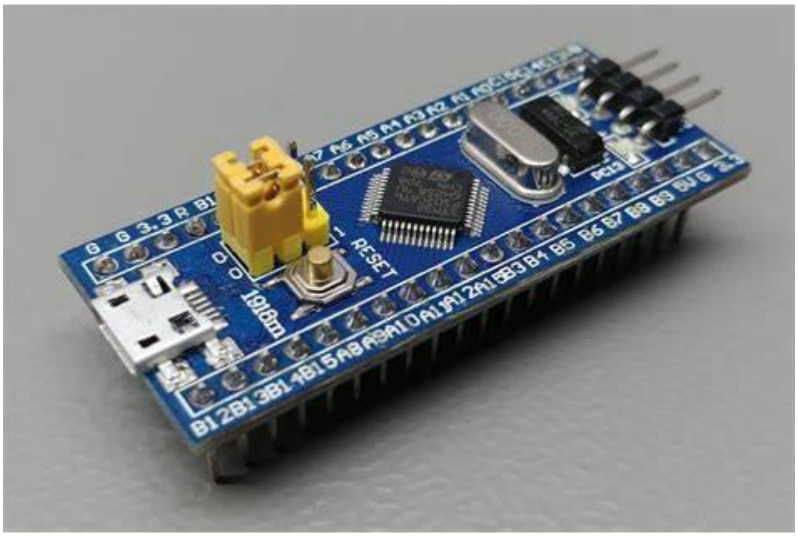

G G 3.3 R B11
A7 A6 A5 A4 A3 A2 A1 A0
RESET
B12 B13 B14 B15 A8 A9 A10 A11 A12 A15 B3 B4 B5 B6 B7 B8 B9 5V G 3.3
PC13

Figure 5: STM32F103C8T6 Microcontroller [11].

(4) Wireless Communication Module using HC-12

The wireless communication module is crucial in the proposed IoT system, enabling data transmission between sensor nodes and the central processing unit. While technologies like Wi-Fi, Zigbee, and Cellular require existing infrastructure, and LoRa offers long-range communication with higher cost and complexity, the HC-12 module provides a low-cost, easily configurable alternative well-suited to the project's goals. As described in Section 2.1.1, the HC-12 is integrated into the transmission block, where it wirelessly sends sensor data – processed by the STM32F103C8T6 microcontroller – through RF communication.

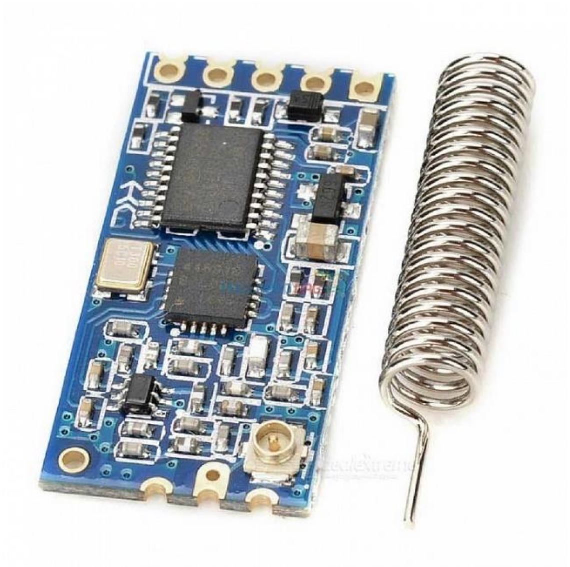


Figure 6: HC-12 Wireless Communication Module [12].

(5) Antenna

To enhance the wireless communication performance of the sensor node, an external 433 MHz antenna is connected to the HC-12 module through an SMA male connector. This antenna improves the range and stability of RF transmission between nodes and the central unit. Specifically, the selected antenna working at a frequency of 433 MHz, with a gain of 30 dBi, input impedance of 50 **Ω**, and a standing wave ratio (SWR) of ≤ 1.5. It has a height of 205 mm and supports a maximum input power of 100 W. The use of this antenna helps ensure reliable data transmission in both indoor and outdoor environments, supporting the system's goal of long-range, low-cost communication without requiring existing infrastructure.

Figure 7: 433 MHz RF Antenna [13].

2.1.2. Central Processing Unit System Architecture

(1) Wireless Communication Module using HC-12

In the *2.1.2 Central Processing Unit System Architecture*, the HC-12 module is placed in the receiver (Rx) block. After the data is transmitted from the node to the central processing unit through radio communication, the HC-12 module at the receiver receives incoming signals and transmits them to the central processor for handling.

(2) Central processing unit using Raspberry Pi 5

In this system, the central processing unit handles incoming data received wirelessly via the HC-12 module. To meet the project's goals, a Raspberry Pi 5 is used for its practical application in real-world IoT scenarios. As a single-board computer (SBC), the Pi 5 runs full operating systems like Raspberry Pi OS or Ubuntu and supports microSD storage up to 1 TB. Powered by a quad-core ARM Cortex-A76 processor and equipped with 8 GB of RAM, it delivers strong performance comparable to standard PCs. It also offers many connectivity, including USB, Ethernet, Wi-Fi, Bluetooth, micro HDMI, and a 40-pin GPIO header for interfacing with sensors and modules like HC-12. With its power, flexibility, and ease of integration, the Raspberry Pi 5 is well-suited for modern IoT applications.

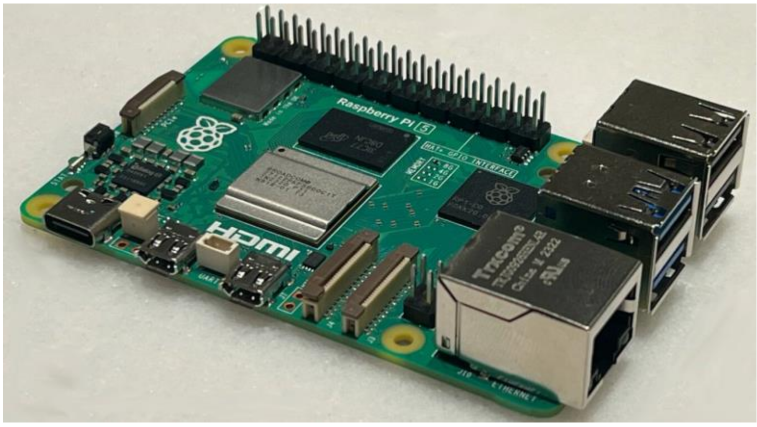


Figure 8: Single board Computer - Raspberry Pi 5 [14].

(3) Touchscreen Monitor using ASUS ZenScreen

Although this project does not focus on physical user interfaces, adding a touchscreen in the demo setup improves system visualization and user interaction. Therefore, an ASUS ZenScreen touchscreen is used. It is fully compatible with the custom monitoring software developed in this project. This addition enhances user understanding, simplifies testing, and provides a basis for future improvements.

Figure 9: Touchscreen [15].

2.2. Software & Firmware System Architecture

Section 2.1 focuses on introducing the hardware components used in this thesis, along with the reasons for their selection and their unique benefits. *Section 2.2. Software & Firmware System Architecture* discusses the working flow of the full system implementation. *Section 2.2* is divided into two subsections: *Section 2.2.1. Firmware at the Node* explains how data is handled at the sensor node after collecting data from flame, temperature, and humidity sensors. It also provides the standard requirements for protecting the privacy and security of data during wireless transmission. *Software at the Central Processing Unit* uses software to receive, decode, and process incoming data packets. The decrypted data will be uploaded in real-time onto a web-based platform. This subsection also covers user administration, data logging, notification handling, and other system operations, which will be described in further depth.

2.2.1. Firmware at the Node

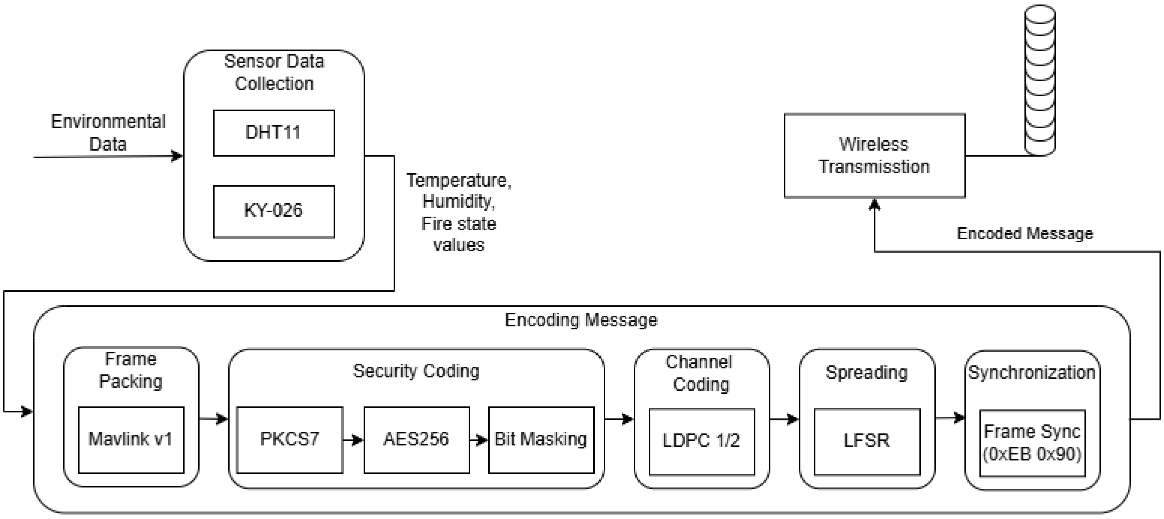


Figure 10: Firmware Architecture at the Sensor Node.

