University of Padova

Department of Information Engineering

Master Thesis in ICT for Internet and Multimedia

Proof-of-Concept of a Fall Detection System Based on Low-Cost IoT Devices

Supervisor
Prof. Andrea Zanella
University of Padova

Master Candidate
Monisha Vardharajan

Student ID
2050799

Academic Year
2022-2023

“Do or do not. There is no try”
— Yoda

Abstract

Fall-related injuries, especially among the elderly, can have severe consequences, including fractures, head injuries, and even death. Fall detection systems can help save lives by quickly alerting caregivers or medical professionals when a fall occurs, allowing for prompt medical intervention. This project explores a practical implementation of a domotic fall detection system that uses off-the-shelf low cost IoT devices interconnected through the Home Assistant platform. The objective of this study is to experiment and test that the design concept and the proposal is feasible for enhancing convenience, and security within the residential environment. The project begins by selecting a set of motion and presence sensors commercially available and compatible with the selected platform. These devices are integrated into a unified ecosystem using the Raspberry Pi 4 and Home Assistant software, which acts as a central hub for device management and data analysis. The results of the implementation demonstrate the system's effectiveness in providing assistance when a fall is detected by collection and analysis of the sensor data. Furthermore, the integration of the two motion sensors and a presence sensor help discriminate between regular walking/sitting/raising and a dangerous fall followed by an alarm system by triggering an alarm panel in order to turn on a siren and notify the caregiver by sending a WhatsApp message for immediate assistance. This implementation highlights the potential of utilizing readily available IoT technologies and the Home Assistant platform to demonstrate a fall detection system. This proof of concept aims at showing that such a fall detection system can be implemented.

Contents

Listing of figures

Listing of acronyms

IoT .... Internet of Things

SBC .... Single board computer

OS .... Operating System

GUI .... Graphical user interface

CLI .... Command line interface

GPIO .... General purpose Input/Output

DHCP .... Dynamic Host Configuration Protocol

IP .... Internet Protocol

SSH .... Secure Socket Shell

API .... Application programming interface

MQTT .... Message Queuing Telemetry Transport

HA .... Home Assistant

HAOS .... Home Assistant operating system

PIR .... Passive infrared sensor

SSL .... Secure Sockets Layer

HTTP .... Hypertext Transfer Protocol

UI .... User Interface

1
Introduction

Falls are a fearsome reality for older people especially those who live alone. The Centers for Disease Control and Prevention (CDC) reports that 3 million adults aged 65 and older are treated in emergency departments for fall injuries every year [1]. Equally alarming, one out of five falls causes a serious injury, which may permanently affect the person's mobility and independence.

Still according to the CDC report released in 2015, the fall-related injuries are among the 20 most expensive medical conditions. In 2020, a study published in the American Journal of Public Health estimates that the direct medical costs of fall injuries among older adults in the United States could reach up to $67.7 billion annually. The average cost [2] for a person admitted to a hospital due to falls was between $396.51 to $1429.35. It was determined that costs varied according to the type of the injury.

Although research shows that it is possible to prevent or lessen many injuries caused by falls, the reality is that falls do and will continue to negatively affect older adults. Fortunately, fall detection systems can help prevent the worst outcomes from occurring. Harnessing this technology can provide caregivers and families an added layer of support that can help offer additional peace of mind when it comes to their loved one's safety.

Today, fall detection systems help reduce the risk of long-term injury by detecting a person's abrupt change in position. The system calls for help even if the person is unable to do so themselves, signaling prompt emergency response to address the situation.

Traditional fall detection systems often rely on the individual's ability to activate an alert, such as pulling a cord or pressing a button. Other systems may use physical apparatus such as floor mats or infrared beams, which can deteriorate quickly, cause many false alarms, or even be considered invasive (they can be relatively expensive). While these devices do provide a degree of safety, they lack the sophistication needed to preemptively identify fall risks, resulting in delayed responses and an increased risk of injury. Not to mention, they can be uncomfortable and cumbersome for the user, thereby reducing the effectiveness and overall adoption rate.

Unlike conventional systems, sensor-based fall detection systems [3] leverage advanced Internet of Things (IoT) and Artificial Intelligence (AI) technologies to provide unobtrusive, real-time monitoring. These systems offer several notable advantages:

- Non-Intrusive Monitoring: Sensor-based systems use contact-free passive sensors that provide privacy, respect, and dignity to the individual. Without the need for intrusive cameras or wearable devices, users can freely go about their day with minimal disruption.
- Continuous Monitoring: Sensor-based solutions offer 24/7 monitoring, capturing valuable data on in/out of bed status, sleep patterns, activity levels, and more. This wealth of information can be used to create personalised risk avoidance plans and enable early interventions.
- Improved Efficiency and Cost: Sensor-based systems are cost-effective and offer a significant return on investment. By reducing the number of false alarms, these systems enable caregivers to focus on meaningful tasks, increasing their productivity and effectiveness.

This thesis is a Proof of Concept of a fall detection and alarm system that uses low cost IoT devices (motion and presence sensors). In chapter 2, we discuss the problems faced by the elderly, some solutions to these problems and the commercially available platforms to realize these domotic solutions. Chapter 3 explains the design principles of the fall detection system. Chapter 4 is dedicated to description of the various IoT devices used in this project. Chapters 5 and 6 include the testing and methodology of the Proof of Concept. Lastly, Chapters 8 and 9 provide references to the technical aspects of this thesis project.

2
Literature Survey

Home automation for the elderly and disabled focuses on making it possible for older adults and people with disabilities to remain at home, safe and comfortable. Home automation is becoming a viable option for older adults and people with disabilities who would prefer to stay in the comfort of their homes rather than move to a healthcare facility. This field uses much of the same technology and equipment as home automation for security, entertainment, and energy conservation [4] [5] but tailors it towards old people and people with disabilities.

2.0.1 Domotic solutions to problems faced by the elderly

Many living spaces are not adequate to support the needs of people with limited mobility and strength. Some of the most common problems encountered by elderly, for example, include difficulties in moving around the house because of the presence of steps, stairs, or furniture that is not designed for people in wheelchair or moving with the help of a cane or walker. People with impairment disabilities may find it hard to move around and perform tasks that require physical effort. The old traditional homes, unlike the modern ones, may not be designed with switches with remote control options making it hard for them to access with ease. There is no safety and could result in dangerous falls and accidents.

2.0.2 Technological solutions

Such problems can be partially solved or, at least, alleviated, by adopting proper technological solutions. Some examples of such systems are the following [6]

Voice-controlled home: This can include controlling lights, adjusting room temperature, opening or closing curtains, playing music, and even interacting with other smart home devices using custom voice commands. Integrate smart plugs or Wi-Fi relays to control appliances and devices through voice commands. This enables individuals with limited mobility to independently operate various devices such as fans, TVs, or kitchen appliances without the need for physical interaction.

Fall detection and emergency response: Utilize motion sensors or wearable devices with built-in accelerometers to detect falls. When a fall is detected, the system can automatically send alerts to caregivers or emergency services, turn on lights in the vicinity of the fallen person, and activate a two-way communication system to provide immediate assistance.

Smart medication management: Implement a system to remind users when to take their medications. This can involve using LED strips or lights to indicate the appropriate time for each medication and sending notifications to their smartphones or smartwatches. Additionally, integrate pill dispensers that can be remotely controlled to provide the correct medication dosage at the designated time.

Virtual caregiver presence: Use smart displays or tablets connected to the Home Assistant platform to create a virtual caregiver presence. Caregivers can remotely check in on the individual, communicate via video calls, provide reminders or prompts, and monitor their well-being through integrated sensors [7] and cameras.

Adaptive lighting: Configure lighting systems to automatically adjust based on the individual's needs. For instance, use motion sensors to turn on lights as the person enters a room and adjust the brightness level based on their preference or time of day. Additionally, implement color-changing LED strips to create visual cues or signals for specific actions or events.

Automated door and accessibility control: Install automatic door openers or smart locks that can be controlled remotely or via voice commands. This enables easy access for individuals with limited mobility, eliminating the need for physical keys or manual door handling. Additionally, integrate accessibility devices such as wheelchair lifts or ramps that can be controlled and monitored through the Home Assistant platform.

Personalized room automation: Set up personalized room automation profiles tailored to the needs of the individual. For example, configure the system to adjust the room temperature, lighting, and audiovisual settings based on their preferences or specific requirements, enhancing their comfort and overall well-being.

Environmental monitoring and air quality: Utilize sensors to monitor environmental factors such as temperature, humidity, and air quality. The system can provide real-time feedback and alerts to ensure a safe and comfortable environment for individuals with specific sensitivities or conditions.

Personal emergency response system (PERS): Implement a PERS using wearable devices with panic buttons or voice-activated triggers. In case of an emergency or distress, pressing the button or issuing a voice command can trigger alerts, notify designated contacts, and provide location information for immediate assistance.

2.0.3 Popular platforms to realize domotic solutions

There are several popular platforms commercially available for realizing domotic (home automation) solutions

Amazon Alexa is a voice-controlled assistant integrated into various devices, including smart speakers, TVs, and other smart home devices. Key Features are voice control, extensive device compatibility, third-party skill integration, routines, multi-room audio, and smart home hub capabilities. Widely adopted, extensive ecosystem, user-friendly voice control, supports a wide range of devices, and offers a variety of third-party skills and integrations. They have limited customization options, heavy reliance on voice commands, and potential privacy concerns.

Google Assistant is a voice-controlled assistant available on devices such as smartphones, smart speakers, and smart displays. Key Features are voice control, device compatibility, routines, multi-room audio, and integration with Google services and products. Broad ecosystem, extensive integration with Google services, strong AI capabilities, and supports a wide range of devices. Limited customization options, potential privacy concerns, and may require multiple apps for full functionality.

Apple HomeKit is a framework that allows users to control compatible smart home devices using Apple devices. Key Features are seamless integration with Apple devices, secure and privacy-focused, Home app for centralized control, and automation through scenes and triggers. Strong focus on privacy and security, seamless integration with Apple devices, intuitive user interface, and wide device compatibility. Limited device selection compared to other platforms, may require additional hardware (such as a HomePod) for full functionality.

Samsung SmartThings is a platform that connects and controls various smart home devices, compatible with both Samsung and third-party products. Key Features are broad device compatibility, extensive automation options, app-based control, and integration with Samsung devices and services. Supports a wide range of devices, robust automation capabilities, integration with Samsung ecosystem, and customizable through SmartApps. Initial setup can be complex, occasional connectivity issues, and limited third party integration compared to some other platforms.

Hubitat Elevation is a locally controlled home automation hub that allows users to connect and control various smart devices. Key Features are local processing for enhanced privacy and reliability, supports Z-Wave and Zigbee devices, customizable through rules and apps, and extensive automation capabilities. Local processing enhances privacy and reliability, robust automation features, customizable rules and apps, and supports a wide range of devices. Steeper learning curve for beginners, limited cloud integration, and smaller user community compared to larger platforms.

Home Assistant is an open-source home automation platform that allows users to control and automate various smart home devices. While it is not a commercially available platform in the traditional sense, it is widely used and supported by a passionate community of developers and enthusiasts. Home Assistant is a flexible and extensible platform that runs on various hardware, including Raspberry Pi, servers, and cloud-based solutions. It provides a centralized interface for managing and controlling a wide range of smart home devices from different manufacturers. Home Assistant supports a vast number of integrations and protocols, allowing users to connect and control devices through a unified platform.

2.0.4 Fall detection systems

There are various existing solutions [1] for fall detection systems and new technologies may emerge in the future. Fall detection systems are designed to detect and respond to falls, especially in vulnerable populations such as the elderly. Here are some common existing solutions:

Wearable Devices: Many modern smartwatches and fitness trackers are equipped with sensors (accelerometers and gyroscopes) that can detect sudden movements indicative of a fall. Some devices provide automatic fall detection and can send alerts to predefined contacts or emergency services.

Mobile Apps: There are mobile applications specifically designed for fall detection. These apps utilize the sensors present in smartphones to detect falls and can send alerts or notifications to emergency contacts.

Motion-Activated Systems: These systems use motion sensors placed in a person's living environment (such as a home or assisted living facility) to detect unusual movements or a lack of movement. If a fall is detected, an alert can be sent to caregivers or emergency services.

Camera-Based Systems: Some fall detection systems use computer vision technology through cameras. These cameras monitor the movements of individuals and can detect falls based on changes in posture and movement patterns.

In-Home Monitoring Sensor Networks: Integrated sensor networks within the home can monitor activities and detect falls. These systems may include pressure sensors on the floor, bed, or furniture, which can identify sudden changes or absence of pressure.

Traditional Medical Alert Systems: Many medical alert systems include fall detection features. These systems typically involve wearable devices with buttons that users can press to call for help manually. Some also automatically detect falls.

Advanced Analytics: Some fall detection systems leverage machine learning algorithms to analyze data from various sensors. These algorithms can learn and adapt to individual movement patterns, improving the accuracy of fall detection.

