SYSC3010 Computer Systems Development Project
FANS Project Proposal
Group L1-G8

Grant Achuzia, 101222695
Javeria Sohail, 101197163
Matteo Golin, 101220709
Saja Fawagreh, 101217326
TA: Sean Kirkby

Created: February 11th, 2024
Modified: February 11, 2024

Contents

1 Introduction

This document presents the proposal for FANS (Fire Alarm Notification System), a comprehensive solution addressing the alarming rate of preventable fire deaths in Canada. The L1-G8 group has developed FANS as a multi-system fire alarm device with advanced features to significantly reduce fire deaths through timely detection and response to fire emergencies.

The proposal reveals the detailed system design and comprehensive work plan for the development and implementation of FANS to reduce the number of preventable fire-related deaths in Canada. It outlines the system's architecture and provides a structured work plan detailing the phases of development, testing and deployment to ensure the timely deployment of FANS to effectively mitigate fire disasters.

1.1 Background

The background for the FANS (Fire Alarm Notification System) project outlines the limitations of traditional fire alarm systems, which often lack modern communication features and real-time monitoring capabilities. These limitations contribute to delays in response time and can lead to preventable fire-related deaths. The FANS project aims to address these shortcomings by introducing a comprehensive solution that includes multi-system fire detection, real-time notifications via SMS and email, and configurable thresholds for proactive emergency response. By leveraging technological advancements, FANS seeks to improve the effectiveness of fire alarm systems and reduce fire-related fatalities in Canada.

1.2 Motivation

The motivation for the FANS project is to address the critical problem of preventable fire-related deaths in Canada. Fire safety is a fundamental concern for individuals, families, and communities nationwide. By leveraging modern technological advancements, the FANS system aims to significantly enhance the effectiveness of fire alarm systems, reducing response times, and ultimately saving lives. The importance of this project cannot be overstated, as it directly impacts the safety and well-being of Canadians.

1.3 Project Objective

The overarching objective of the FANS project is to develop and deploy a functional fire detection and notification system that enhances fire safety. FANS accomplishes this by providing real-time fire detection, timely notifications to users, and proactive monitoring of environmental conditions.

1.4 Specific Goals

The goals that the FANS development team aims to achieve in this project are as follows:

1. Develop and test a smoke detection algorithm on Raspberry Pi to ensure accurate and reliable fire detection.
2. Integrate various sensors with Raspberry Pi for real-time data collection, including temperature and smoke levels.
3. Implement a robust notification system capable of triggering alarms at installation sites and sending email alerts to users in affected areas.

4. Create a user-friendly web interface for monitoring and interaction, allowing users to access real-time data, configure system settings, and receive alerts.
5. Establish a secure and efficient database for storing employee contact information and system data, ensuring the availability of necessary information for emergency response.

1.4.1 Functional Requirements

1. Consistent accuracy from the smoke sensors to avoid any false alarms/triggers.
2. Real-time data collection and monitoring of temperature and smoke levels.
3. Timely and reliable email notifications to users during fire emergencies.
4. User-friendly web interface with intuitive design and real-time updates.
5. Database with secure storage and retrieval of employee contact information and system data.

1.4.2 Non-Functional Requirements

1. High system reliability and availability to ensure immediate response during emergencies.
2. Scalability to accommodate a growing number of users and installation sites.
3. Data security and privacy measures to protect sensitive information.

2 System Design

2.1 System Overview Diagram

Figure 1 shows the deployment diagram for FANS. There will be three devices which communicate with each other on a local network: the alarm system on a Raspberry Pi 4, a smoke detection and temperature logging system on a second Raspberry Pi 4, and a notification system on a third Raspberry Pi 4. When the smoke detection system detects smoke above the limiting threshold, it will signal the alarm system to sound the alarm. It will signal the notification system to send out notifications to the affected persons. The smoke detection and temperature logging system will also update the cloud database (hosted by Firebase) with the recorded smoke and temperature levels, and enable a flag that indicates the emergency. The fourth device, the Pi Pico, will buzz when the cloud database indicates that an emergency has been detected by the detection system. This will notify the user of the emergency.

The notification and alarm system are on separate devices to preserve their integrity in the case of an emergency. It is possible that the smoke detection system will be closer to the source of the fire, and therefore be at risk of being damaged/disabled. With the notification and alarm systems on separate devices connected through a local internet connection, they will be able to perform their responsibilities with less risk of being damaged by flames.

The cloud database is a real-time database to facilitate the real-time nature of display information on the web GUI. In addition, it will allow the haptic feedback device to buzz as soon as the smoke detection system signals an emergency to the cloud database.