(1) Sensor Data Collection

Input: Value from the environment.

Output: Temperatue, Humidity, Fire state data.

During the Sensor Data Collection stage, environmental data is collected using two sensors: the DHT11 for temperature and humidity and the KY-026 for fire detection.

Algorithm 1: Read DHT11 Data and Fire Sensor State.

```
Loop forever
        start DHT11, confirm Presence (check DHT11 response)
        Rh_byte1, Rh_byte2, Temp_byte1, Temp_byte2, crc ← read five bytes
        valid ← ((Rh_ byte1+Rh_ byte2+ Temp _byte1+Temp_byte2) & 0xFF) = crc
        Temperature ← float(Temp_byte1)
        Humidity ← float(Rh _byte1)
        Fire ← readGPIO(GPIOA, PIN1)
end loop
```

The DHT11 employs a 1-Wire interface, with the STM32 initiating communication through GPIOA Pin 6 set as output with accurate time delays. After setting up, it returns 5 bytes (humidity, temperature, and checksum), and data validity is validated by checking the checksum. The KY-026 flame sensor, which is attached to GPIOA Pin 1 (digital) and, alternatively, GPIOA Pin 0 (analog via ADC), produces HIGH (1) when no flame is detected and LOW (0) when there is a fire.

(2) Decoding Message

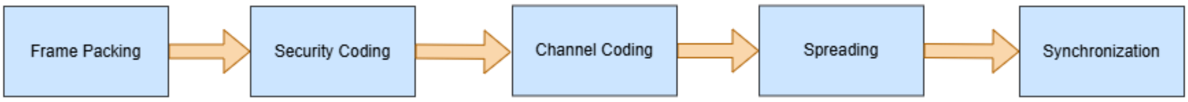


Figure 11: Data protection sequence in wireless transmission.

This is a highly important and interesting part when transmitting messages over wireless communication. It ensures compliance with standard requirements for protecting privacy and securing data in wireless transmission. In wireless data security, the main components typically include (Figure 12): Frame Packing, Security Coding, Channel Coding, Spreading, and Synchronization.

a) <u>Frame Packing using Mavlink v1</u>

Input: Temperature, Humidity, Fire state data.

Output: MavLink frame.

In the Frame Packing step of the thesis, sensor data – including temperature, humidity, and fire state – is packed into a standardized communication format using the MAVLink protocol (version 1). This protocol, widely used in embedded and wireless systems, ensures reliable and efficient data exchange. Each MAVLink frame comprises three sections: a 6-byte Header, a Payload of up to 255 bytes, and a 2-byte CRC.

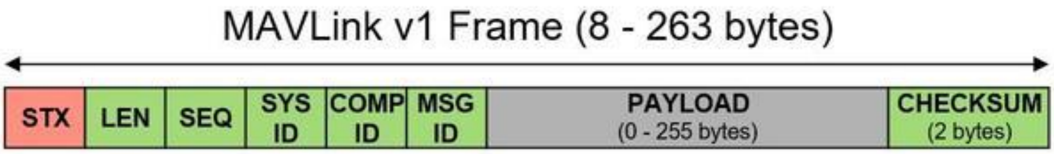


Figure 12: Mavlink v1 Frame [gắn link].

As in Figure 13, the Header includes the start byte (0xFE), payload length, a message sequence number, system ID, component ID, and message ID. The Payload carries the original sensor values (in hexadecimal format), while the CRC is used to verify data integrity and detect transmission errors. This structure enables robust, consistent communication between nodes in the fire monitoring system.

b) ***Security Coding using AES 256 in mode ECB***

Input: MavLink frame**.**

Output: Ciphertext.

In the Security Coding step, to enhance the protection of MAVLink messages, this thesis applies the *AES-256* block encryption algorithm, one of the most powerful and widely adopted encryption standards today. With a 256-bit key length and 14 rounds of transformation, *AES-256* provides strong resistance against modern cryptographic attacks. Because *AES-256* works with 16-byte blocks, the MAVLink message must be padded accordingly. The *PKCS7* padding is used: if the message length is not a multiple of 16, the missing bytes are filled with the number of padding bytes; if the message is already a multiple of 16, a full block of padding with the value 0x10 is appended. After padding, the message is divided into fixed-length blocks and encrypted sequentially with AES-256. Additionally, to further confuse the data, this thesis uses the bit masking technique – specifically, bit reversal, which reverses all the bits in each block.

c) ***Channel Coding using LDPC 1/2***

Input: Despread message.

Output: Channel decode message.

At this stage, LDPC decoding is employed to detect and correct errors caused by wireless transmission. As said in Spreading, a shared parity-check matrix H is applied to the received message to compute the syndrome. If the syndrome vector contains non-zero values, it indicates the presence of errors. The decoder then tries to correct the bit error by finding and flipping the bit related to the most incorrect parity. The syndrome is recalculated by using the calculate syndrome after each bit flip, and this process continues until the syndrome is zero or a maximum of ten iterations is reached. If problems continue after the allowed rounds, the message becomes uncorrectable and is destroyed.

d) Spreading using LFSR

Input: Channel-encoded message.

Output: Spread message.

In the Spreading phase, this thesis uses a Fibonacci-type Linear Feedback Shift Register (LFSR) to enhance communication safety and reliability. By generating pseudo-random sequences, the LFSR confuses the signal, making it more difficult to track, detect, and attack during wireless transmission. This step receives an LDPC-encoded message whose length has been doubled by channel coding. Each data block is XORed with the LFSR output, and the same register is shared between the transmitter and receiver for detection purposes.

e) Synchronization

Input: Channel-encoded message.

Output: Framed message.

After undergoing the processes of frame packing, encryption, channel coding, and spreading, the message becomes significantly confused, making it difficult for the receiver to identify the beginning or structure of the transmission. Therefore, as the final step, two header bytes (0xEB and 0x90) are added after LFSR encoding.

These specific bytes were chosen because 0xEB and 0x90 are commonly detectable in wireless environments. They frequently appear in the radio spectrum, making them reliable markers. By adding these two bytes at the beginning of the final message, the receiver can more easily synchronize and detect the start of a valid transmission.

In the Encoding Message section, each of these components will be described in detail through corresponding steps, allowing for a clear understanding of how the message is processed and prepared for transmission.

(3) Wireless Transmission (through UART)

Input: Framed message.

Output: Message transmitted.

In the Wireless Transmission stage, the framed message is sent from the STM32 to the HC-12 module using UART communication. The STM32 delivers the data over its UART interface, while the HC-12 module handles modulation and RF transmission. To ensure communication, both the STM32 and HC-12 must be configured with the same baud rate of 115200 and operate on the same channel. These parameters are set via AT commands during initialization. Once configured, the HC-12 broadcasts the message wirelessly, completing the transmission process.

2.2.2. Data Processing at Central Processing Unit

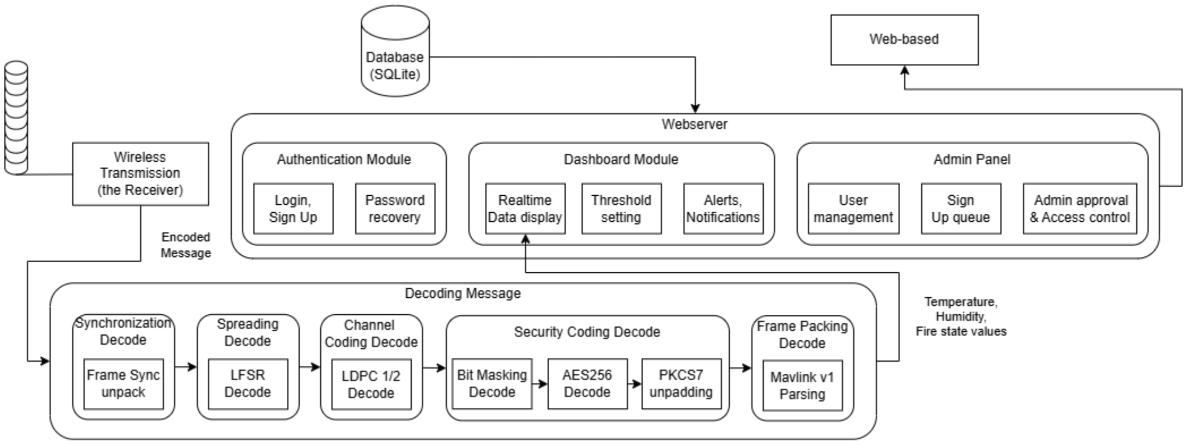


Figure 13: Software Architecture at the Central Processing Unit.