IoT (Internet of Things) Connected Devices: IoT devices, such as sensors embedded in clothing or accessories, can be part of a fall detection system. These devices communicate with a central system to relay information about the wearer's movements.

It is essential to note that the effectiveness of these solutions can vary, and no system is foolproof. The choice of a fall detection system often depends on the specific needs of the individual, the environment, and any existing health conditions.

3

Design Principles of the Fall Detection System

The goal of this project is to use **low cost IoT devices [2] and Motion Sensors** to realize a fall detection system with the following characteristics:

- Very low implementation costs: The system prioritizes the use of affordable, standard off-the-shelf devices. This approach is essential to reduce both purchase and installation costs, thereby ensuring the solution is economically viable and widely accessible.
- No need for wearable devices: Some users find certain wearable devices bulky, heavy, or uncomfortable, leading to issues such as skin irritation or discomfort. Therefore, our solution should function autonomously, eliminating the need for users to wear any device or perform any specific action to activate the system.
- Good accuracy and precision: The system should provide accurate or consistent data by capturing the real time events.
- Low false alarm rate: There may be very few exceptions that lead to false alarms but the occurrence of these unwanted events shall be minimized.
- Small delay in fall detection: The detection of fall events shall be quick, in the order of tens of seconds, to enable prompt reactions while limiting the number of false detections.

3.1 Implementation of a fall detection system using low cost IoT sensors

3.1.1 Proposed idea

The idea was to use standard motion sensors [8] in two ways: one sensor should only detect motion above a certain height (say, 0.7 m), the other should be configured to detect motions below another height (say, 0.4 m).

The following events were considered in order to detect a fall:

- Case 1 : If a person is walking in the room/standing, both sensors should detect motion.
- Case 2: If a person sits/simply bends down to pick an object the upper sensor must not detect any motion for a few seconds but the state must still remain the same (delay in the state change) while the sensor below should detect motion.
- Case 3: If the person falls and remains on the ground, the upper sensor shall stop detecting motion and change its state, while the lower one keeps detecting motion if the person does not lose consciousness.

By analyzing these events, we may be able to detect a fall. There are however different events that may result in false detection:

- A pet entering the room triggers the lower sensor, but not the upper one.
- A person does fall, but he/she remains still on the floor, so that even the lower sensor does not detect motion.

The accuracy of the detection can be elevated by using a presence sensor that has a wider range, which should be able to detect micro-movements. Once the fall is detected, an alarm is triggered in order to notify the caretakers or the neighbors for immediate assistance. Such a sensor, therefore, shall be able to detect the presence of a person even when still (e.g., become unconscious), by exploiting the small movements due to breathing. By adding this sensor, then, it will be possible to recognize a person lying on the floor, even if unable to move. Clearly, the sensor will also react to the presence of pets, which may trigger false alarms. However, the sequence in which the different motion sensors are activated shall reveal the nature of the event. In case of a pet entering the room, in fact, only the lower motion sensor and the micro-movement sensor would be triggered, thus making it possible to identify this event. Of course, there are more complex situations that might still give rise to false alarm, e.g., when a pet enters a room together with a person, and then the person leaves the room and the pet remains there. These specific cases can be treated by adding some extra sensors (e.g., on the room door) or deploying more sophisticated signal processing algorithms, but these developments are left to future research.

3.1.2 Development of the Fall Detection System

For this project, the **Home Assistant** was a good choice for automation and data management since it is open source and compatible with a wide range of IoT devices. The latest Home Assistant operating system works best on the Raspberry Pi 4 microcontroller. The Aqara FP2 Presence sensor, Tapo hub H100 and Tapo standard Motion sensors T100 are used since they work with technologies supported by Home Assistant. We then Pair and Integrate all the sensors and IoT devices with Home assistant followed by testing the entities of the sensors to detect motion/presence, delays, area coverage and sensitivity. Next we set up the Test bed by wall mounting the Tapo sensors, one at an average shoulder height and the other below knee level, and the presence sensor can be placed anywhere, preferably mounted at the top of a wall in the room as per the proposed idea. The Tapo sensors shall be mounted in such a way that it covers a range of one of the zones of the Aqara sensor. We may now detect fall in only one zone with this set up due to the Tapo range constraints. Influxdb can be used on Home Assistant to create a database and store the sensor data and Grafana to represent and analyze the data by generating graphs. To trigger alarms when a fall is detected, we automate the alarm system by adding the triggers, conditions and actions. Once the alarm panel is triggered the siren must turn on and ring for 60 seconds and a WhatsApp message is sent to a pre-determined phone number from the Home Assistant.

The idea was further developed and tested at home conditions only. The current implementation is a proof of concept, aimed at showing that such a fall detection system can be implemented.

4

Description

4.1 Home Assistant

Home Assistant is free and open-source software [9] for home automation designed to be a central control system for smart home devices with a focus on local control and privacy, the developers being Paulus Schoutsen, Home Assistant Core Team and Community development team. It is written in python 3.11, running with Software appliance / Virtual appliance (Linux) operating system using the ARM, ARM64, IA-32 (x86), and x64 (x86-64) platform. After the Home Assistant software application is installed as a computer appliance, it will act as a central control system for home automation, commonly referred to as a smart home hub, that has the purpose of controlling IoT connectivity technology devices, software, applications and services which are supported by modular integration components, including native integration components for wireless communication protocols such as Bluetooth, Zigbee, and Z-Wave (used to create local personal area networks with small low-power digital radios). Home Assistant also supports controlling open and proprietary ecosystems if they provide public access via an Open API or MQTT for third-party integrations over the local area network or the Internet.

4.1.1 History

In July 2017, a managed operating system called Hass.io was initially introduced to make it easier to use Home Assistant on single-board computers like the Raspberry Pi series. Its bundled supervisor management system allowed users to manage, backup, update the local installation and introduced the option to extend the functionality of the software with add-ons. In January 2020, branding was adjusted to make it easier to refer to different parts of the project. The main piece of software was renamed to Home Assistant Core, while the full suite of software with the Hass.io embedded operating system with a bundled supervisor management system was renamed as Home Assistant (though it is also commonly referred to as “HAOS” as in short for “Home Assistant OS”). In January 2021, Home Assistant made a public service announcement, disclosing vulnerabilities with its 3rd party custom integrations. Later in January 2021, it made a second security disclosure about a security vulnerability.

4.1.2 Features of Home Assistant

The primary front-end dashboard system is called Lovelace (named after Ada Lovelace), which offers different cards to display information and control devices. Cards can display information provided by a connected device or control a resource (lights, thermostats, and other devices). The interface design language is based on Material Design and can be customized using global themes. The GUI is customizable using the integrated editor or by modifying the underlying YAML code. Cards can be extended with custom resources, which are often created by community members.

Home Assistant acts as a central smart home controller hub by combining different devices and services in a single place and integrating them as entities. The provided rule-based system for automation allows creating custom routines based on a trigger event, conditions and actions, including scripts. These enable building automation, alarm management of security alarms and video surveillance for home security systems as well as monitoring of energy measuring devices. Since December 2020, it is possible to use automation blueprints - pre-made automation from the community that can be easily added to an existing system.

In terms of security, Home Assistant, as an on-premises software product, with its focus on local control for the purpose of privacy in combination with its state as an open-source application, has been described as beneficial to the security of the platform specifically when compared to closed-source home automation software based on proprietary hardware and cloud-services. There is no remote access enabled by default and data is stored solely on the device itself. User accounts can be secured with two-factor authentication to prevent access even if the user password is known by the attacker. Add-ons get a security rating based on their access to system resources.

4.2 IoT devices and sensors

4.2.1 Aqara presence sensor FP2

The Aqara [10] offers multiple features that include zone positioning, multi-person detection, ultra high precision, multi-Ecosystem Support, flexible Placement, built-in light sensor and local automations.

Zone positioning offers significant advantages over traditional Passive Infrared (PIR) sensors by monitoring rooms of up to 40 square meters, and dividing them into up to 30 zones. It is suitable for a practically endless variety of use cases based on the actual presence in each of such zones. Additionally, using just two conditions of Presence and Absence will make the automation configuration simpler than ever before. Moreover, the FP2 supports detecting up to 5 people simultaneously, making it particularly useful for households with multiple occupants. It can be used for home automation and security purposes, but also for energy management, such as heating and cooling systems based on the actual room presence. By implementing millimeter-wave radar technology, the FP2 provides a precise presence detection without producing false negatives, even for the slightest movements. The FP2 is exposed as multiple sensors in HomeKit (Also on Home Assistant), Amazon Alexa, Google Home and Alice. We are able to integrate the sensor easily with no Aqara hub required, which is a good solution for smart home enthusiasts to expand your smart home setup.

The FP2 is revolutionary because of its mmWave radar technology which is the next big thing in smart home sensing. We will not need several sensors to detect different areas within a room, and careful configuration and long-term management of each sensor. With FP2, we will have multiple infrared (PIR) sensors and even one extra light sensor.

Specifications

- Operating Temperature : -10°C - 40°C (14°F - 104°F)
- Ports : USB-C
- Wireless Protocols : Wi-Fi IEEE 802.11 b/g/n 2.4 GHz, Bluetooth 4.2
- Operating Humidity : 0 - 95% RH, no condensation

- Input Power : 5V==1A
- Dimensions : 64 × 64 × 29.5 mm (2.52 × 2.52 × 1.16 in.)

Figure 4.1: Aqara Presence Sensor FP2

4.2.2 Tapo Hub H100

The Tapo hub [11] is a Smart Hub that comes with a door bell feature. It connects up to 64 smart devices for a complete smart-living experience. It has a wide coverage by keeping secondary devices always connected through the wireless network. The Low-Power wireless protocol allows you to connect secondary devices more efficiently, decreasing the power consumption of battery-operated devices. The hub has an inbuilt smart alarm by pairing the Smart Hub with a Smart Sensor to trigger an audible alarm when any intruders pass by. It also has a smart doorbell that works when Pairing the Smart Hub with a Smart Button and choosing from 19 ringtones to create your own smart doorbell. The Tapo Smart Hub transmits signals over a less crowded, low-frequency broadband, allowing it to reliably reach devices in the coverage area without interference. When paired with the Tapo motion sensors, door/window sensors, and more, the Tapo Hub can sound a loud siren (up to 90 dB) to warn of danger or deter intruders. Also, when paired with Tapo environmental sensors, the Tapo Hub helps to keep an eye on your home's comfort levels. You will receive an instant hub alarm when water leaks are detected or when temperature and humidity levels fall outside your customized ranges.

Specifications

- Protocol : 868 / 922 MHz
- Wireless Type : 2.4GHz Wi-Fi
- Operating Temperature : 0-40°C
- Dimensions : 72 x 62.5 x 51 mm (2.8 x 2.5 x 2 inch)
- Buttons : SYNC/Mute Button
- Power Requirements : AC, 100-240V, 50/60Hz

4.2.3 Tapo Motion Sensor T100

The Tapo Motion sensor T100 is a smart motion sensor that has a reasonable detection range of up to 7m at an angle of 120°. It has a lighting control and energy saving feature by turning Tapo bulbs and LED strips on or off as you pass by. The smart sensor alerts and sends notifications via app by setting Away Mode to receive detection notifications when you are away. It has a long-lasting battery up to over a year of use. Easy to Install using the included adhesive backing or built-in magnet. The sensor is powered by a long-lasting battery, applying it to any type of surface thanks to the integrated magnetic support or the adhesive included in the package. Tapo T100 connects to the smart hub using a low-power 868 MHz / 922 MHz protocol that optimizes the use of battery energy, guaranteeing a duration of up to over a year of use. By combining the action of your Tapo smart devices via Smart actions simultaneously activate the lights of a room when you enter, or turn them off when you are leaving. A Hub is required to function correctly and communicate with the other devices in the range, Tapo T100 must be paired with a Tapo smart Hub such as the H100 model.

Specifications

- Wireless : 868MHz / 922MHz
- Detection Angle : 120°
- Detection Distance : Max 7 m / 23 ft.

- Operating Temperature : 0-40°C
- Dimensions : 42.3 x 42.3 x 34mm
- Power Requirements : CR2450 Battery
- Hub Required : Yes
- LED indicator : Yes

Figure 4.2: Tapo Motion Sensor T100 and Tapo Hub H100

4.3 Raspberry Pi

Raspberry Pi is a popular single-board computer (SBC) that has gained popularity in recent years due to its versatility and affordability. It has a wide range of applications, from learning programming to building smart homes, IoT devices, and media centers. One of the crucial decisions that Raspberry Pi users have to make, is choosing the right operating system.

Some of the main features of Raspberry pi OS

- Lightweight and optimized: Raspberry Pi OS is optimized to run on Raspberry Pi's ARM-based processor, which makes it lightweight and fast. It also includes a set of pre-installed software packages that are optimized for the hardware, such as the Chromium web browser and the Thonny Python IDE.