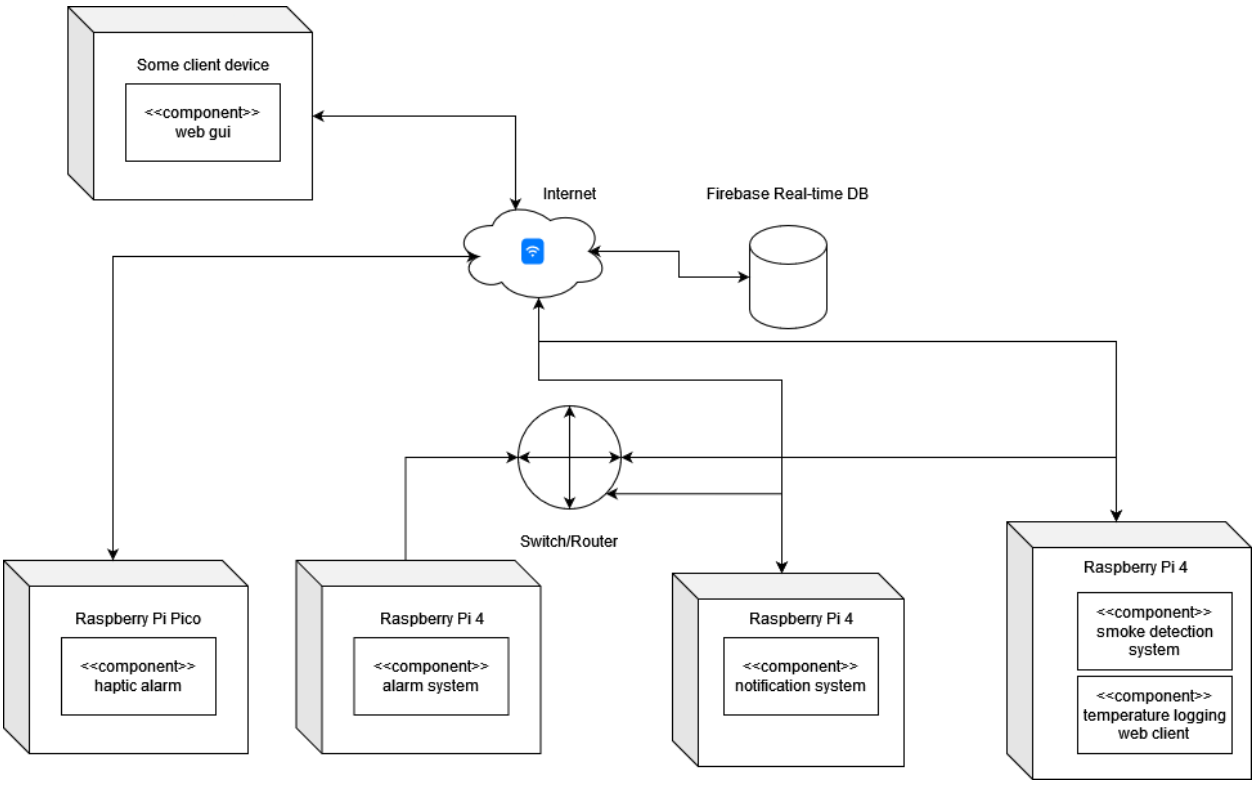

Figure 1: Deployment diagram of FANS.

The web GUI is displayed as being on the client device, because it will be accessible by any device with an internet connection. It will be served on the cloud.

This configuration would allow us to scale our system horizontally very easily. Multiple different alarm systems and smoke detectors could be connected over a local network to a notification system, allowing smoke detection in multiple rooms and alarms sounding in multiple rooms. Additionally, several haptic alarm wearables could be clients to the real-time database, allowing any number of users to be notified haptically in the event of an emergency.

Each subsystem has a single responsibility that allows for modularity in FANS. The alarm system simply waits to receive a notification to sound its alarm, and then does so. The smoke detection and temperature sensing system periodically writes sensor data to the real-time database, and also signals the database and the local notification system when it detects a smoke or temperature measurement above the emergency threshold. The notification system is responsible for receiving notice from the smoke detection system of an emergency, and then signalling the alarm system and sending SMS/emails to all affected users in the database. The wearable haptic alarm system simply waits to be signalled by the real-time database. Each subsystem has a clearly defined interface with specific messages passed between them, allowing subsystems to operate without knowledge of each others' implementations.

2.1.1 Communication Protocols

The smoke detection system will be communicating with an array of temperature and smoke sensors using the I2C protocol over the GPIO pins of a Raspberry Pi 4.

Each node on the local network (the smoke detection system, alarm system and notification system) will communicate with each other using UDP packets over the local network.