(1) Wireless Receiver

Input: Message transmitted.

Output: Message received.

On the receiver side, the HC-12 wireless module is pre-configured with the same AT command parameters, such as baud rate and channel, as described in the Wireless

Transmission section. After the message is successfully transmitted, the Central Processing Unit listens for incoming data through the USB UART.

(2) Decoding Message

a) Synchronization Decode

Input: Message received.

Output: Frame message unpack.

When the message is received through USB UART at the Central Processing Unit, it checks the validity of the message by inspecting the first two bytes. If the first two bytes are 0xEB and 0x90, the message is valid. The synchronization decode step then unpacks the frame by removing these two header bytes, allowing the system to proceed with further processing.

b) Spreading Decode

Input: Frame message unpack.

Output: Despread message.

Because both the transmitter and the receiver use the same register after applying the LFSR Fibonacci, to decode the spreading, XOR each block again with the LFSR Fibonacci output, just as in the LFSR encoding step.

c) Channel Coding Decode

Input: Despread message.

Output: Channel decode message.

At this stage, LDPC decoding is employed to detect and correct errors caused by wireless transmission. As said in Spreading, a shared parity-check matrix H is applied to the received message to compute the syndrome. If the syndrome vector contains non-zero values, it indicates the presence of errors. The decoder then tries to correct the bit error by finding and flipping the bit related to the most incorrect parity. The syndrome is recalculated by using the calculate syndrome after each bit flip, and this process continues until the syndrome is zero or a maximum of ten iterations is reached.

If problems continue after the allowed rounds, the message becomes uncorrectable and is destroyed.

d) Security Coding Decode

Input: Channel decode message.

Output: Plaintext.

After LDPC decoding, if the syndrome vector contains only zeros, the message is considered error-free and proceeds to the Security Coding Decode stage. As mentioned earlier, bit masking in this system involves a simple bit-reversal operation during encoding, so decoding applies bit reversal again to restore the original bit order, preparing the message for AES-256 decryption and PKCS7 unpadding.

In the AES-256 decryption phase, the same key used during encryption is reused, but the round keys are applied in reverse order, from the last to the first, across 14 decryption rounds. Once decryption is complete, the message enters the PKCS7 unpadding stage. Here, the decoder checks the last byte to determine the number of padding bytes added. For example, if the last byte is 0x03, the decoder verifies whether the final three bytes all equal 0x03. If so, they are removed; otherwise, the message is invalid. After successful unpadding, the original MAVLink frame is fully recovered.

e) Frame Packing Decode

Input: Plaintext.

Output: Temperature, Humidity, Fire state data.

In this final stage, the system verifies whether the plaintext corresponds to a valid MAVLink frame. If the first byte is 0xFE, it indicates the start of a MAVLink packet. The decoder then checks the LEN byte to confirm that the payload length matches the actual size, and verifies the SEQ byte to ensure message order – an important factor for logging and detecting missing or out-of-sequence packets. The SYS ID and COMP ID fields help identify which system and component sent the message. Most importantly, the CRC is recalculated from byte 2 onward (not including the start byte

0xFE) and compared with the received CRC. If they match, the frame is valid; otherwise, it is discarded. Once validated, the payload is extracted to obtain the final output: temperature, humidity, and fire status data.

2.2.3. Software development at Centrall Processing

This section presents the implementation of the software system on a web platform, aiming to improve user interaction, usability, real-time monitoring, and overall system management.

Đối với Web server, thesis này sử dụng ngôn ngữ lập trình chính là Python với mô hình MVC làm backend, build server bằng Flask, sử dụng database là SQLite. Ngoài ra về front end thì sử dụng HTML, CSS, JS.

Đối với build server bằng Flask sử dụng Python thì, Python là ngôn ngữ dễ sử dụng, tính kết hợp tốt khi sử dụng để giao tiếp mở cổng COM hay kết nối với các chân GPIO của Raspberry Pi 5. Về MVC thesis tuân thủ mô hình Model View Controller khi có Controller là chạy chính, có Service để kiểm tra tính Logic, có Model để làm việc với Database, có Repository để trỏ tới Model thao tác với Database. Database SQLite là 1 database dễ sử dụng tuy đã cũ nhưng nó ,…. Về front end HTML, CSS, JS là những cái cơ bản dễ tiếp cận với người sử dụng,….

a) Requirement analysis

b) Requirement analysis

Functional requirement

The following table outlines the functional requirements of the system, based on the roles of the two main actors: Admin and User.

ID	Functionality		Description	Roles
1	Maintain User Information	Sign Up	Register a new account	Admin, User
2		Password Recovery	Recover forgotten password	
3		Login	Log in to the system	

4		EditPersonal Information	Edit username, password, email, and phone number	
5	Maintain Dashboard	View Log & Fire Events	View daily logs and fire event history	
6		Real-time Data Monitoring	View live sensor data (temperature, humidity) on the dashboard	
7	Export logs		Export the data of the days	
			Export fire events	
8	User Management	Create new user	Add user accounts	Admin
		Delete exist user	Remove user accounts	
9		Approval Sign Up queue	Approval Sign Up queue	
10	Maintain Alert & Threshold	Alert Deactivation & Notification	Send system-wide alert if fire detected or alert is deactivated	
11		Threshold Settings	Modify temperature and humidity thresholds and notify all users	

Table 1: Table Functional requirement.

Non-functional requirements

- **Performance:**

The system must show sensor data (temperature and humidity) on the dashboard within 2 seconds after receiving it from the STM32 node.

- **Wireless Transmission Reliability:**

Since the system uses HC-12 wireless modules, it must detect and handle possible transmission errors to keep data accurate and complete.

- **Data Processing Efficiency:**

The process of data from node to central processing unit - encode and send sensor data than decode each message in under 1 second.

- **Data Security:**

All data sent wirelessly must be encrypted to prevent tampering. User passwords must be securely hashed using reliable algorithms.

- **Email Alert Delivery:**

The system must send email alerts reliably, with at least a 98% success rate when a fire is detected or thresholds are exceeded.

- **Scalability:**

The The system should handle multiple sensor nodes at the same time without slowing down or becoming unresponsive.

- **Cross-Platform Compatibility:**

The web application must work well on major browsers like Chrome, Firefox, Microsoft Edge, and others.

- **Reliability:**

Even though wireless transmission may sometimes fail, the system should remain reliable by using error checking or upgrading the wireless module if needed.

- **Usability:**

The system should be easy to use. Users should be able to view sensor data, receive alerts, and export logs without needing technical knowledge.

c) ***<u>System analysis and design</u>***

Use case diagram

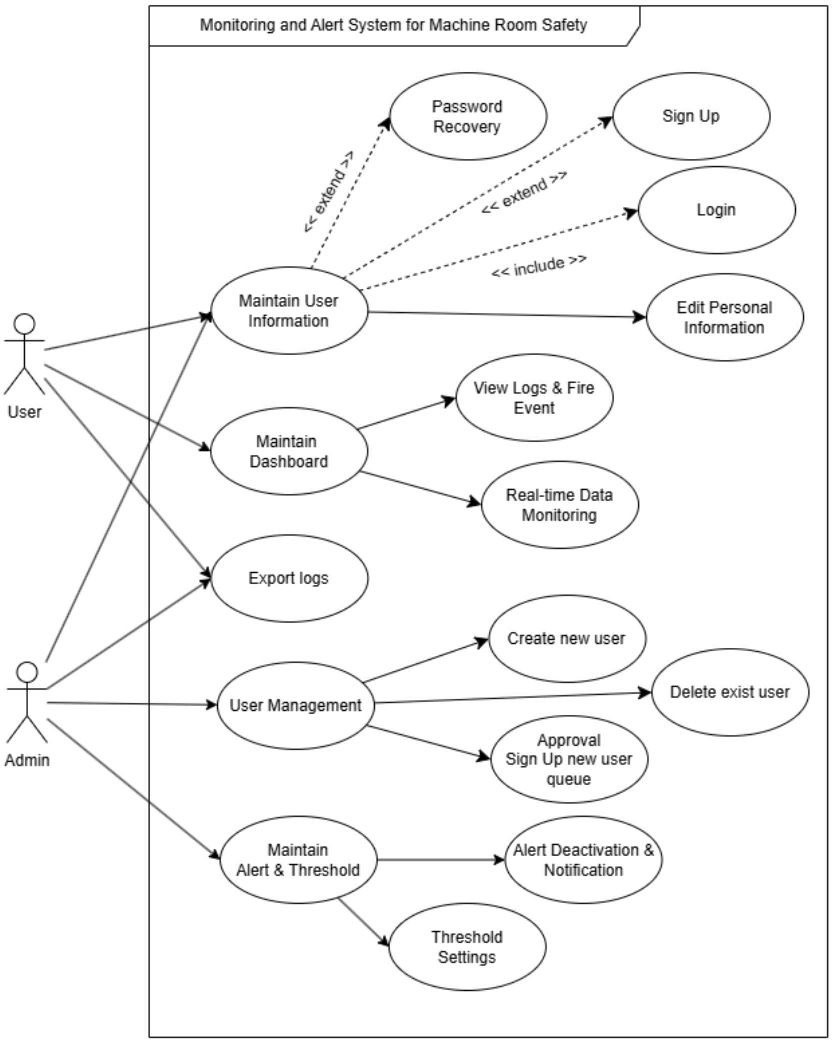

Monitoring and Alert System for Machine Room Safety
Password Recovery
Sign Up
<< extend >>
<< extend >>
Login
<< include >>
Maintain User Information
Edit Personal Information
User
View Logs & Fire Event
Maintain Dashboard
Real-time Data Monitoring
Export logs
Create new user
Admin
User Management
Delete exist user
Approval Sign Up new user queue
Maintain Alert & Threshold
Alert Deactivation & Notification
Threshold Settings

Figure 14: Use case diagram.