- User-friendly interface: Raspberry Pi OS has a user-friendly graphical user interface (GUI) based on the LXDE desktop environment. It includes a taskbar, start menu, and file manager, making it easy for beginners to use.
- Command-line interface: Raspberry Pi OS also includes a powerful command-line interface (CLI) that provides access to the full range of Linux tools and utilities. This is particularly useful for advanced users and developers.
- Customizable: Raspberry Pi OS is highly customizable, with a range of configuration options and tools available. This allows users to tailor the operating system to their specific needs and preferences. Overall, Raspberry Pi OS is a versatile and powerful operating system that is well-suited for a wide range of applications, from home automation to robotics and education.

In this Proof-of-Concept each of the above described IoT devices and sensors play a vital role in the development of the fall detection system.

Tapo Motion Sensors are one of the key elements of this fall detection system. Their role is to detect motion at the ground and few meters above the ground levels. The changes in their state help determine if a person is walking/standing/sitting or remains on the ground indicating a dangerous fall.

Aqara FP2 Presence Sensor is used to increase the accuracy of the fall detection system. For instance, if an object is placed as an obstacle in the detection zone area, the Tapo Motion Sensors detect their presence. The sensors are in "detected" state. This may lead to a false alarm. To avoid this, the Aqara FP2 Presence sensor is used because of its key feature to be able to differentiate between persons and objects. Aqara FP2 Presence Sensor also helps divide the room into multiple zones because of its multi-zoning feature. This division makes it easy while working with automating the alarm system. Therefore, this sensor is used to detect presence in a particular area of the room and avoid false alarms.

The Home Assistant acts as a central hub where all the sensors are integrated into a unified platform for sensor data management and automation. Home Assistant server runs best on either Raspberry Pi 4 or 3 as per the official Home Assistant documentation.

5
Testing

In this section we will see how the test bed is setup and the Proof-of-Concept of the fall detection. The sensors are wall mounted as mentioned in the proposed idea to begin with the testing.

Note the following references:
Detection Zone 1: Aqara FP2 Sensor's focus area of the fall detection.
Sleeping Area: One of Aqara FP2 Sensor's zone in the room. This zone is used for testing false alarms.
Motion Sensor High / Motion Sensor 1: Tapo Motion Sensor mounted at the shoulder level, few meters above the ground.
Motion Sensor Low / Motion Sensor 2: Tapo Motion Sensor mounted at the ground level.
Tapo Motion Sensors: Referring to Motion Sensor High and Motion Sensor Low.

5.1 Test bed

The Aqara Presence Sensor FP2 is a Wi-Fi IoT device that is powered by using a USB cable. Therefore the sensor is powered up using a power bank with 2.1A USB port. The Aqara is connected to a 2.4Ghz Wi-Fi and is wall mounted using some tape at around 2m above ground level placed in the center of a wall. The sensor divides the room into 4 zones including **Detection Zone 1, 2, 3 and Sleeping Area**. It also consists of two other entities namely, **Occupancy and Luminescence sensor**. Refer to Figure 5.4 to see these entities on the Home Assistant.

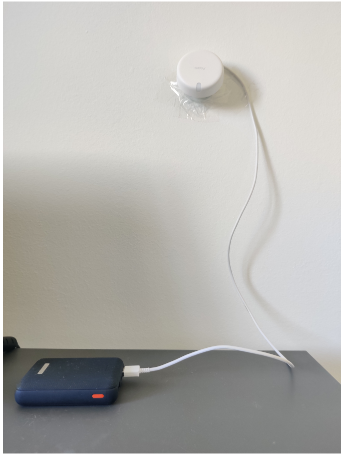

Figure 5.1: Aqara Presence Sensor position

The Tapo Hub H100 and the Raspberry Pi are plugged into a socket to get powered up. Motion Sensor High and Motion Sensor Low are paired to this hub. Refer to Figure 5.2 for a picture of the devices.

Figure 5.2: Tapo Hub and Raspberry Pi position

The Tapo Motion Sensors come with a CR2450 Battery. Motion Sensor High is placed at a height of about 1m and the Motion Sensor Low is placed at around 15cm from the ground level. Both these sensors are placed vertically on the wall. The pink zone is the coverage area of the Tapo Motion Sensors and the Detection Zone 1, refer to Figure 5.3

Figure 5.3: Tapo Motion Sensors position

5.2 Test Cases: Presence detected in one zone only

Due to the range constraints of the Tapo Motion Sensors, in this project we focus on detecting a fall in **Detection Zone 1**. The test bed is set up as described in Section 5.1 to proceed with the testing of different scenarios to collect sensor data and analyse them in order to implement a fall detection system.

Case 1 : Nobody in the room.

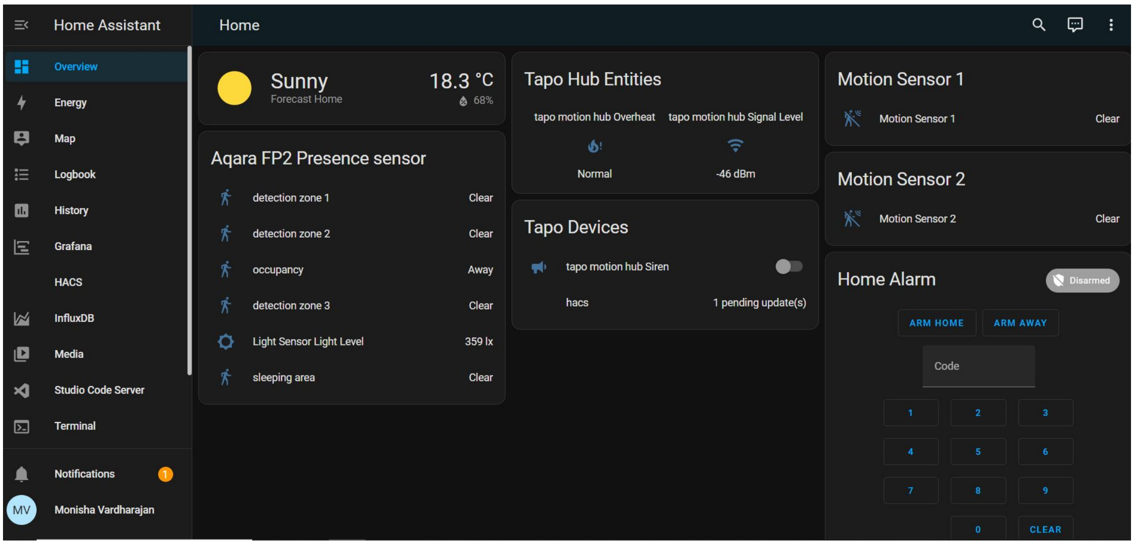

Figure 5.4: Home Assistant dashboard entities when there is nobody in the room

As we observe from Figure 5.4, the Aqara sensor entities including Detection Zones 1, 2, 3 and Sleeping Area are in "clear" state. The Occupancy sensor indicates the state "away" which means there is no presence detected in the room. The Tapo Motion Sensors are in "clear" state as well.

Case 2 : A person or a pet enters the room but is away from the Detection Zone 1.

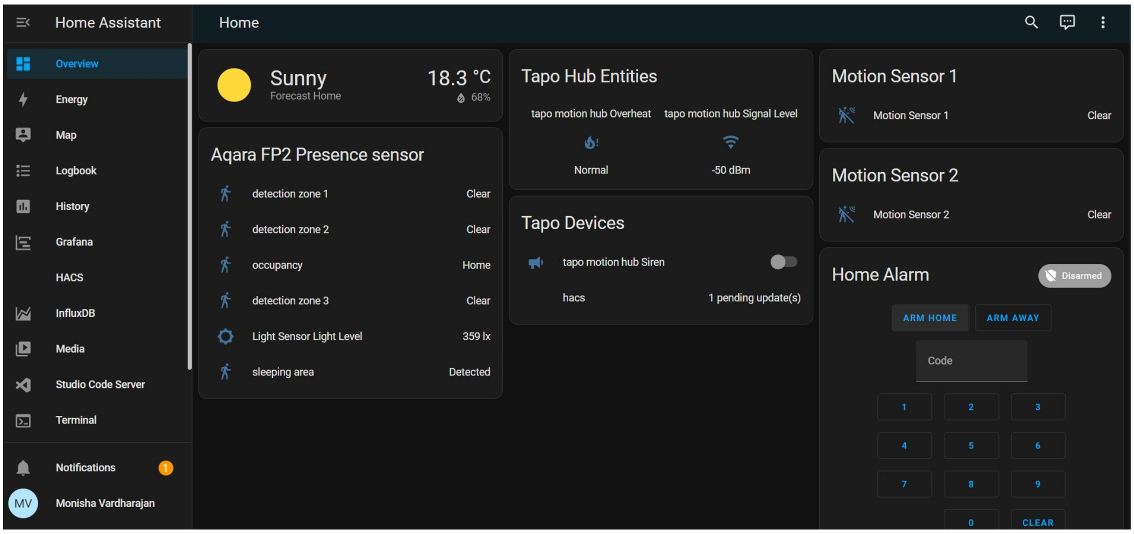

Figure 5.5: Home Assistant dashboard entities when a person is away from detection zone 1

Figure 5.5 Depicts the home assistant dashboard with the Aqara presence sensor and the Tapo Motion sensors entities clearly showing the states of each entity when a person walks into the room and is present on the Sleeping Area of the room. The person might be walking, standing still or sitting on the sleeping area. This case demonstrates a scenario in which when a person or a pet enters the room and is in motion, or is standing still, or is sitting in any of the zones apart from Detection Zone 1. The Tapo motion sensors will not detect the presence because of its range constraints, it can only detect presence/motion in the Detection Zone 1. Therefore in this case the Tapo Motion sensors are in "clear" state while presence is detected in one of the Aqara zones (Sleeping Area) other than Detection Zone 1. We see that there is presence/motion detected in the Sleeping Area and the Occupancy Sensor has changed from "away" to "home", which means that presence has been detected in the room. The Tapo Sensors are in "clear" state. The other entities including Detection Zone 1, 2, 3 are also in "clear" state since there is no detection of presence or motion in those areas.

Case 3 : A person walks/stands/sits/bends to pick an object on Detection Zone 1.

(a) motion

(b) standing still

(c) sitting

Figure 5.6: Motion/standing still/sitting on detection zone 1

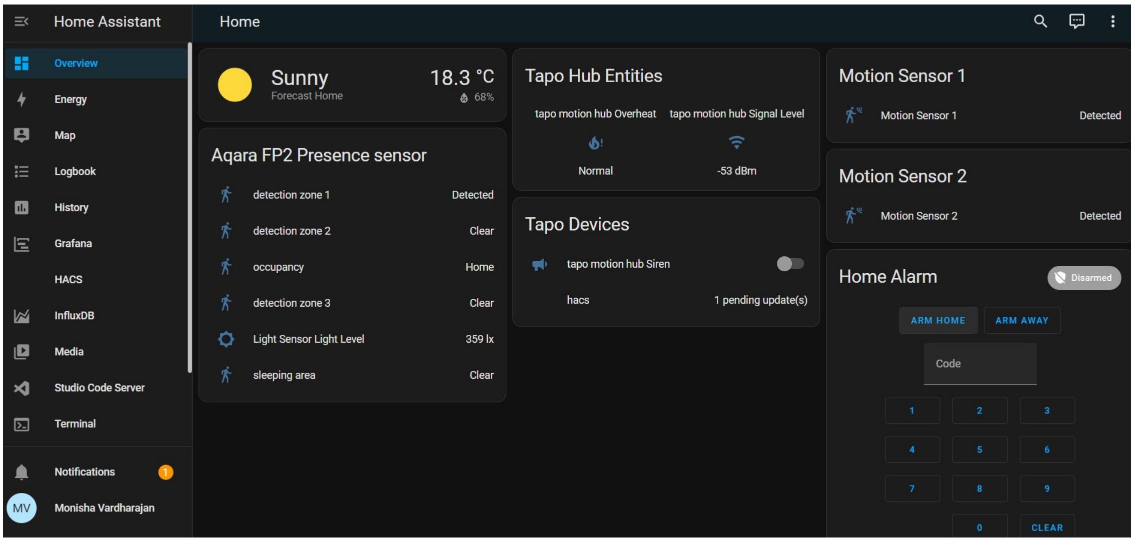

Figure 5.7: Home Assistant dashboard entities when a person walks/stands/sits/bends to pick on detection zone 1

From Figure 5.7 we observe the changes in the entities of the Tapo Motion Sensors. Tapo Motion Sensors are now in "detected" state as the person is walking, standing or sitting as seen in Figure 5.6 on Detection Zone 1. The Aqara Sensor entities have also changed where Detection Zone 1 is in "detected" state and the Sleeping Area is now in "clear" state.

In case a person bends to pick an object and rises back the motion sensor 1 does not change its state since the sensor by default works with a delay of at least 40 seconds to change its state from "detected" to "clear" and vice versa. So if the person bends to pick something and gets back within 40 seconds, the alarm will not get triggered. However, if a person remains crouched for more than 40 seconds, a false alarm is encountered.

Case 4 : A person is falling on Detection Zone 1.

Figure 5.8: Falling on detection zone 1

Figure 5.8 shows a person falling on the Detection Zone 1. The Home Assistant dashboard entities in this case remain the same as Figure 5.7 for the same reason as mentioned earlier that the Tapo Motion Sensors have a default delay when there is a change of state from "detected" to "clear" and vice versa.