Nodes which communicate with the cloud hosted real-time Firebase database will use HTTP requests to interact with the database. The smoke detection system will post JSON payloads to the database to write data. The web GUI will request JSON over HTTP to update its visual components (dashboard with charts, etc.) with the data stored in the database. The notification system will request contact information from the database over HTTP to send notifications to affected users.

Finally, the notification system will communicate with affected users over email and SMS text notifications.

2.2 Component Details

The following section will describe the interfaces and responsibilities of each major node in the FANS system.

2.2.1 Smoke Detection System

The first component in the proposed system is the smoke detection and temperature sensing system. It will receive sensor data using the I2C protocol from a locally connected smoke sensor and temperature sensor. It will then update the real-time Firebase database with the collected sensor data. If any temperature or smoke measurements are above the critical threshold for signalling an emergency, it will send a message to the notification system over TCP to signify an emergency.

2.2.2 Notification System

The notification system is responsible for notifying affected users of an emergency and triggering the alarm system. When it receives an emergency notification over TCP from the smoke detection system, it will forward the message on to the alarm system. It will also send SMS and email notifications to all users in the database to notify them of the detected emergency.

2.2.3 Alarm System

The alarm system is responsible for sounding an alarm in the building when an emergency has been detected. The alarm system receives notification over TCP from the notification system when an emergency has been detected, at which point it sounds an alarm and flashes lights. The lights will be an onboard SenseHat, and the alarm is still to be decided. The alarm will continue to sound until the system receives a TCP message to stop.

2.2.4 Haptic Alarm System

The haptic alarm system is designed to be a wearable device that vibrates during an emergency. The device queries the real-time database for a flag indicating an alarm. Once the flag is raised, the device vibrates until the emergency has ended (the flag is "lowered").

2.2.5 Firebase Real-time Database

The real-time database is responsible for storing smoke and temperature level data in real-time, a flag indicating emergency and user contact information. It will also contain user-configured settings (such as smoke level threshold for triggering an emergency). It will be cloud hosted by Firebase.

2.2.6 Web GUI

The web GUI is responsible for displaying the sensor information from the real-time database. It will also allow users to configure settings for FANS (emergency thresholds, etc.) and add more user contact information in the case of an emergency. It will also allow users to disable the alarm when an emergency has been resolved.

2.3 Use Cases

2.3.1 Trigger Emergency

Flow of events

1. The smoke detection system sends a TCP message to the notification system to warn of an emergency.
2. The smoke detection system raises a flag in the real-time database to indicate the emergency.
3. The notification system forwards this message to the alarm system.
4. The notification system begins sending SMS and email notifications to the users who are listed in the database.
5. The alarm system receives the TCP message from the notification system.

6. The alarm system sounds the alarm and flashes LEDs.
7. The haptic alarm system reads the flag from the real-time database.
8. The haptic alarm system begins buzzing.

3 Work Plan

The following section describes the approach taken by the FANS development team to collaborate effectively and complete the FANS system before its deadline.

3.1 The Project Team

Each member of the project team is a third year computer systems engineering student.

Grant Achuzia
Grant's strengths lie in effective project management and organization, and he is passionate about delivering successful projects in academic settings. Grant also has significant experience designing web interfaces that are accessible and aesthetically pleasing.

Matteo Golin
Matteo has background in developing embedded system on Raspberry Pi's from his work at Carleton University's rocketry design team, where he is leading the development of a real-time telemetry system using Blackberry's QNX RTOS.

Saja Fawagreh
Saja has developed skills in user interface design from her time on co-op, and excels and making visually appealing interfaces because of her sharp attention to detail.

Javeria Sohail
Javeria has expertise in system architecture design, and has a strong knowledge of object-oriented design patterns. She is passionate about optimization, efficient design and modularity.

3.1.1 Roles and Tasks

Team Member	**Major Role/Task (Primary)**	**Secondary Responsibility (Testing)**	**Reason for assignment**
Grant Achuzia	Project coordination	Organize the entire system (software and hardware)	Strong project management and organizational skills, ensuring academic milestones are met on time.
Matteo Golin	Software development	Testing and debugging	Skilled in programming and problem-solving, enthusiastic about tackling coding challenges.
Saja Fawagreh	UI design	Front-end testing and usability	Proficient in front-end development, dedicated to creating user-friendly interfaces.
Javeria Sohail	Project architecture	hardware integration	Familiar with system architecture and design for hardware components, ensuring seamless hardware functionality within FANS.

Table 1: Assignment of team members to major project tasks

3.1.2 Teamwork Strategy

Regular Team Meetings
We will schedule regular team meetings to discuss project progress, share updates, and address any challenges or questions. Our team will establish MS Teams as our main communication channel to ensure efficient communication within the team and with the TA. Additionally, the team meetings will be documented to review what we talked about in case someone missed the meeting.