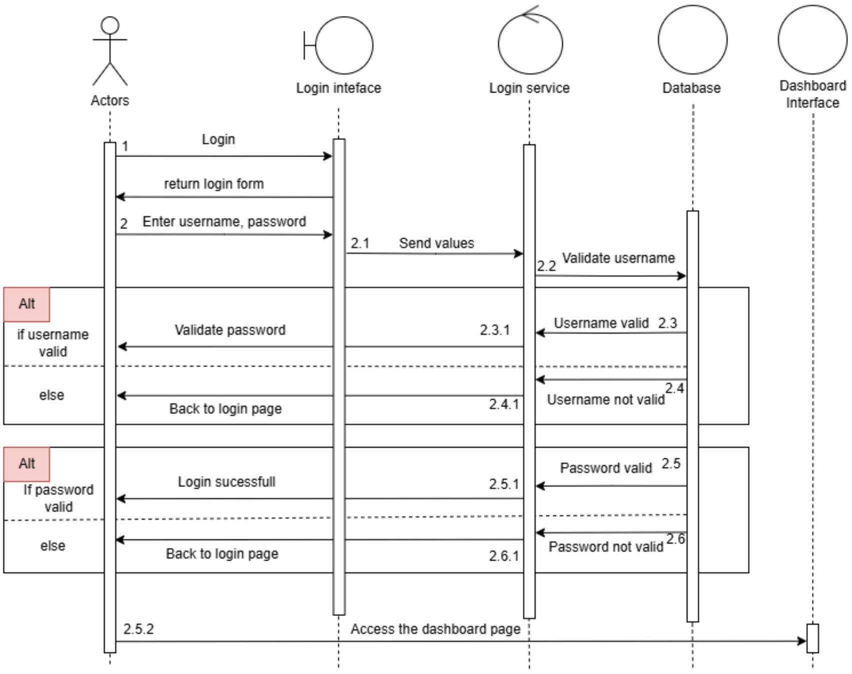
The use case diagram show two main actors of the system:

- User: A regular user who can sign up, log in, update personal information, monitor real-time data, receive alerts, and export logs.
- Admin: A superior user who manages accounts, approves sign-up user requests, sets alert thresholds, and sends system notifications.

The system’s core functionalities are grouped into Maintain User Information, Maintain Dashboard, Export logs, User management and Maintain Alert & Threshold.

Sequence Diagrams

Login

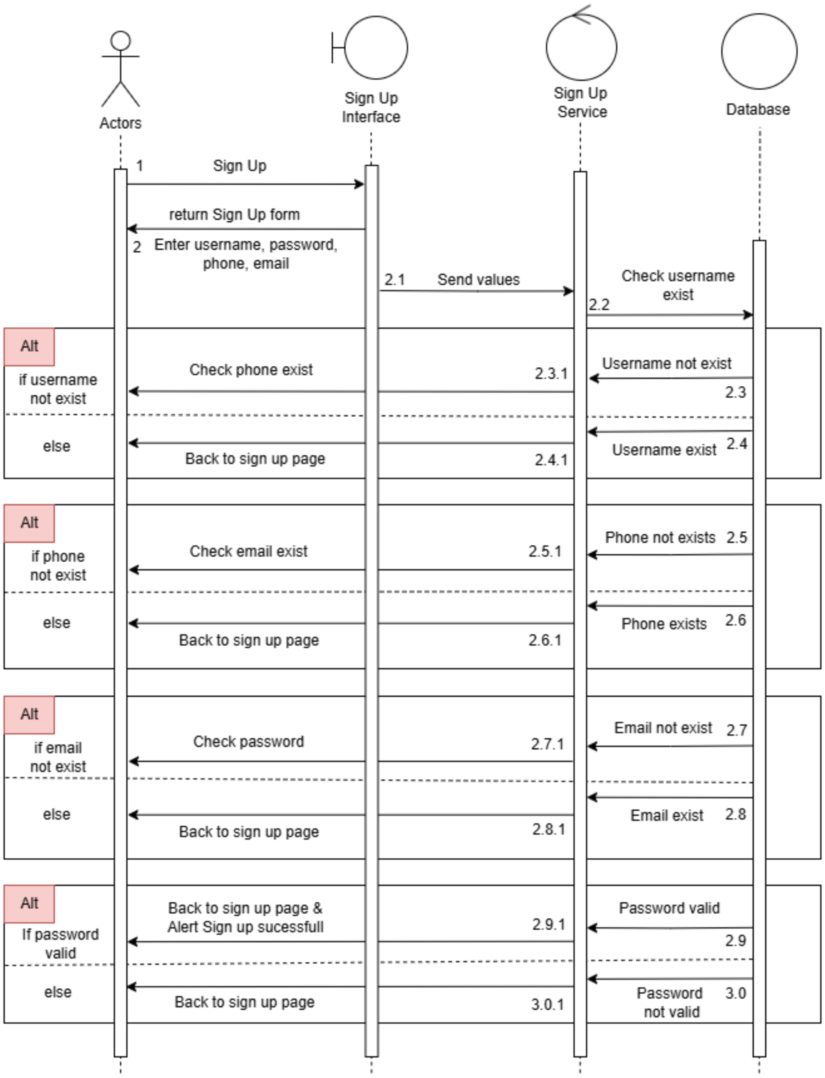


Appendix 1: Login Sequence Diagram.

When the user accesses the website, the server automatically redirects them to the Login Interface. At this interface, a login form is displayed with two input fields: *Username* and *Password*. Once the user fills in both fields, the request is handled by the Controller, which forwards the login information to the Authentication Service. The service first verifies the validity of the *Username*. If the *Username* exists, it proceeds to validate the *Password*. If the *Password* is incorrect, the service returns an alert message to the user and redirects them back to the login form. If the *Password* is correct, the user is successfully authenticated and redirected to the *Dashboard Page* for further interaction with the system.

This sequence ensures that only legitimate users receive access while providing feedback on incorrect login attempts.

Sign Up



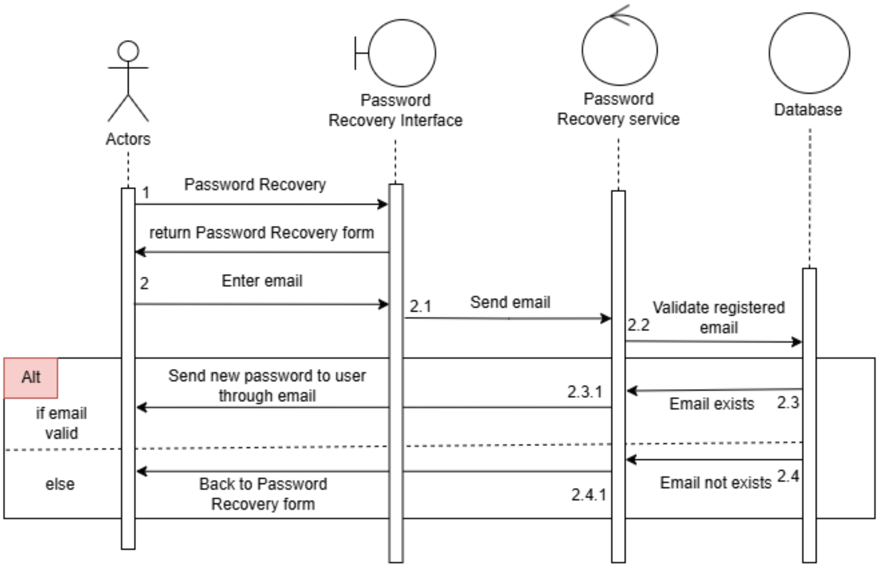
Appendix 2: Sign Up Sequence Diagram.

If the user does not have an account, they can select the Sign Up option. The Controller will then redirect them to the Sign Up page, where a registration form is displayed with four fields: Username, Password, Phone Number, and Email.