Case 5 : A person remains on the ground after falling on Detection Zone 1.

Figure 5.9: Fall on detection zone 1 and remains on the ground for more than 40 seconds

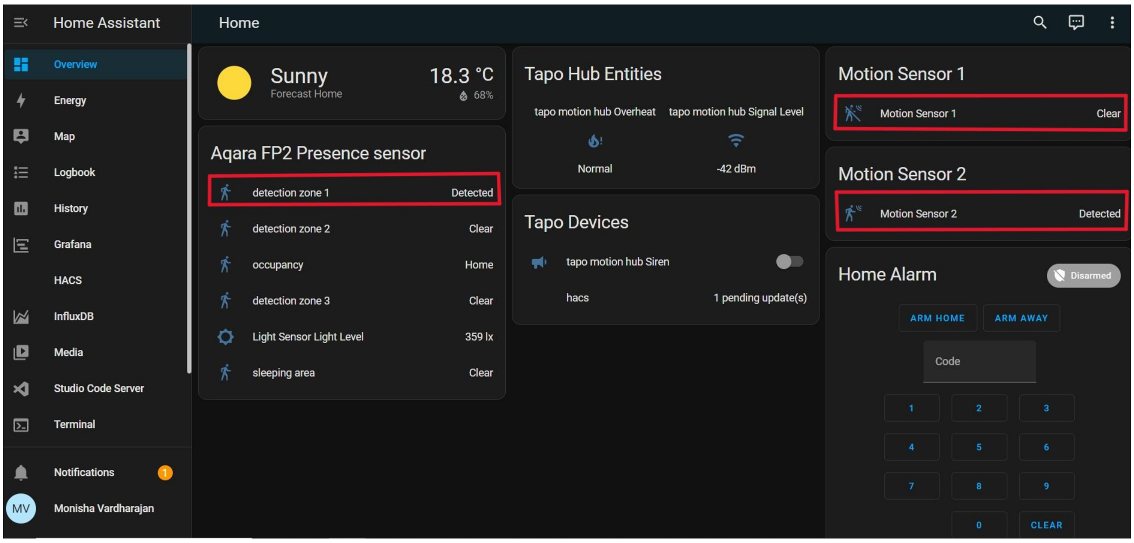

Figure 5.10: Home Assistant dashboard entities when a person falls on detection zone 1 and remains on the ground for more than 40 seconds

From Figure 5.10 we observe the sensor entities on the dashboard when a fall has occurred on Detection Zone 1 and the person remains on the ground as seen in Figure 5.9 for more than 40 seconds, the Aqara Sensor entity Detection Zone 1 is in "detected" state and the Motion Sensor Low is also in "detected" state while the Motion Sensor High changed its state from "detected" to "clear". At this instance a smart alarm is automated to be triggered in such a way that the neighbors and the caretakers get notified that a fall has occurred and requires immediate assistance.

Case 6 : A pet is in motion/sitting still on Detection Zone 1.
In this case the Home Assistant dashboard entities are the same as Figure 5.10. The Motion Sensor High is in "clear" state, the Motion Sensor Low is in "detected" state and the Detection Zone 1 is in "detected" state. This triggers a **false alarm**. We might need additional IoT devices, for instance, a camera to differentiate between a pet and a person and creating automations accordingly.

Case 7: If presence is detected in 2 or more zones as seen in Figure 5.11, the alarms will still be **triggered** if the state of Detection zone 1, Motion Sensor Low are in "detected" state and Motion Sensor High is in "clear" state irrespective of the states of the other aqara entities. This project focuses on detecting a fall in Detection Zone 1 only to simplify the test cases. However, automations can be created and modified based on the application scenarios.

5.3 Test Cases: Presence detected in two zones

Figure 5.11 and Figure 5.12 represent test cases performed when multiple people/pets were present in the room to check for false alarms. In all the cases mentioned below Detection Zone 1, Sleeping Area and Motion Sensor Low are in "detected" state while Motion Sensor High is in"clear" state.

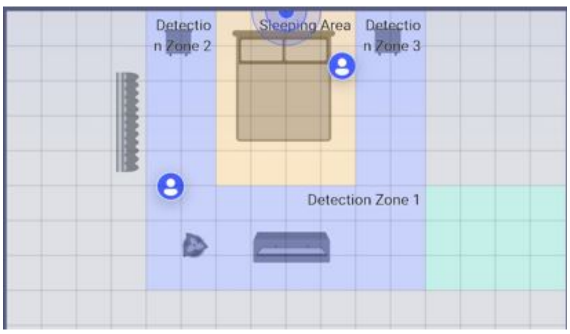

Figure 5.11: Aqara app dashboard representing presence on two zones in the room

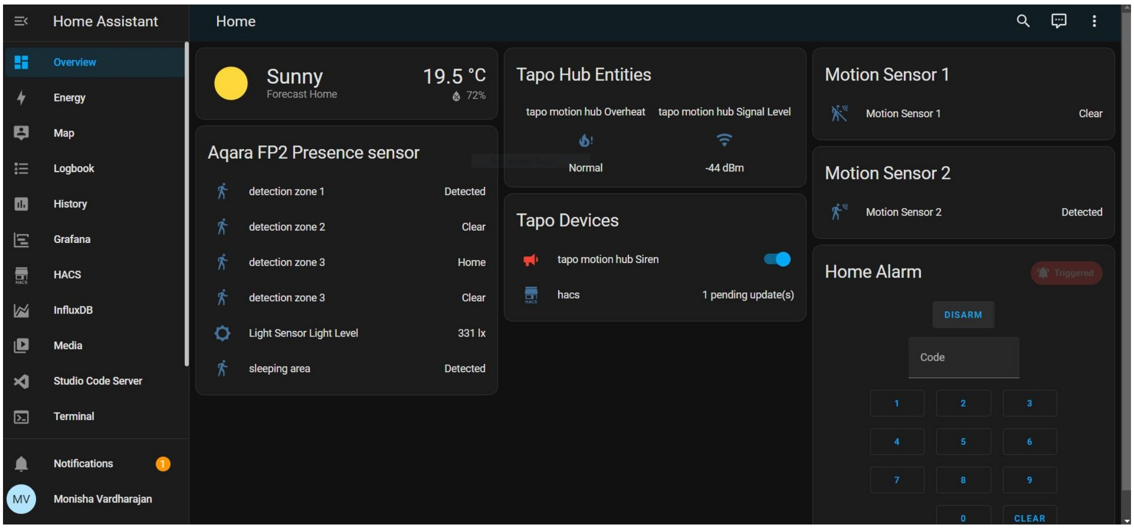

Figure 5.12: Home Assistant dashboard entities when two people/pets are present in the room and trigger alarms

Case 1: A person is on Detection Zone 1 and a pet is on the Sleeping Area. As we observe that the state of Motion Sensor High is "clear", the person has fallen and remained on the ground for more than 40 seconds.

Case 2: A pet is on Detection Zone 1 and a person is on the Sleeping Area. As we observe in this case that the state of Motion Sensor High is "clear" and Motion Sensor Low is "detected", which means the pet has triggered only the Motion Sensor Low leading to a false alarm.

Case 3: There are two persons in the room, one is on Detection Zone 1 and the other is on the Sleeping Area. Since the state of the Motion Sensor High reflects a "clear" state, it is evident that the person on Detection Zone 1 has fallen and remained on the ground for more than 40 seconds.

Case 4: There are two pets in the room, one on the Detection Zone 1 and the other on the Sleeping area. In this case the state of Motion Sensor High is "clear" and Motion Sensor Low is "detected" , which means the pet on Detection Zone 1 has triggered only the Motion Sensor Low leading to a false alarm.

Cases 1 and 3 trigger alarms that require immediate assistance while cases 2 and 4 trigger false alarms. To conclude, the above discussed cases can either be genuine or false alarms. To overcome this challenge, we may have to use additional IoT sensors/devices compatible with Home Assistant in order to differentiate between pets and people.

6
Methodology

6.1 Data Collection, Management and Analysis

Data collection, management, and analysis are crucial components in various fields and industries, driven by the increasing reliance on data-driven decision-making. Here are some key reasons highlighting the need for these processes:

- Informed Decision-Making: Organizations use data to formulate strategies, set goals, and make informed decisions. Data analysis provides insights into market trends, customer preferences, and other factors that influence strategic planning. Data analysis helps identify potential risks and allows organizations to develop strategies to mitigate them, leading to more informed and proactive decision-making.

- Performance Evaluation: Data is essential for evaluating the success of projects, campaigns, and overall business performance. Key performance indicators (KPIs) derived from data help assess progress towards goals and objectives. Regular data analysis allows organizations to identify areas for improvement, optimize processes, and enhance overall performance.

- Customer Understanding: Businesses collect and analyze customer data to understand their preferences, behaviors, and needs. This information enables the customization of products, services, and marketing efforts to better meet customer expectations. Data-driven insights help in identifying factors influencing customer satisfaction and loyalty, enabling businesses to implement strategies for customer retention.

- Operational Efficiency: Data analysis helps organizations optimize resource allocation, streamline processes, and reduce operational costs. This is particularly important in industries with tight margins and high competition. Efficient supply chain operations depend on accurate data for inventory management, demand forecasting, and logistics optimization.

- Regulatory Compliance: Many industries are subject to regulatory requirements regarding data storage and reporting. Proper data management ensures compliance with legal and regulatory standards. As the importance of data grows, so does the need for robust data security measures to protect sensitive information from unauthorized access and breaches.

- Research and Development: Data plays a crucial role in research and development, helping organizations identify trends, opportunities, and areas for innovation. It facilitates evidence-based decision-making in the development of new products and services.

In summary, data collection, management, and analysis are integral to informed decision-making, efficiency, innovation, and compliance across a wide range of industries and sectors. The ability to extract meaningful insights from data is a competitive advantage in today's data-driven world.

In this project we use two Home Assistant Add-ons for the sensor data collection, management and analysis, namely InfluxDB and Grafana for data storage and visualization respectively.

6.1.1 InfluxDB

Home Assistant Community Add-on: InfluxDB
InfluxDB [12] is an open source time series database optimized for high-write-volume. It is useful for recording metrics, sensor data, events, and performing analytics. It exposes an HTTP API for client interaction and is often used in combination with Grafana to visualize the data. This add-on comes with Chronograf and Kapacitor pre-installed as well which gives you a good InfluxDB admin interface for managing your users, databases, data retention settings, and lets you peek inside the database using the Data Explorer. Refer to Figure 6.1 to view an example GUI of InfluxDB.

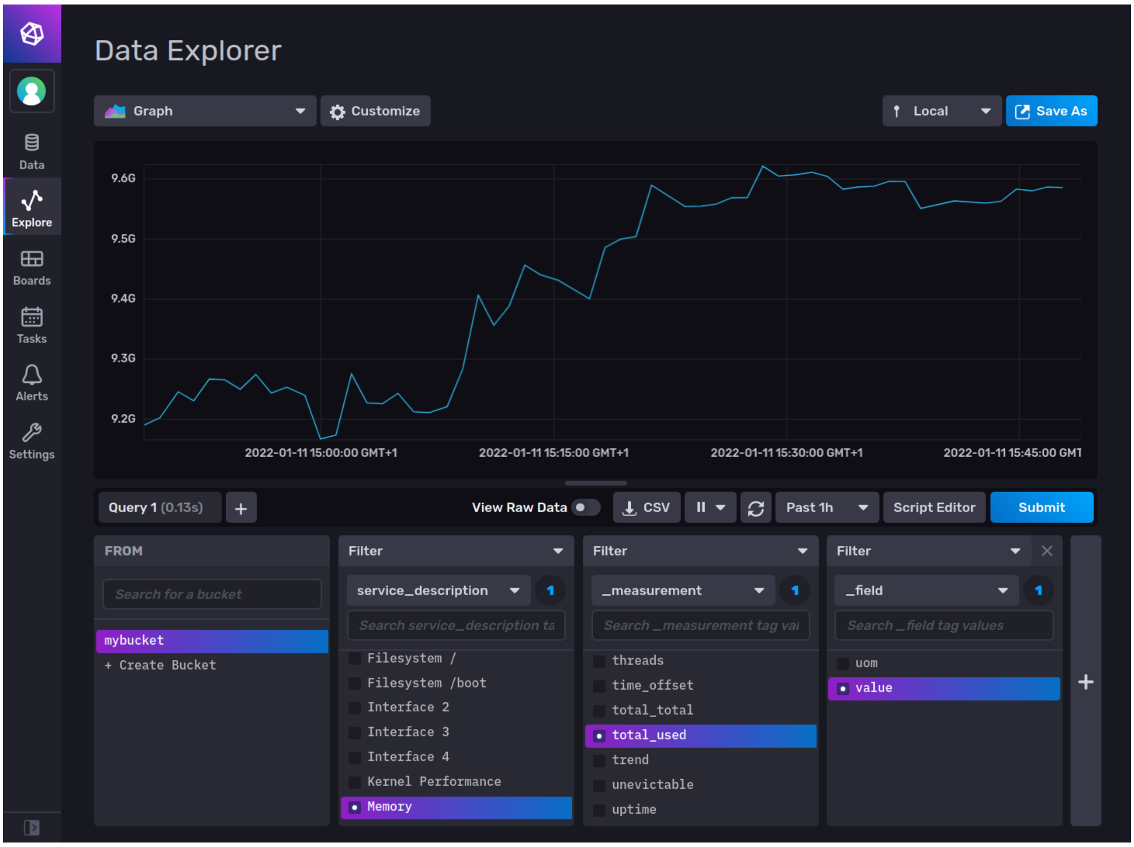

Figure 6.1: Example of an InfluxDB GUI

Here are some common features you might find in InfluxDB GUIs:

- Dashboard Views: A dashboard that provides a high-level overview of key metrics and data points. The ability to customize dashboards with different panels and visualizations.
- Data Exploration: Tools for creating various types of charts and graphs to visualize time-series data. The ability to zoom in and pan across time ranges to explore specific periods of data.
- Query Builder: A user-friendly interface for constructing InfluxQL queries without needing to write raw InfluxQL commands. A history feature that stores previous queries for easy reference.
- Data Management: Functions for importing and exporting time-series data. Options for managing tags and fields associated with time-series data.