Version Control
We will use GitHub to manage codebase changes. This includes practices such as creating branches for features or bug fixes, committing regularly, and using pull requests for code reviews. Informative commit messages will be prioritized to avoid misunderstandings.

Code Reviews
Before merging code changes, thorough code reviews will be conducted to ensure code quality and maintainability.

Testing
We will follow a testing strategy that includes unit testing, integration testing, and system testing to ensure the reliability and robustness of the FANS.

Feedback
Team members will provide constructive feedback to each other for continuous improvement and learning (also making use of FBF). We also want to take constant feedback from our TA at every milestone so that our big picture comes together seamlessly.

3.1.3 What We Will Need to Learn

Hardware integration
Since the FANS includes both hardware and software components, we need to learn how to integrate them seamlessly. We utilize online resources and forums where hardware integration techniques are discussed.

Real-time communication
Implementing real-time communication features such as SMS and email notifications requires knowledge of APIs and protocols for sending messages. We will look at the documentation and instructions provided by communication service providers (e.g. SMS) and learn how to integrate these services into our system.

Developing a web interface
Creating a user-friendly web interface for the FANS requires knowledge of front-end development, including HTML, CSS and JavaScript. We will use online tutorials, courses and documentation of web development frameworks such as React or Angular to improve our web development skills.

Test strategies
Developing effective testing strategies, including unit testing, integration testing, and system testing, requires an understanding of various testing frameworks and methodologies. We will explore tutorials, documentation, and best practices for testing in software development.

3.2 Project Milestones

Milestone Name	Date	Description
UI	2024-02-13	Develop a user-friendly web interface for monitoring system status, configuring thresholds, and interacting with the system.
Database	2024-02-17	Set up a database to store employee contact information and system data, ensuring efficient management and retrieval.
Smoke detection	2024-02-20	Develop and test the smoke detection algorithm on the Raspberry Pi to ensure accurate detection of fire and smoke.
Sensor integration	2024-03-02	Integrate sensors with the Raspberry Pi for real-time data collection, enabling the system to monitor environmental conditions.
Notifications	2024-02-12	Implement a notification system to trigger alarms and alert users via SMS and email in case of fire emergencies.
Testing	2024-02-20	Conduct comprehensive testing to ensure the functionality and integration of both hardware and software components.
Finalization	2024-02-27	Finalize system configuration, including thresholds and notification settings, and conduct comprehensive testing.

Table 2: Major milestones for FANS development.

3.3 Schedule of Activities

Under construction!

4 Project Requirements Checklist

Computer Components

- 1 Raspberry Pi per each of the following: Smoke Detector device, Smoke Alarm, Wearable Accessory
- 1 Raspberry Pi will be used to transfer info to and from all the other RPis to the Web GUI (running in headless mode). There are 4 Raspberry Pi's used in the project overall.

Hardware Components

- MQ2 Smoke Sensor (x1), LCD Display (x2), AudioHat Module (x1), SenseHat Module (x1)
- Actuator: LCD Display, AudioHat Module
- Smoke sensor waits till it senses smoke, once that requirement is met it triggers the alarm, wearable accessory, and sends a notification to the Web GUI

Software Components

- Database for employee contact information.
- The computer hosting the database changes depending on user configurations (Level of smoke to detect, Web GUI themes, alarm sounds).
- Database includes a table for user info (employee names, phone numbers, emergency numbers) and another table for sensor data (temperature data, smoke detection).
- Periodic timing loop is given by the smoke sensor monitoring the environment for smoke. The smoker sensor will log this data and send it to the web GUI every minute.
- The data the smoke detector logs will be displayed on the user website in the form of varying graphs and widgets.
- Smart watch will have a buzzer, Raspberry Pi with the alarm will have notification software. Raspberry Pi will send notifications to the Smart watch and to the web GUI.
- Bidirectional GUI included (user can reset the smoke detector from the web GUI, and changes made to the physical detector will show in Web GUI settings)

5 Additional Hardware Required

Developing FANS requires additional hardware that is described in the following tables. Table 3 lists hardware that is already provided by Carleton University, while Table 4 shows any additional components that will need to be purchased by the team.

Component	Amount	Digi-Key Number
Wires	50-100 pieces	N/A
SenseHat Module	4	N/A
Breadboard	4	N/A

Table 3: Additional hardware provided by Carleton University.

Component	Amount	Digi-Key Number
AudioHat Module	1	N/A
Smoke Sensor	1	333102
16x4 LCD Display	2	EA DOGS164W-A
Raspberry Pi Zero W	1	SC0065

Table 4: Additional hardware to be bought by the team.

A Something

This is a placeholder appendix.