The user is required to fill in all four fields. Upon submission, the Controller forwards the input data to the Service layer for validation. The service performs multiple checks, including: Ensuring the phone number contains only numeric characters. Verifying the email follows a valid email format. Checking whether the entered username, phone number, or email already exists in the system, either in the current user database or in the sign-up queue.

If all inputs are valid and do not conflict with existing user information, the sign-up request is considered successful, and a confirmation alert is displayed to the user. If any input is invalid or already in use, error messages are returned and shown as alerts to the user in Sign Up page.

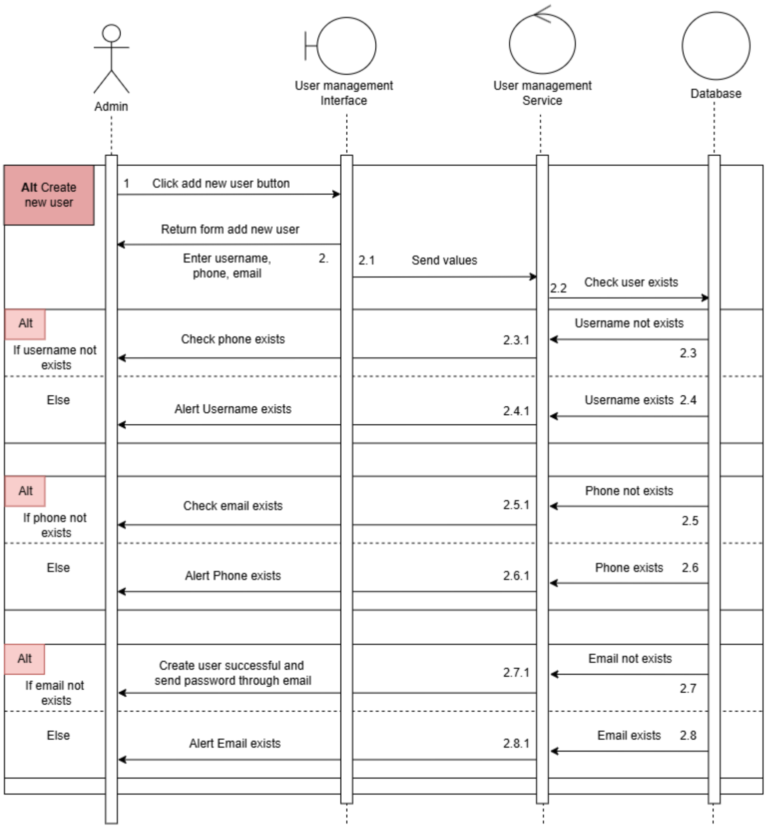
Password Recovery


Actors
Password Recovery Interface
Password Recovery service
Database
1
Password Recovery
return Password Recovery form
2
Enter email
2.1
Send email
2.2
Validate registered email
Alt
if email valid
Send new password to user through email
2.3.1
Email exists
2.3
else
Back to Password Recovery form
2.4.1
Email not exists
2.4

Appendix 3: Password Recovery Sequence Diagram.

This sequence diagram show the password recovery process in the system. The user enters their email to request a password reset. The system verifies the email and, if valid, generates a new password. The new password is then sent to the user's email via the mail service. If the entered email is invalid or not found, the system alert the user that email not exists.

Create new user

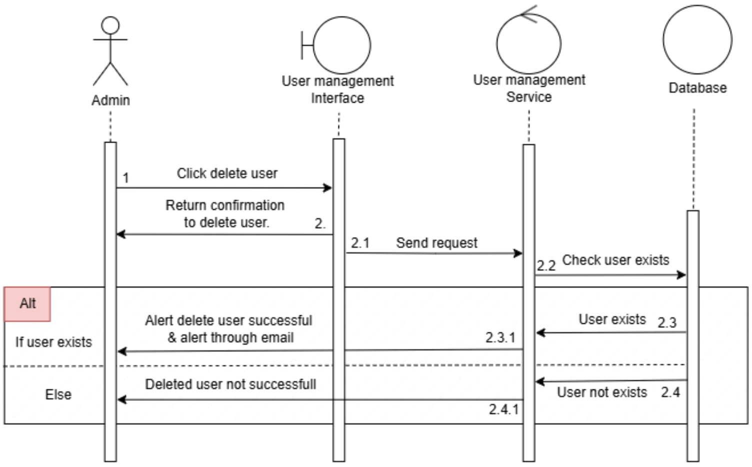


Appendix 4: Create New User Sequence Diagram.

This sequence diagram shows the process of an admin creating a new user. The admin fills in three fields: **username**, **phone number**, and **email**, then submits the form to the service for validation.

If all three fields are valid, the system generates a random password and sends it to the user’s email. If any of the fields are invalid, the system returns a specific alert message, as shown in the diagram.

Delete user



Appendix 5: Delete User Sequence Diagram.

This sequence diagram illustrates the process of an admin deleting a user. After selecting the user to delete, the service verifies whether the user exists. If the user exists, they will be deleted from the system, and a notification email will be sent to inform that the account has been removed. If the user does not exist, the system returns a failure alert indicating that the deletion could not be completed.

Threshold Settings

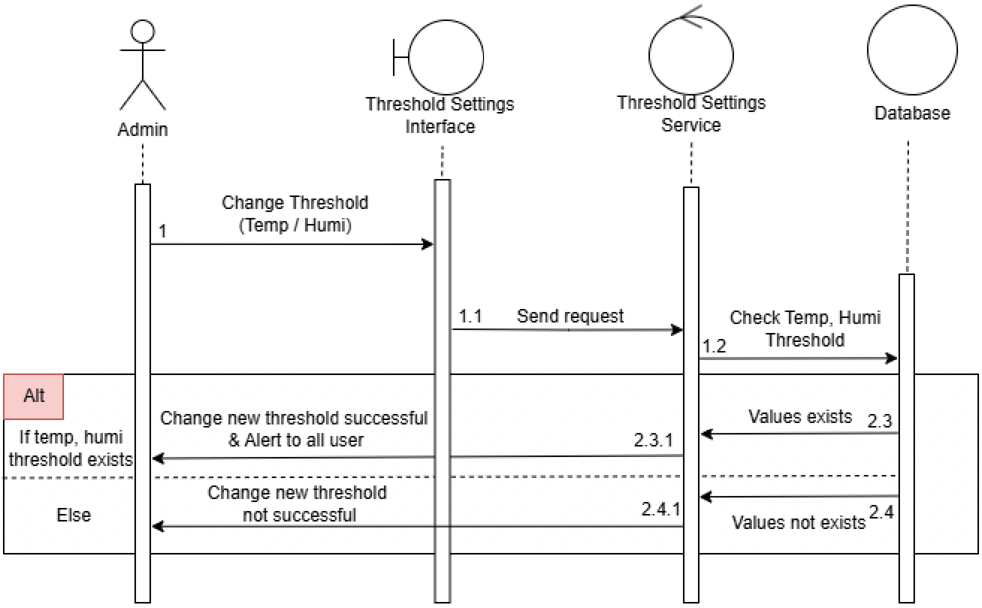

Admin
Threshold Settings Interface
Threshold Settings Service
Database
1 Change Threshold (Temp / Humi)
1.1 Send request
1.2 Check Temp, Humi Threshold
Alt
If temp, humi threshold exists
Change new threshold successful & Alert to all user
2.3.1
Values exists 2.3
Else
Change new threshold not successful
2.4.1
Values not exists 2.4

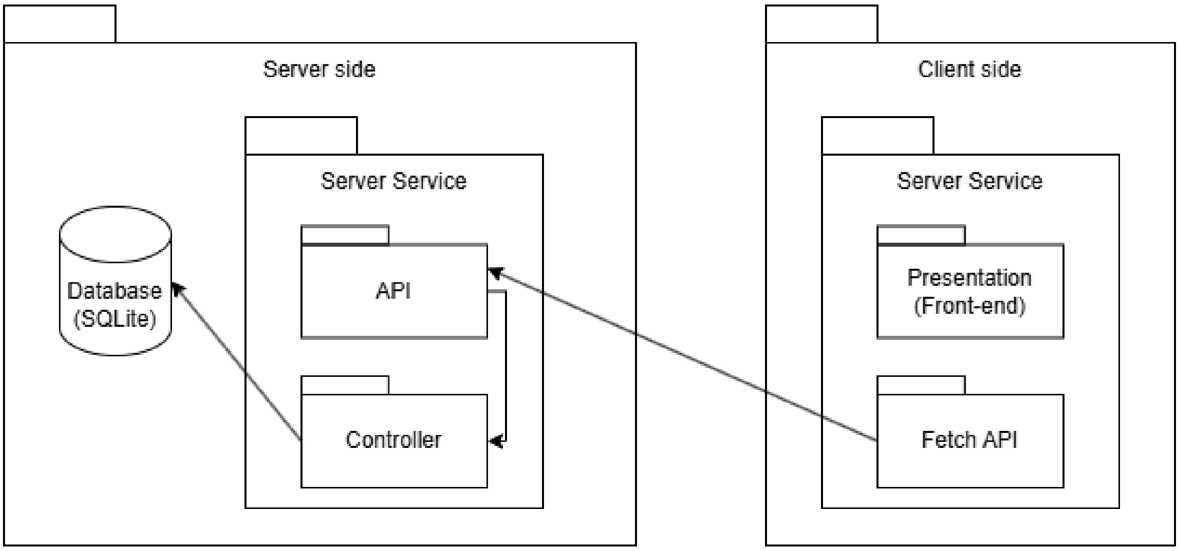
Figure 15: Threshold Setting Sequence Diagram.