- Integration with Other Tools: Some InfluxDB GUIs integrate with Grafana for more advanced data visualization and dashboard creation. The ability to execute InfluxDB commands directly from the GUI.
- Historical Data and Retention Policies: Features for managing retention policies for time-series data.

6.1.2 Grafana

Home Assistant Community Add-on: Grafana

Grafana [13] is an analytics platform for all your metrics. Grafana allows you to query, visualize, alert on and understand your metrics no matter where they are stored. You can create, explore, and share dashboards. Also, learn about your Home Automation system using compelling graphs, and other data visualizations. When we combine this add-on with the InfluxDB add-on we get very powerful insights to our home. Refer to Figure 6.2 to view an example GUI of Grafana.

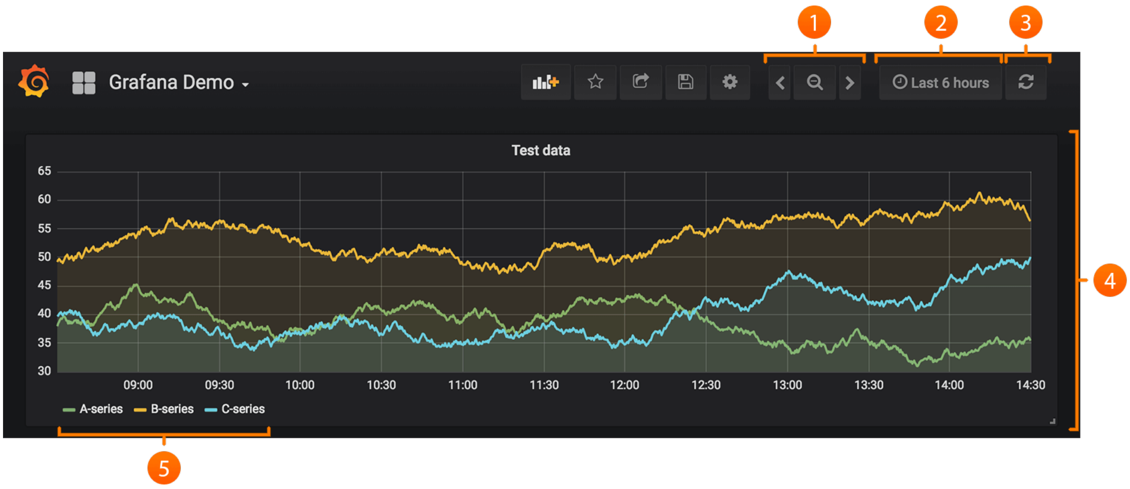

Figure 6.2: Example of a Grafana GUI

where,
1 - Zoom out time range.
2 - Time picker drop-down: You can access relative time range options, auto-refresh options, and set custom absolute time ranges.

3 - Manual refresh button: Will let all panels refresh (fetch new data).

4 - Dashboard panel: Tap the panel title to edit panels.

5 - Graph legend: You can change series colors, y-axis, and series visibility directly from the legend.

6.2 Alarm System

A notification alarm system [14] is designed to alert individuals or entities about specific events, incidents, or conditions. The features of such a system may vary depending on its intended use, but here are some common features associated with notification alarm systems:

- Customizable Alerts: Users can set specific criteria or thresholds for triggering alerts. This customization allows the system to notify users based on their unique requirements.
- Multi-Channel Notifications: Notification systems often support multiple communication channels, such as text messages, emails, phone calls, push notifications, or even integration with messaging platforms like Slack or Microsoft Teams.
- Real-time Alerts: The system provides real-time alerts, ensuring that users are promptly informed of critical events as they occur.
- Escalation Policies: For important events, escalation policies can be implemented, meaning that if the initial notification is not acknowledged within a specified time frame, the system can escalate the alert to additional recipients or higher levels of severity.
- Integration with Other Systems: Notification alarm systems often integrate with other systems, such as security cameras, environmental sensors, or monitoring platforms, to provide a comprehensive overview of the situation.
- Geolocation: Some systems can send notifications based on the geographic location of users or assets, ensuring that alerts are relevant to specific areas.
- Historical Logging: The system keeps a log of past alerts and events, allowing users to review the history of notifications for analysis, compliance, or troubleshooting purposes.
- User Authentication and Authorization: To prevent unauthorized access, notification systems often include authentication and authorization mechanisms to control who can configure and receive alerts.

The features of a notification alarm system are designed to enhance communication, situational awareness, and response efficiency in various contexts, including emergency management, security, and operational monitoring.

6.2.1 WhatsApp Notification Alarm System

WhatsApp primarily serves as a messaging platform, and it does not have built-in features to function as a comprehensive notification alarm system. However, developers and businesses have found ways to leverage WhatsApp for notification purposes. WhatsApp can be utilized for notification purposes:

- API Integration: Businesses can use the WhatsApp Business API to send automated messages and notifications to customers. This is typically used for transactional messages, order updates, and customer support.
- Third-Party Services: Some third-party services provide integrations between notification systems and messaging platforms, including WhatsApp. These services may allow you to send alerts or notifications to WhatsApp contacts or groups.
- Custom Solutions: Developers can create custom solutions using WhatsApp's APIs or by automating interactions through bots. This might involve programming scripts or using third-party platforms that facilitate integration.
- Alerts and Reminders: You can set up manual alerts or reminders within WhatsApp groups or individual chats. While this is more manual and lacks the automation features of a dedicated notification system, it can still serve as a rudimentary form of communication.
- Event-triggered Messages: If your organization uses a system that can trigger messages based on specific events (e.g., server issues, security alerts), you may explore the possibility of sending these notifications to WhatsApp through custom integrations.

6.2.2 WhatsApp, CallMeBot and Home Assistant

CallMeBot [15] is a service that provides a simple **API** for sending WhatsApp messages and making WhatsApp calls. Home Assistant users can leverage this service to receive notifications or make calls through WhatsApp. Refer to Figure 6.3 to get a basic visual representation of its working principle.

Figure 6.3: WhatsApp text messages from Home Assistant using CallMeBot API

Once the CallMeBot integration is configured, you can set up automations or scripts in Home Assistant to trigger WhatsApp calls or messages. Here are some potential advantages of using the CallMeBot.

- Ease of Integration: CallMeBot offers a straightforward API that is easy to integrate with various platforms and services. Users can quickly set up WhatsApp notifications or calls without complex configurations.
- WhatsApp Integration: CallMeBot leverages WhatsApp, a widely used messaging platform. This allows users to receive notifications and calls directly through WhatsApp, which is convenient for individuals who are already familiar with the platform.
- No Additional App Required: Since CallMeBot uses WhatsApp, recipients do not need to install a separate app to receive notifications or calls. Many users already have WhatsApp installed on their smartphones, making it a convenient and accessible communication channel.
- Multi-Platform Support: CallMeBot's API can be integrated with various platforms and services, making it versatile for different use cases. It can be utilized with home automation systems, monitoring tools, and other applications.

- Free Tier: CallMeBot offers a free tier that allows users to send a limited number of messages or make a limited number of calls each day. This is beneficial for individuals or small-scale applications with modest notification needs.
- Flexibility in Notification Types: Users can send both messages and calls through CallMeBot, providing flexibility in how notifications are delivered. Depending on the urgency or importance of the alert, users can choose the appropriate notification type.
- Integration with Home Automation Systems: Home automation enthusiasts often find CallMeBot useful for integrating with platforms like Home Assistant. This allows for automation of notifications based on events or triggers in a smart home environment.
- API Documentation: CallMeBot provides clear and accessible API documentation, making it easier for developers to understand how to integrate the service into their applications or systems.

It is important to note that the advantages of using CallMeBot may be context-dependent, and the service may be more suitable for certain use cases than others. For our project, the CallMeBot is a good choice because the caregivers or neighbors can be notified when a fall is detected with a WhatsApp text message since this application is widely used in recent days.

7
Results and Conclusion

7.1 Results

7.1.1 Sensor Data Management and Analysis

Using InfluxDB (create a database) and Grafana (data visualization ans analysis), sensor data was stored and represented graphically for further analysis of fall detection. The data of entities of the Aqara Presence Sensor, Tapo Motion Sensor 1 and Tapo Motion Sensor 2, flowing from the database created in InfluxDB to Grafana, was graphically represented for visual analysis. The data from these three sensors were collected for 2 hours in order to analyse the events occurred in this time period. The events include motion/walking/standing/sitting/bending and a dangerous fall on **Detection Zone 1**.

From Figure 7.1 we observe:

- The data was collected for about two hours between the time 11:20 to 13:15.
- **"On"** represents the sensor in **"detected"** state and **"Off"** represents the sensor in **"clear"** state.
- From the testing and the Proof-of-Concept, we know that if Tapo Motion Sensor High/1 is in "clear" state while Aqara FP2 Detection Zone 1 and Tapo Motion Sensor Low/2 are in "detected" state, a fall has occurred.

- Further analyzing the three panels on the dashboard in Figure 7.1, we observe that a fall has occurred at time 11:32, 11:53, 12:23, 12:35 and 1:08 based on the Proof-of-Concept. At these instances, the Motion Sensor High/1 is in "off" state while Aqara FP2 Detection Zone 1 and Tapo Motion Sensor Low/2 are in "on" state indicating a **fall**

Figure 7.1: Sensors Data Grafana Dashboard

7.1.2 Alarm System

An alarm system was developed using the Home Assistant and CallMeBot API service in order to trigger alarms when a fall was detected. The alarm was automated using triggers, conditions and actions. The Alarm system consisted of:

- An Alarm Panel
- A Siren
- A WhatsApp notification

Application scenario:

Consider a scenario where a caregiver wants to step out for a couple of hours to run some errands leaving an elderly person alone in the room. Before she/he leaves the house, they will have to change the state of the configured Alarm Panel from **"Disarmed"** to **"Armed Home"** by simply clicking on the **ARM HOME** as seen in Figure 8.10 and Figure 8.11. When the elderly person suddenly encounters a dangerous fall on the Detection zone 1 of the room, the Alarm Panel will get triggered. As soon as the panel is triggered, the siren begins to ring to alert the neighbors, and the caregiver will immediately receive a pre-configured WhatsApp message from the Home Assistant via the CallMeBot.

Figure 7.2 and Figure 7.3 represent the Home Assistant dashboard with the Alarm Panel being triggered since the Aqara Sensor Detection Zone 1 is in "detected" state, the Tapo Motion Sensor High/1 changed from "detected" to "clear" state and the Tapo Motion sensor Low/2 is in "detected" state. Therefore, a fall was detected as per the proof-of-concept leading to triggering the alarms. Once the alarm panel is triggered the siren begins to ring, alerting caregivers and the neighbors for immediate assistance.

In case of a false alarm, alarm can be disarmed by entering the code 1234 as programmed in the YAML file. The siren can also be turned off manually on the home Assistant dashboard. It is evident that a fall has occurred at time 11:32, 11:53, 12:23, 12:35 and 1:08. At these instances the Alarm Panel was triggered and a WhatsApp message was sent to a pre-configured phone number as seen in Figure 7.4.

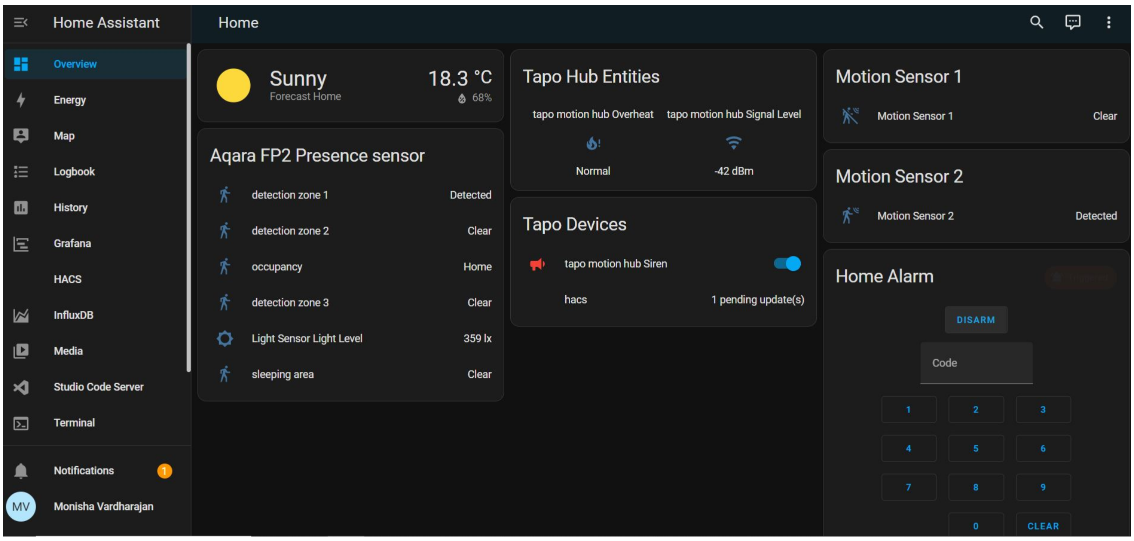

Figure 7.2: Alarm Panel Triggered

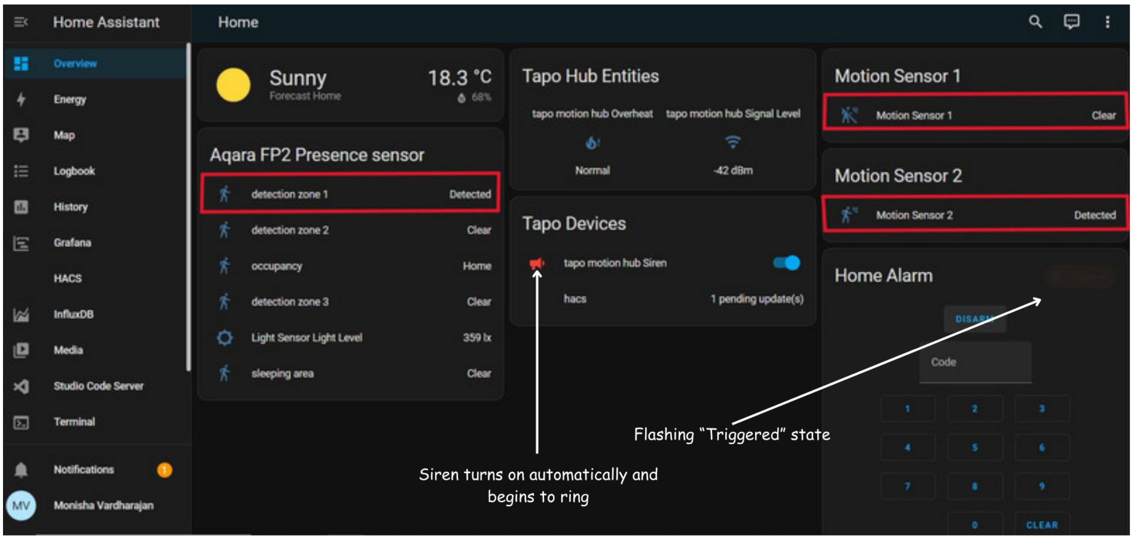

Figure 7.3: Alarm Panel Triggered state

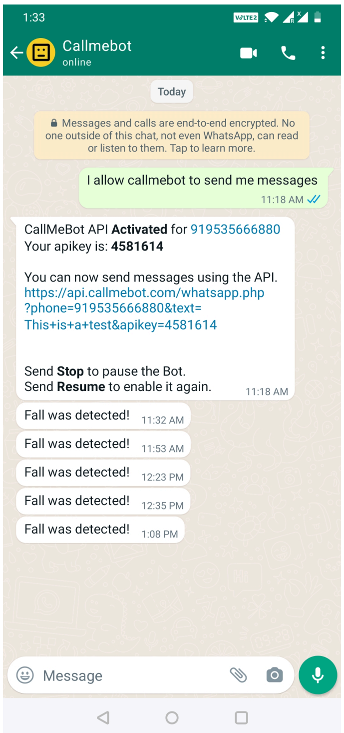

Figure 7.4: WhatsApp Notification from Home Assistant via the CallMeBot service

7.2 Conclusion

In conclusion, this thesis has presented a robust proof of concept for a fall detection system and accompanying an alarm system leveraging low-cost IoT sensors. The primary objective of this research was to explore the feasibility and efficacy of employing affordable sensors to enhance elderly care and safety.

Throughout the study, we successfully developed, implemented, and tested a prototype system capable of detecting falls with a good level of accuracy. The integration of Home Assistant with the low-cost IoT sensors proved to be instrumental in capturing relevant data, enabling the system to distinguish between normal activities and a potential fall incident.

The system's alarm functionality demonstrated its responsiveness in promptly notifying caregivers and the neighbors in the event of a fall. This feature addresses the critical need for timely intervention, potentially reducing the severity of injuries sustained during a fall.

Moreover, the cost-effectiveness of the IoT sensors used in our system opens up new possibilities for widespread adoption in both healthcare facilities and home environments. The affordability of the components not only makes the system economically viable but also scalable, paving the way for broader implementation and accessibility.

The results obtained in our experimentation phase indicate promising prospects for real-world deployment. Nevertheless, it is important to acknowledge the limitations of our study, including the need for further validation in diverse scenarios. Future research should focus on refining this fall detection system by expanding the usage of more IoT devices and conducting trials in varied environments to enhance the system's accuracy and reliability .

In summary, this thesis has contributed a valuable proof of concept for a fall detection and alarm system that leverages low-cost IoT sensors. The potential impact of such a system on enhancing the safety and well-being of the elderly population is substantial. As technology continues to advance, the integration of IoT sensors in healthcare applications holds great promise for improving the quality of life for individuals at risk of falls.

8

Configuration Setup: A Step-by-Step Approach

8.1 Home Assistant Operating System

Required hardware:

1. Raspberry Pi 4 or Raspberry Pi 3
2. Power Supply for Raspberry Pi 4 or Power Supply for Raspberry Pi 3
3. Micro SD Card.A 32 GB or bigger card is recommended
4. SD Card reader
5. Ethernet cable is required for installation. After installation, Home Assistant can work with Wi-Fi, but an Ethernet connection is more reliable and highly recommended.

Installation of Home Assistant Operating System onto Rapsberry Pi 4 using Raspberry Pi imager [16]

1. Download and install the Raspberry Pi Imager on your computer as described under https://www.raspberrypi.com/software/

2. Write the image to your SD card by downloading and installing the RaspberryPi Imager online from the above link
3. Open the Raspberry Pi Imager
4. Choose the OS > Other specific Purpose OS > Home assistants and home
5. automations > Home Assistant
6. Choose the storage > select your SD card
7. Write the installer onto the SD card
8. Eject the SD card
9. Insert it into your Raspberry Pi
10. Power up Pi
11. Plug ethernet cable for the first initial setup
12. Reach out to your Home Assistant at homeassistant.local:8123 or http://X.X.X.X:8123 where X.X.X.X is the IP address of the raspberry pi

You will now be allowed to change the credentials of your home assistant. Once done, you will be able to view the Home assistant dashboard.

To set up Wi-Fi you will need to use the Home Assistant Community Add-on: Advanced SSH & Web Terminal.
This add-on allows you to log in to your Home Assistant instance using SSH or a Web Terminal, giving you the option to access your folders and also includes a command-line tool to do things like restart, update, and check your instance. This is an enhanced version of the provided SSH add-on by Home Assistant and focuses on security, usability, flexibility and also provides access using a web interface.

Setting up the Wi-Fi [17]

1. Install SSH & Web terminal Add-on from the Community Repository
2. Set protection mode

3. Change the SSL option to false if you are not using SSL

4. Save and start Add-on

5. Open the Web UI to use the web terminal

Now run the following nmcli commands to configure the WiFi:
nmcli radio
nmcli device wifi rescan
nmcli device wifi
nmcli device wifi connect "SSID" password "WI-FI PASSWORD"
nmcli con show

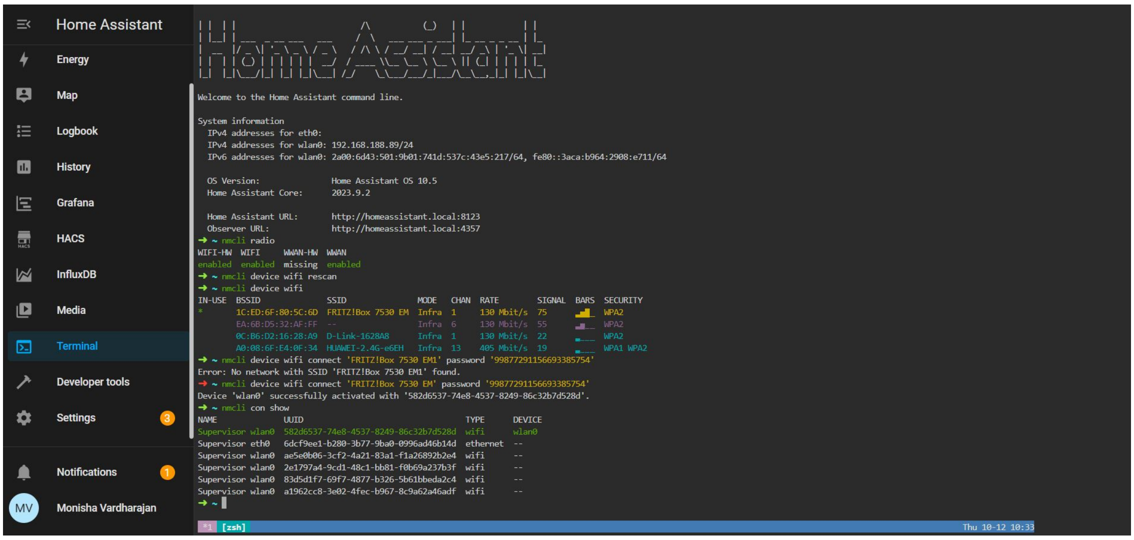

Figure 8.1: Wi-Fi Configuration

You can now verify if the Wi-Fi is connected, by clicking on Setting > System > Network > WLAN0 > Wi-Fi. Also, you can set up your Host Name,the name your instance will have on your network. In my case, I have named my **Host Name** as **homeassistant**.

8.2 Aqara Presence Sensor FP2

The FP2 is a Wi-Fi device unlike the the FP1 which is a zigbee device. Hence FP2 does not require an additional hub to pair. Wi-Fi also does not mean it is dependent on the cloud. All automations can be done locally ensuring uninterrupted operation even without internet connection.

Setting up the sensor on the Mobile App:

1. Download [18] and install the **Aqara** app from playstore or appstore
2. Wall mount the sensor before your start to pair.
3. Pair the sensor using 2.4Ghz WiFi (does not work with 5Ghz).
4. Define the working mode, zones and the edges. ReferFigure 8.2
5. Set the sensitivity of the detection zone. For my project I have set the detection zone 1 with high sensitivity.

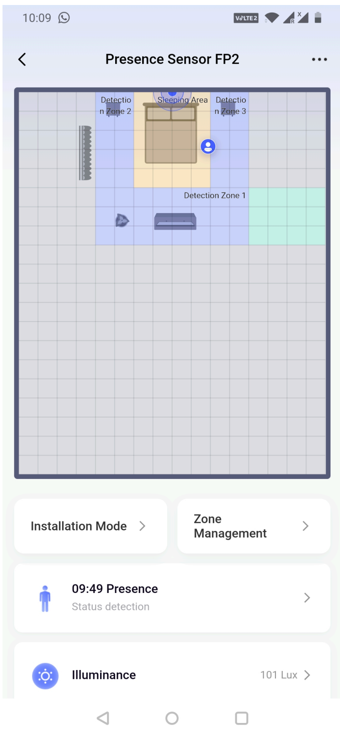

Figure 8.2: Aqara zone management

Integrating the FP2 sensor with Home Assistant [19]:
The HomeKit Device integration allows you to connect accessories which works with HomeKit logo to Home Assistant. This integration should not be confused with the HomeKit Bridge integration, which allows you to control Home Assistant devices via HomeKit.

To add a HomeKit device (Aqara Presence sensor FP2) via Ethernet or Wi-Fi :

1. Make sure your device is powered up.
2. Make sure the device is on your network, but not paired with another HomeKit controller. If your device is in your network but is paired with an Apple device via HomeKit, remove the device from the Apple Home app. Otherwise it will not be able to pair with Home Assistant.
3. The device should have been discovered under Settings > Devices & Services.
4. On Home Assistant, Go to Setting > devices & services > Add Integration > Apple > HomeKit > HomeKit Device > Select your presence sensor > Pairing code (present at the back of the device) > Configure.
5. The device will now be added to your Home Assistant instance.