This function allows the Admin to adjust the threshold values used for triggering alerts to users. The threshold includes two parameters: temperature and humidity.

Although the DHT11 sensor can theoretically return values up to 50°C and 20% humidity, using values too close to these limits may reduce the sensor's reliability and negatively impact the alert system. Therefore, in this thesis, the thresholds are intentionally set to 45°C for temperature and 40% for humidity to ensure both accurate alerts and the long-term safety of the sensor.

d) System Architecture

System Design



Appendix 6: System Architecture diagram.

Một số thành phần chính là một phần không thể thiếu đối với hoạt động liền mạch của hệ thống ở phía máy chủ:

Database: cơ sở dữ liệu đóng vai trò là kho lưu trữ, chứa dữ liệu quan trọng liên tục lieen quan đến thông tin người dùng, các log dữ liệu hàng ngày, event fire cảnh báo, các ngưỡng,…etc..

Dịch vụ máy chủ (Server Service): Thành phần này đóng vai trò trung gian giữa client và cơ sở dữ liệu, đảm nhận việc giao tiếp và xử lý các yêu cầu từ phía người dùng.

API: API là cổng kết nối giữa client và server thông qua các endpoint. Những endpoint này tiếp nhận và xử lý các yêu cầu từ phía client, thực hiện truy vấn dữ liệu từ cơ sở dữ liệu, xử lý logic và thực thi các chức năng cần thiết cho hệ thống.

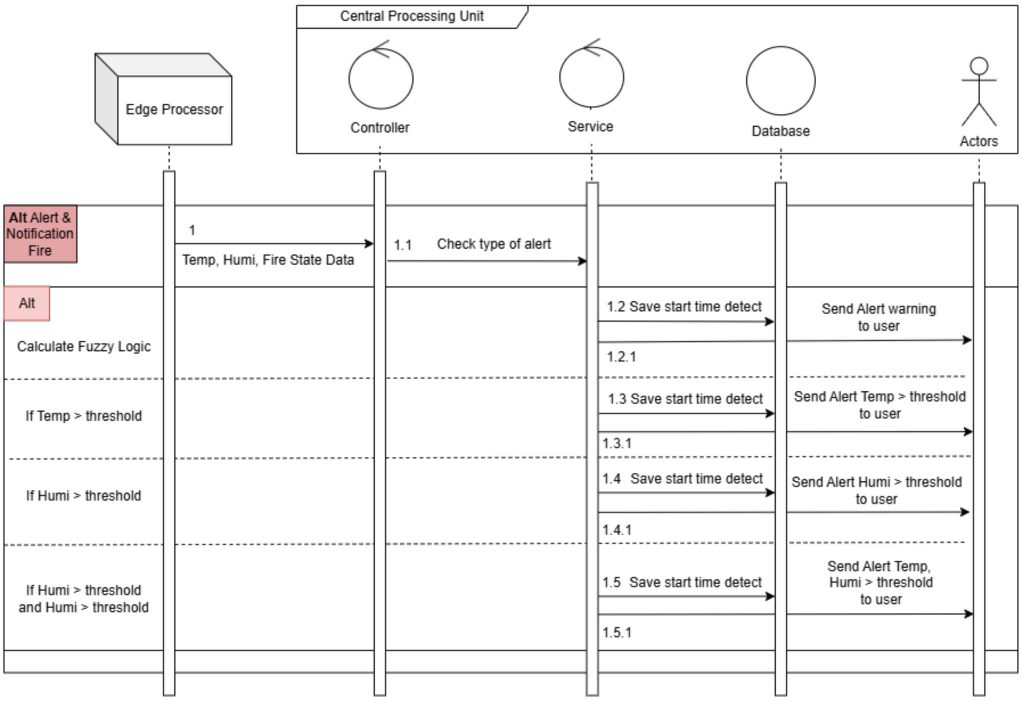
Cotroller: Controllers chịu trách nhiệm xử lý logic nghiệp vụ của ứng dụng. Chúng tương tác với API để truy xuất hoặc cập nhật dữ liệu từ cơ sở dữ liệu, quản lý luồng dữ liệu và thực thi các chức năng quan trọng đảm bảo hoạt động tổng thể của hệ thống.

Ở phía client (giao diện người dùng), lớp trình bày (presentation layer) được xây dựng bằng **HTML**, **CSS** và **JavaScript**:

- **Giao diện người dùng (User Interface):** Lớp này cho phép người dùng tương tác với hệ thống. Nó có khả năng gọi các API từ phía server để truy xuất hoặc cập nhật dữ liệu nhằm hiển thị cho người dùng.
- **Fetch API:** Công cụ này cho phép client lấy dữ liệu từ server một cách linh hoạt và cập nhật nội dung trên trang web mà không cần tải lại toàn bộ trang.

2.2.4. System Alert

This is an important part of the system that handles user notifications. This section contains warnings for Fuzzy Logic, Temperature Threshold, Humidity Threshold, and Both Temperature and Humidity Threshold.



Central Processing Unit
Edge Processor
Controller
Service
Database
Actors
Alt Alert & Notification Fire
1
Temp, Humi, Fire State Data
1.1 Check type of alert
Alt
Calculate Fuzzy Logic
1.2 Save start time detect
1.2.1
Send Alert warning to user
If Temp > threshold
1.3 Save start time detect
1.3.1
Send Alert Temp > threshold to user
If Humi > threshold
1.4 Save start time detect
1.4.1
Send Alert Humi > threshold to user
If Humi > threshold and Humi > threshold
1.5 Save start time detect
1.5.1
Send Alert Temp, Humi > threshold to user

Figure 16: Alert Sequence Diagram.

a) <u>Temperature & Humidity Threshold Alerts</u>

In this section, the system determines fire alarms based on the sensor's capabilities. The DHT11 sensor operates within a temperature range of 0-50°C and a humidity range of 20-90%. However, to prevent hardware damage and enable early fire detection, this thesis does not wait for the maximum temperature to activate alarms. Instead, an alert occurs when the temperature reaches 40°C and the humidity drops below 40%. In both cases, an email notification is sent to all users.

b) <u>Combined Threshold Alert for Temperature and Humidity</u>

When both conditions are occure together (temperature > 40°C and humidity < 40%), the system sends an alert to all users depending on a specific threshold. This combined threshold alert provides as an addition to the Fuzzy Logic-based fire detection system (which will be explained later). Although Fuzzy Logic may predict fire danger using predetermined rules, it may occasionally generate erroneous or imprecise findings. As a result, this dual-threshold alert mechanism acts as an extra precaution to improve the alert system's dependability in critical situations.

c) Fuzzy Logic

Fuzzy Logic [16] [17]is a data processing approach based on approximate reasoning, allowing values to range between 0 and 1, rather than being completely binary (true = 1 or false = 0) as in traditional logic systems. Instead of assigning a crisp value like "Cold" (0) or "Hot" (1), fuzzy logic introduces transitional states—for example, a value in between, such as "Warm," is represented by a degree of membership between 0 and 1.

As illustrated in the figure below, this flexible representation enables the system to evaluate the changes and generate risk levels ranging from low to high. Consequently, it allows for more nuanced alerts and earlier warnings, rather than waiting until a hard threshold is reached.

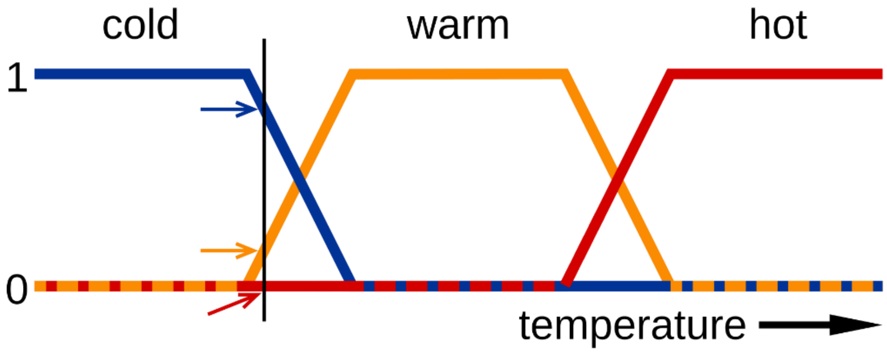


Figure 17: Fuzzy Membership Functions for Water State (Cold, Warm, Hot).

There are various types of Fuzzy Logic systems, including Rule-Based, as well as more advanced variants such as Learning-Based, Adaptive, and other forms like Fuzzy Weighted Sum.