8.3 Tapo Hub H100 & Tapo Motion Sensors T100

Setting up the hub and the sensors on the Mobile App:

1. Download [20]the Tapo app from playstore or appstore. Create an account and login.
2. Power up the Tapo Hub and pair using 2.4Ghz WiFi.
3. Add the hub device and wait for pairing mode, LED lights blink orange and blue alternatively.
4. Go to your Wi-Fi settings and connect to your hub Tapo.Hub.XXXX
5. Connect your hub to your home 2.4Ghz Wi-Fi.

6. Power up the Tapo sensors and add the sensors by pairing them to the hub. These sensors are now the child devices of the Hub. Open the dashboard to see the log events. Refer Figure 8.3

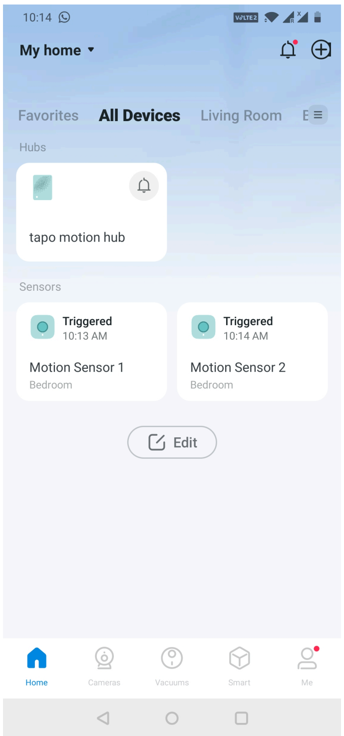

Figure 8.3: Tapo App dashboard

Integrating the Tapo Hub H100 with Home Assistant [21]:
Home Assistant Community Store, or HACS, is a powerful integration for Home Assistant that allows users to download and install custom add-ons, integrations, themes, elements, etc.

To integrate the Tapo devices via the HACS:

1. Go to devices and services and install the HACS integration and restart the HA.

2. Open HACS from the sidebar and integrate this custom repository from github to custom integrate the Tapo motion sensors using the Tapo Controller.This is a custom integration to control Tapo devices from home assistant.[22]

3. Now add and install the Tapo Controller integration to Home Assistant.
4. Restart Home Assistant.
5. Add the integration on your HA to setup your hub and child devices.
6. To add devices to your Tapo controller you will need the IP address of the Hub, username and password credentials of your tapo app on the phone. You can find the IP address of the hub on the mobile app. Tap on Device info and then you can find the IP address of your Hub. Also, you can login to your router's web interface to look for the IP address allocated to the hub.

Note : Once the hub is integrated with Home Assistant, the child devices and their corresponding entities will sync up automatically. The child devices poll high frequency to the hub to capture events although this generates a lot of unnecessary network traffic this was the only improper solution to get the real time events captured.

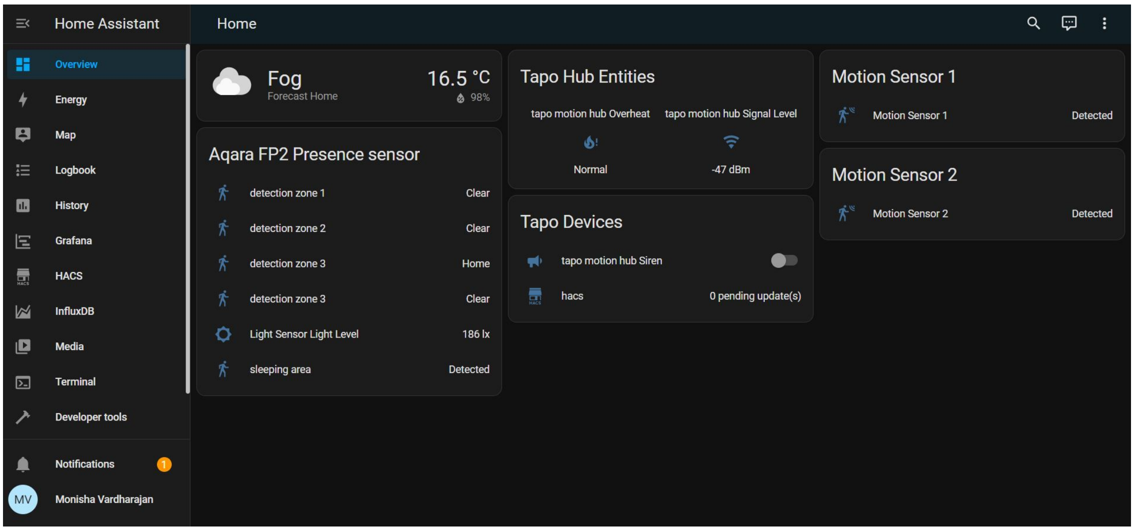

Figure 8.4: Home Assistant dashboard with entities

The Figure 8.4 is my Home Assistant dashboard after the Aqara Presence FP2 Sensor, Tapo Hub H100 and the two Tapo Motion Sensors T100 are all integrated with Home Assistant.

8.4 InfluxDB

Installation

1. The installation of this add-on is pretty straightforward and not different in comparison to installing any other Home Assistant add-on.
2. Open the add-on on your Home Assistant instance.
3. Search for the InfluxDB add-on in your Home Assistant instance.
4. Click the Install button to install the add-on.
5. Start the InfluxDB add-on.
6. Check the logs of the InfluxDB to see if everything went well.
7. Click the OPEN WEB UI button.
8. Restart the HA.

Configuration

1. Click on OPEN WEB UI to open the admin web-interface provided by this add-on.
2. On the left menu click on the InfluxDB Admin.
3. Create a database for storing Home Assistant's data in, e.g., homeassistant.
4. The retention policy is set to infinite by default
5. Go to the users tab and create a user for Home Assistant, e.g., homeassistant.
6. Add ALL to Permissions of the created user, to allow writing to your database.
7. Open the Studio Code Server (install this add-on).
8. Add this snippet to the yaml configuration file Figure 8.5
9. Restart HA
10. You should now see the data flowing into InfluxDB by visiting the web-interface and using the Data Explorer.

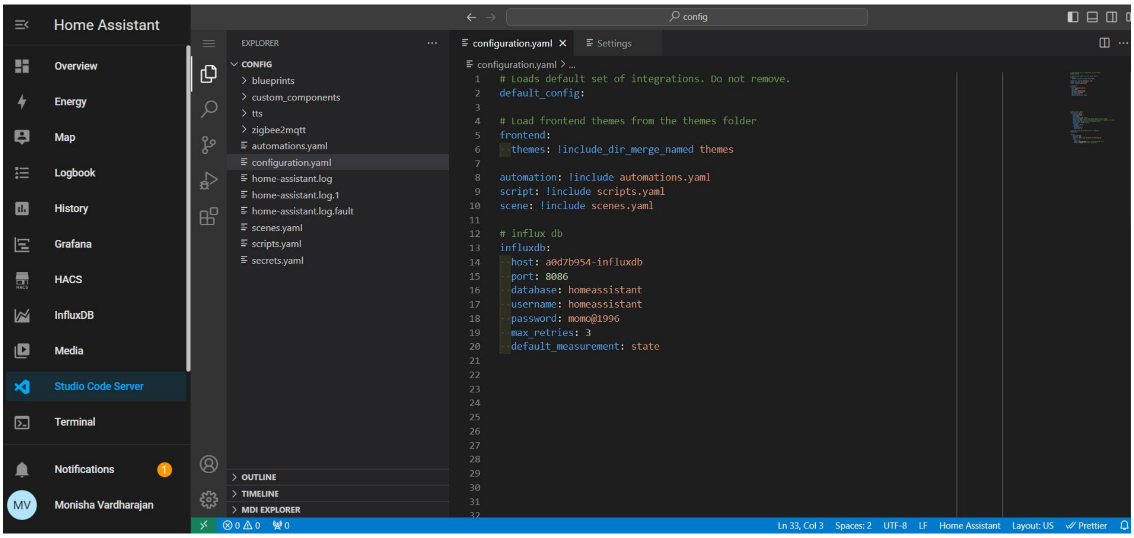

Figure 8.5: InfluxDB YAML snippet

8.5 Grafana

Installation and Configuration

1. Search for Granfana on your Home Assistant add-ons and open the add-on on your Home Assistant instance.
2. Click the Install button to install the add-on.
3. Start the Grafana add-on.
4. Check the logs of the Grafana to see if everything went well.
5. Open the Web UI.
6. You can add your own visualization by editing the panel and create your own dashboard to monitor and analyse the data. Refer Figure 8.6.
7. Click on the Create icon in the sidebar and select Dashboard.
8. Click on the Add new panel button and select Graph.
9. Click on Panel title.
10. In the Query tab, select the InfluxDB data source that you configured
11. Enter your InfluxDB query in the Query editor.

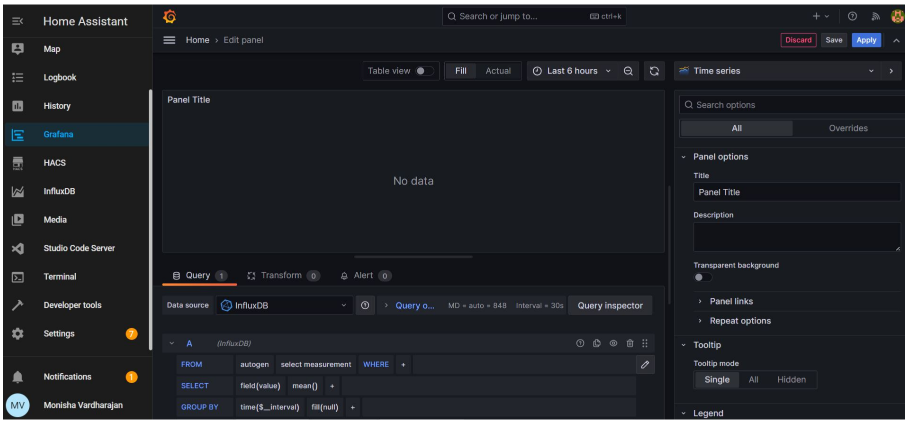

Figure 8.6: Grafana Data Visualization

8.6 Visualization and Analysis of Sensor Data

After creating a database using InfluxDB, a default **DB Retention policy** as seen in Figure 8.7 is automatically created as **homeassistant.autogen**. Next under Measurements & tags > state > entity.id , select the aqara fp2 presence sensor that corresponds to the detection zone 1 (in my case it was fp2 presence sensor 3), motion sensor 1 and motion sensor 2. These are the three entities used for the analysis and detection of a dangerous fall. Next under fields > choose value since that represents the state of the sensors Figure 8.8.

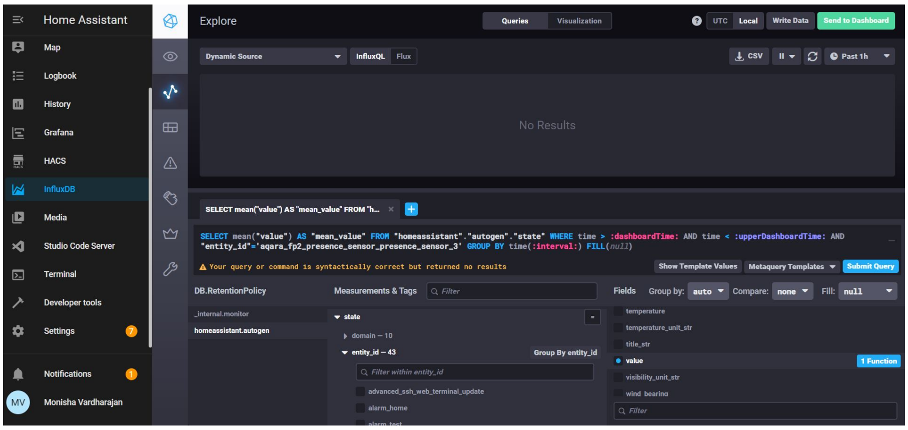

Figure 8.7: InfluxDB queries 1

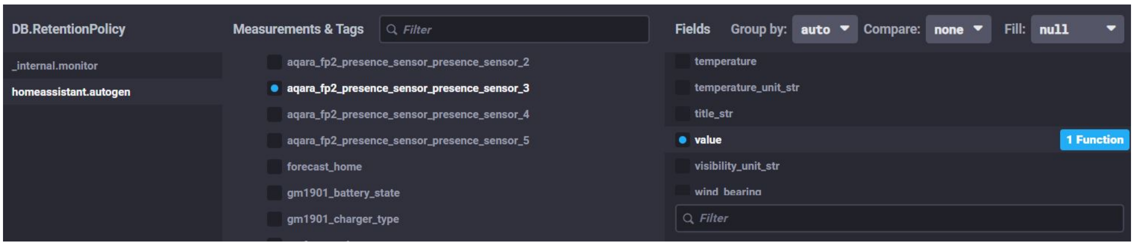

Figure 8.8: InfluxDB queries 2

To visualize all the sensor data using graphs on a singe dashboard :

1. Open Grafana Web UI.
2. Create a dashboard giving it a name **Sensors data**.
3. Refer Figure 8.6 , Begin to edit the panel in order to visualize the sensor data.
4. Choose the source as InfluxDB since this is where all our sensor data is stored.
5. Choose the following options > autogen > state ; where > entity_id::tag > [exact entity of the sensor] ; field(value) > distinct ; time($_interval) > 1s.
6. Finish editing the panel name, legend etc. And save to the dashboard **Sensors data**
7. For this project I have created 3 panels, one for the Aqara FP2 detection zone 1, one for motion sensor 1 and lastly one for motion sensor 2 and added all the three to my dashboard.