However, this thesis adopts the Rule-Based Fuzzy Logic approach, which is also the most fundamental and representative form of fuzzy reasoning. The rule-based model consists of three main steps: Fuzzification – transform all input values into fuzzy membership functions.. Rule Evaluation – apply all applicable rules from the rule base to compute the fuzzy output sets. Defuzzification – convert the fuzzy output sets into crisp output values. This approach is chosen for its simplicity, interpretability, and effectiveness in modeling human-like reasoning in fire detection scenarios.

Fuzzification:

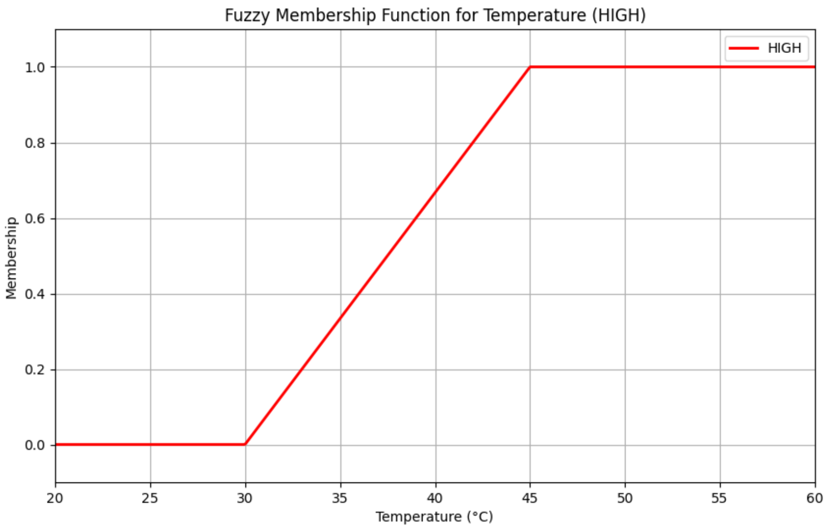


Figure 18: Membership of Temperature.

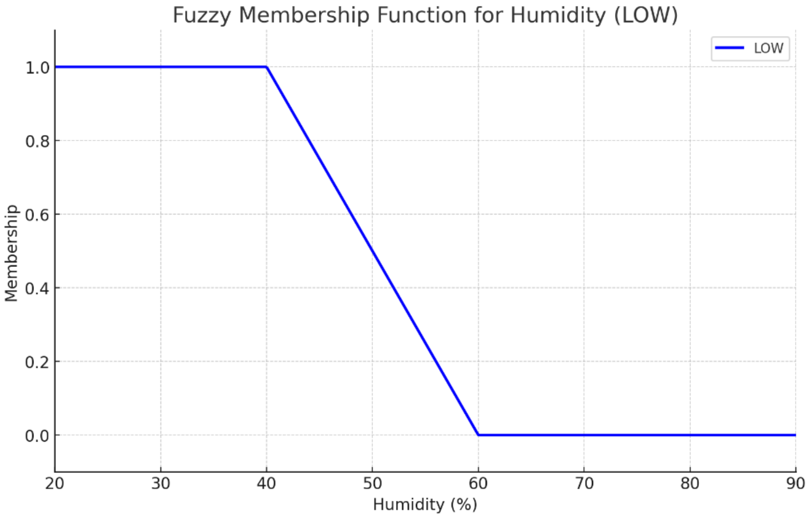


Figure 19: Membership of Humidity

To simplify the initial development process, this thesis only uses two major membership functions: Temperature High and Humidity Low. Limiting the model to these two levels, rather than adding levels such as Low, Medium, or High for each variable, helps to reduce system complexity and prevent rule explosion. Using a larger number of fuzzy rules adds complexity, potentially leading to inconsistencies and decreasedaccuracy. The system's primary purpose is to detect fires quickly and reliably. As a result, the model focuses on the most important and conditions commonly connected with the fire:

- High temperature: Fire make increase in ambient temperature. In this model, temperature is considered to gradually become “High” starting at 30°C, reaching “Fire” (1) at 45°C and above.
- Low humidity: A dry environment provides ideal conditions for fire to spread. So, humidity below 40% is considered an important risk, while levels above 60% are as safe.

This simplification enables a more efficient fuzzy rule base design without significantly compromising performance. Although the model does not account for intermediate states like Medium, it brings a high effective value to concentrate on detecting clearly high-risk fire conditions.

Rule Evaluation:

Rule	Fire status	Temperature	Humidity	Output Level
R1	Yes (1)	High (> 0.5)	Low (> 0.5)	VERY HIGH
R2	Yes (1)	High (> 0.5)	Not Low (≤ 0.5)	HIGH
R3	Yes (1)	Not High (≤ 0.5)	Low (> 0.5)	MEDIUM
R4	No (0)	High (> 0.5)	Low (> 0.5)	HIGH
R5	No (0)	Not High (≤ 0.5)	Not Low (≤ 0.5)	LOW

Figure 20: Rule of Temperature, Humidity, Fire status.

The fuzzy system uses 5 rules to evaluate fire risk based on temperature, humidity, and fire detection status. Each rule is triggered according to the degree of membership in “high temperature” and “low humidity”. Rules with fire detected (fire state = 1) prefer higher alert levels, especially when both temperature are high and humidity are low. Rules without fire serve as early warnings based on bad environmental conditions. Each rule contributes a weighted score based on severity, which is averaged to produce the final fuzzy risk level. This approach ensures flexible and realistic decision-making beyond fixed thresholds.

Defuzzification:

After evaluating all activated fuzzy rules, the system applies a weighted average method to perform defuzzification. Each output label (LOW, MEDIUM, HIGH, VERY HIGH) is assigned a numeric weight reflecting its severity level (3, 5, 7, 9 respectively). The final fuzzy score is calculated by averaging these weights based on rule strength.

This crisp score is then used to analyze the fire risk level:

- A score above 7 consider a high probability of fire, triggering an immediate fire alert.
- A score between 4 and 7 signals a potential fire risk, prompting early warnings.
- A score below or equal to 4 suggests the environment is safe.

This approach ensures that the system reacts appropriately and equally to environmental conditions, enhancing its reliability in early fire detection scenarios.

e) ***System Architecture***

System design

Tool & Technical choices

Back-End

Front-End

2.3. Conclusions

In this

architecture offers a promising pipeline for efficient and accurate human detection for real-world applications.

Chapter 3. Implementation and Evaluations

In the previous

3.1. Experiment setup environment

3.2. Conclusions

Conclusions and Perspective

References

[1] Pradeep, Aneesh; Latifov, Akmaljon; Yodgorov, Ahborkhuja; Mahkamjonkhojizoda, Nurislombek, "Hazard Detection using custom ESP32 Microcontroller and LoRa," *IEEE,* no. 26 May 2023, 2023.

[2] N. I. M. E. N. M. Z. a. K. S. S. K. M. N. N N Mahzan, "Design of an Arduino-based home fire alarm system with GSM module," *Journal of Physics: Conference Series,* vol. 1019, 2917.

[3] M. S. B. Bahrudin, "Development of Fire Alarm System using Raspberry Pi and Arduino Uno," *2013 International Conference on Electrical, Electronics and System Engineering (ICEESE),* 2013.

[4] N. I. M. E. N. M. Z. a. K. S. S. K. M. N. N N Mahzan, "Design of an Arduino-based home fire alarm system with GSM module," *Journal of Physics: Conference Series,* vol. 1019, 2017.

[5] H. Y. ·. J. L. ·. L. L. ·. Z. J. ·. Y. L. ·. D. Zhou, "Development of Four Rotor Fire Extinguishing System for Synchronized Monitoring of Air and Ground for Fire Fighting," *ICIRA,* p. 267–278, 2019.

[6] N. Komalapati, V. C. Yarra, L. A. Vyas, Kancharla and T. N. Shankar, "Smart Fire Detection and Surveillance System Using IOT," *IEEE,* 2021.

[7] A. E. H. 1. A. S. S. 1. K. 2. B. A. 1. a. I. C. y Kuldoshbay Avazov 1, "Forest Fire Detection and Notification Method Based on AI and IoT Approaches," *Future Internet (ISSN: 1999 - 5903),* vol. 15, no. 2, 2023.

[8] L. WHITE and R. AJAX, "Improved Fire Detection and Alarm Systems.," 2025.

[9] "AD Store," [Online]. Available: https://store.arrowdot.io/product/ky-026-flame-sensor-module-detects-infrared-light-emitted-2/.

[10] "Circuit Rocks," [Online]. Available: https://circuit.rocks/products/sensors-temperature-and-humidity-dht11-sensor-html.

[11] "ePanorama.net," [Online]. Available: https://www.epanorama.net/blog/2022/06/18/introduction-to-the-stm32-blue-pill-stm32duino-and-other-stm32-boards/.

[12] "IC Master," [Online]. Available: https://icmasteronline.com/product/433mhz-hc12-wireless-serial-port/.

[13] "Linh Kien 888," [Online]. Available: https://linhkien888.vn/anten-433mhz-chong-nuoc-5dbi-dau-sma-male.

[14] "The Register," [Online]. Available: https://www.theregister.com/2023/09/28/raspberry_pi_5_revealed/?td=amp-keepreading.

[15] "Amazon," [Online]. Available: https://www.amazon.sg/ASUS-ZenScreen-MB16AC-Portable-Monitor/dp/B0BZR6ZNB5?th=1.

[16] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Fuzzy_logic.

[17] P. Bolourchi and S. Uysal, "Forest Fire Detection in Wireless Sensor Network Using Fuzzy Logic," *IEEE,* 2013 Fifth International Conference on Computational Intelligence, Communication Systems and Networks.

[18] A. Suleiman, "An energy-efficient hardware implementation of HOG-based object detection at 1080HD 60 fps with multi-scale support.," 2016.

[19] Dalal, N., & Triggs, B., "Histograms of oriented gradients for human detection," *IEEE computer society conference on computer vision and pattern recognition,* 2005.

[20] Nguyen, N. D., Bui, D. H., & Tran, X. T., "A novel hardware architecture for human detection using HOG-SVM co-optimization," *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05).,* vol. 1, pp.

886-893, 2019.

[21] Nguyen, N. S., Bui, D. H., & Tran, X. T., "Reducing temporal redundancy in MJPEG using Zipfian estimation techniques," *2014 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS). IEEE,* pp. 65-68, 2014.

[22] Richefeu, A. M. J., "A robust and computationally efficient motion detection algorithm based on σ-δ background estimation," *Indian Conference on Computer Vision, Graphics & amp; Image Processing,* vol. 9, pp. 16-18, 2004.

[23] A. Manzanera, "σ-δ background subtraction and the Zipf law," *Progress in Pattern Recognition, Image Analysis and Applications: 12th Iberoamericann Congress on Pattern Recognition, CIARP 2007,* pp. 42-51, 2007.

[24] Lacassagne, L., Manzanera, A., & Dupret, A, "Motion detection: Fast and robust algorithms for embedded systems," *IEEE international conference on image processing (ICIP),* pp. 3265-3268, 2009.

[25] Ho, H. H., Nguyen, N. S., Bui, D. H., & Tran, X. T, "Accurate and low complex cell histogram generation by bypass the gradient of pixel computation," *4th NAFOSTED Conference on Information and Computer Science,* pp. 201-206, 2017.

[26] Dollár, P., Belongie, S., & Perona, P., "The fastest pedestrian detector in the west," 2010.

[27] Dollar, P., Wojek, C., Schiele, B., & Perona, P., "Pedestrian detection: An evaluation of the state of the art," *IEEE transactions on pattern analysis and machine intelligence, 34(4),* pp. 743-761, 2011.

[28] Cristianini, N., & Shawe-Taylor, J., An introduction to support vector machines and other kernel-based learning methods, Cambridge university press, 2000.

[29] Nguyen, T. A., Tran-Thi, T. Q., Bui, D. H., & Tran, X. T, "FPGA-Based Human Detection System using HOG-SVM Algorithm," *International Conference on*

Advanced Technologies for Communications (ATC), pp. 72-77, 2023.

[30] Ferryman, J. & Shahrokni, A., "PETS2009: Dataset and challenge.," *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance,* 2009.

[31] Andriluka, M., Roth, S. & Schiele, B., "People-Tracking-by-Detection and People-Detection-by-Tracking," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 2008.

[32] Andriluka, M., Roth, S. & Schiele, B. , "Monocular 3D Pose Estimation and Tracking by Detection," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 2010.

[33] [Online]. Available: https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/.

[34] [Online]. Available: https://www.v7labs.com/blog/mean-average-precision.

[35] T. Panda, R. Banerjee, A. Pal, S. K. Bishnu and A. Chakraborty, "IOT-Based Home Automation for LPG Gas and Fire Detection System With Automated Safety Measures," *IEEE,* no. 16 April 2025, 2025.

[36] Shanghai Xian Dai Architecture,Engineering&Consulting Co. ,China , "Information fusion technology based on wireless fire detection and alarm system," *Journal of Physics: Conference Series,* pp. 883-887, 21-11-2013.

[37] "ManualMachine," Honeywell Home, [Online]. Available: https://digitalassets.resideo.com/damroot/Original/10002/L_5809SSD_D.pdf.

[38] "spntelecom," [Online]. Available: https://spntelecom.vn/cam-bien-nhiet-pisafe-dau-bao-nhiet-khong-day-phong-chay-canh-bao-chay-thong-minh-wifi-dat-tieu-chuan-pccc-phat-hien-chay-no-tu-xa-qua-dien-thoai.

[39] [Online]. Available: https://www.socketxp.com/iot-remote-monitoring.

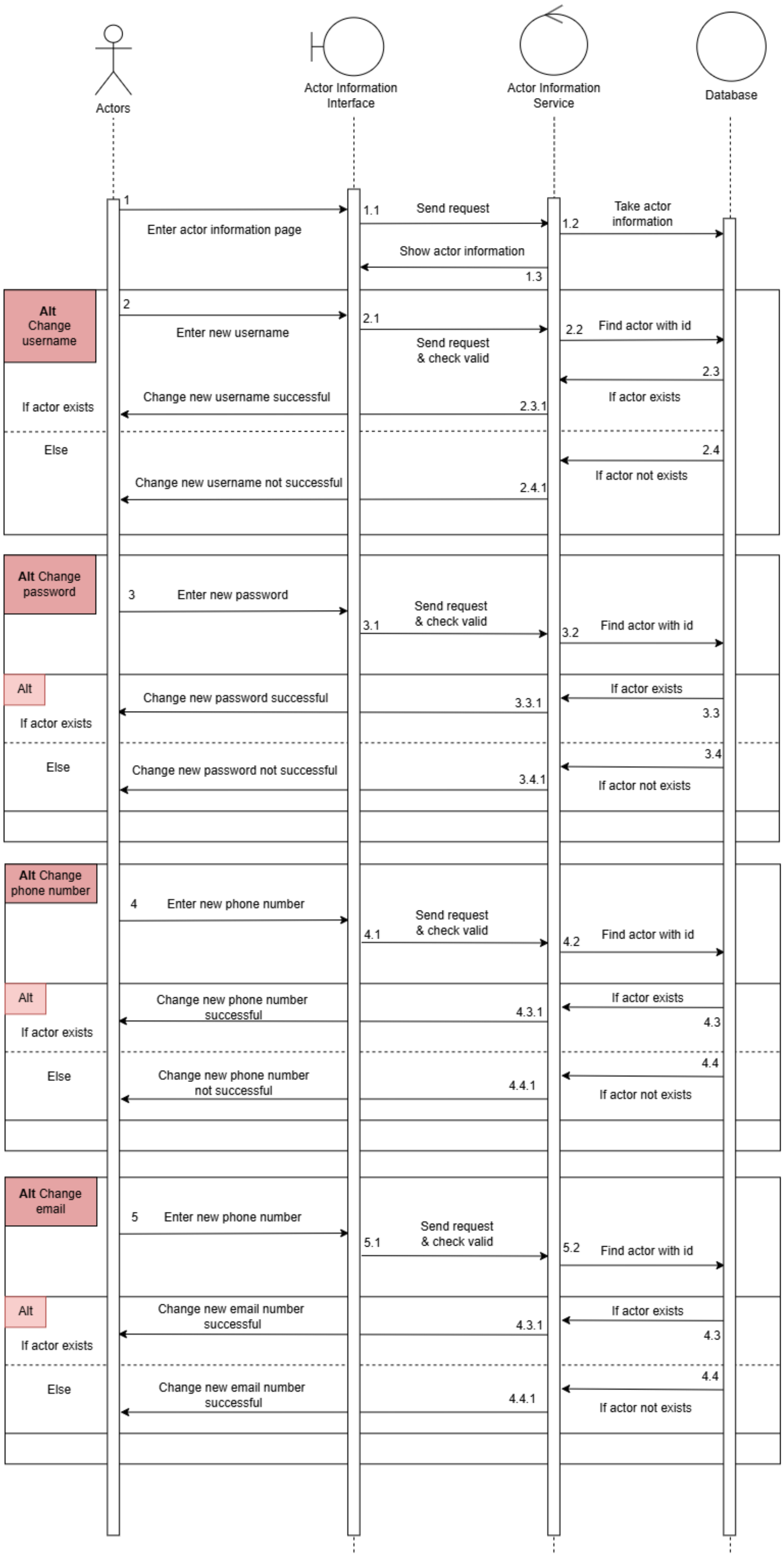
[40] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Low-density_parity-check_code.

[41] "Wekapedia," [Online]. Available: https://en.wikipedia.org/wiki/Linear-feedback_shift_register.

[42] "HShop," [Online]. Available: https://hshop.vn/mach-thu-phat-rf-uart-si4463-433mhzkhoang-coch-1km.

Appendix

List of Figure Appendix:



Appendix 7: Edit User Information Diagram.