Exact entity of the sensor in my case was as follows:

1. Aqara Presence Sensor FP2 for detection zone 1 > aqara_fp2_presence_sensor_presence_sensor_3
2. Tapo motion sensor 1 > motion_sensor_1
3. Tapo motion sensor 2 > motion_sensor_2

8.7 The Alarm Panel

To create an alarm panel [23] on Home Assistant dashboard :

1. Open the Studio code server and add the code snippet in the configuration YAML file [24] as in Figure 8.9.

2. Save and restart Home Assistant.

3. To add the panel, click on the three dots in the top right corner of the dashboard and choose edit dashboard. Add card and select the default alarm panel.

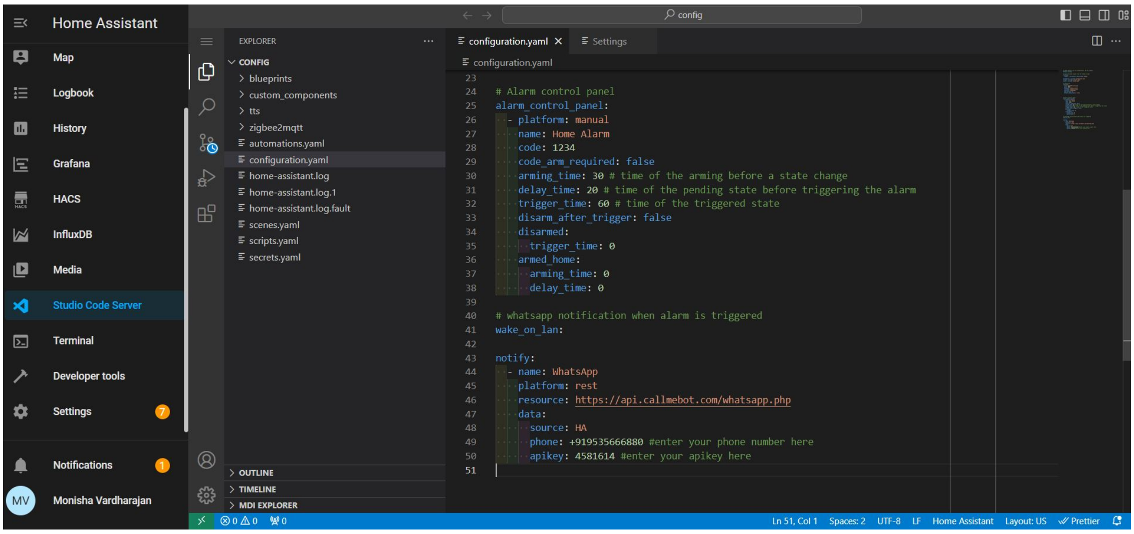

Figure 8.9: YAML snippet code

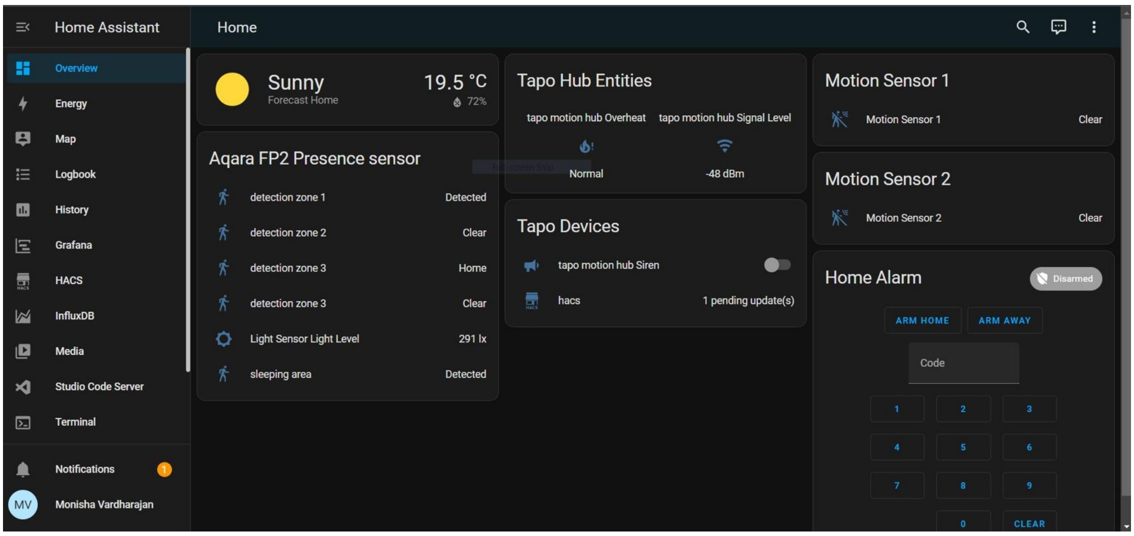

Figure 8.10: Home Assistant dashboard with alarm panel

Figure 8.10 shows the Home Assistant dashboard with the alarm panel. It consists of two entities, ARM HOME and ARM AWAY.

8.8 WhatsApp notification from Home Assistant

Home Assistant can send WhatsApp text messages using the CallMeBot Free API [15]. Home Assistant executes a WGET or CURL command to send WhatsApp Messages.

Implementation:

1. To get the APIKEY from the bot before using the API, save the phone number +34 644 99 26 98 (might change in the future) into your Phone Contacts.
2. Send this message "I allow callmebot to send me messages" to the new contact created.
3. Wait until the API is activated and you receive the APIKEY.
4. On Home Assistant go to studio code server > YAML file > insert the snippet code as shown in Figure 8.9. This creates a new REST Notify service in the configuration.yaml file.
5. Save and restart Home assistant.

8.9 Automation and scenes

To create an automation [25] in order to trigger the alarm panel:

1. Go to settings > automations & scenes > create automation > create a new automation.
2. Configure the triggers, conditions and actions.
3. Rename and Save the automation.
4. The alarm panel will now be triggered if the conditions are all true. When the alarm panel is triggered, the siren automatically begins to ring and a whatapp notification is sent to my phone.

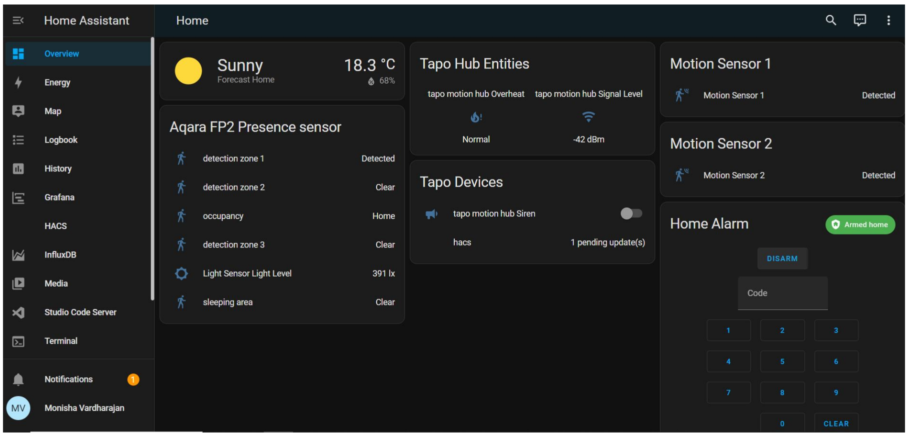

Figure 8.11: Alarm Panel with ARMED HOME state

For the fall alarm system, the following were automated:

1. **Triggers**
   In this project three Triggers are used - Aqara Detection Zone 1, Tapo Motion Sensor 1 and Motion Sensor 2.

2. **Conditions**
   Under conditions I have created a nested **AND** condition where it first checks for the state of Tapo motion sensor 1 followed by Aqara Detection Zone 1 and then Tapo Motion Sensor 2. The alarm panel will be triggered only when the state of Tapo motion sensor 1 changes from "detected" to "clear" state, Aqara Detection Zone 1 is in "detected" state and Tapo Motion Sensor 2 is also in "detected" state. Additionally, I have added one more condition in which only when the alarm panel is in "Armed Home" state the alarm panel should be triggered. This comes in use when the I have step out, I can change the state to "Armed Home", refer to Figure 8.11. This will trigger alarms only when there is no assistance and avoids alarm triggers when the elderly is under supervision. So only when all the four conditions hold true the alarm panel is triggered.

3. **Actions**
   I have called three services to build this alarm system. Firstly, when all the conditions are satisfied the alarm panel is in "triggered" state. Secondly, a siren rings when the alarm panel is in "triggered" state. Call a service Siren: "Turn on" and add the siren entity of the Tapo Hub H100. Thirdly, a WhatsApp notification is sent to my phone when the alarm panel is triggered. Call a service with Notifications: "Send a notification with whatsapp" and enter a message (Fall was detected!) that needs to be sent to my phone. A message that says Fall was detected! will be sent to my phone.

9

Appendices

9.1 Appendix A : Home Assistant Core

The Home Assistant Core Figure 9.1 consists of four main parts. On top of this it includes many helper classes to deal with common scenarios, like providing an entity or dealing with locations.

- Event Bus: facilitates the firing and listening of events – the beating heart of Home Assistant.

- State Machine: keeps track of the states of things and fires a state_changed event when a state has been changed.

- Service Registry: listens on the event bus for call_service events and allows other code to register services.

- Timer: sends a time_changed event every 1 second on the event bus.

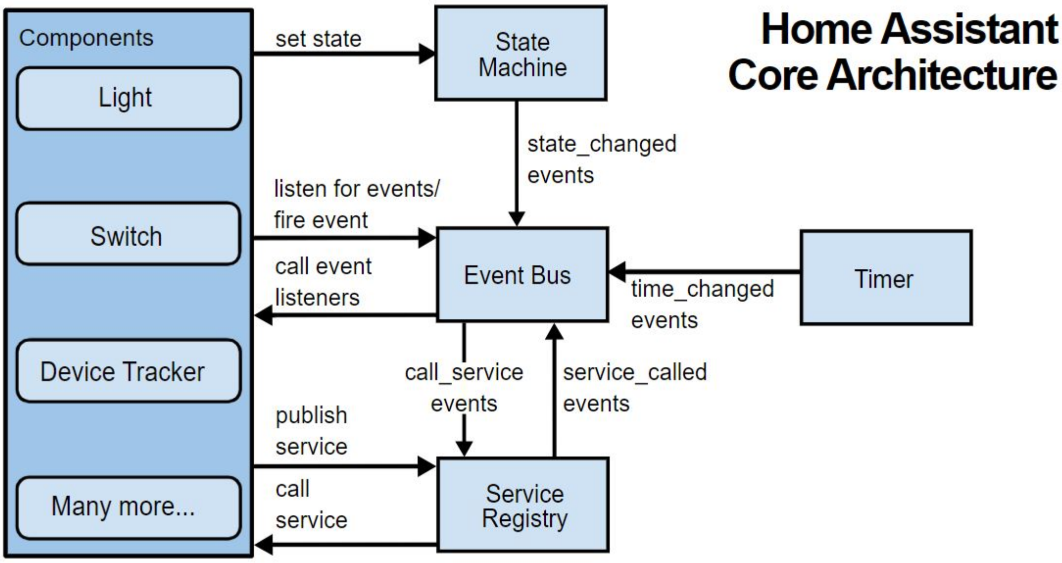

Figure 9.1: HA Core Architecture

Home Assistant Core [26] can be extended with integrations Figure 9.2. Each integration is responsible for a specific domain within Home Assistant. Integrations can listen for or trigger events, offer services, and maintain states. Integrations are made up of a component (the base logic) and platforms (bits that integrate with other integrations). Integrations are written in Python and can do all the goodness that Python has to offer. Out of the box, Home Assistant offers a bunch of built-in integrations.

Home Assistant distinguishes the following integration types:

1. Define an Internet of Things domain : These integrations define a specific device category of Internet of Things devices in Home Assistant, like a light. It's up to the light integration to define what data is available in Home Assistant and in what format. It also provides services to control lights.

2. Interact with external devices & services : These integrations interact with external devices & services and make them available in Home Assistant via integrations that define IoT domains like light. An example of such an integration is Philips Hue. Philips Hue lights are made available as light entities in Home Assistant.

3. Represent virtual/computed data points : These integrations represent entities either based on virtual data, like the input_boolean integration, a virtual switch. Or they derive their data based on other data available in Home Assistant, like the template integration or utility_meter integration.

4. Actions that can be triggered by the user or respond to events [27] : These integrations provide small pieces of home automation logic that do common tasks within your house. The most popular one is the automation integration, allowing users to create automations through a configuration format.

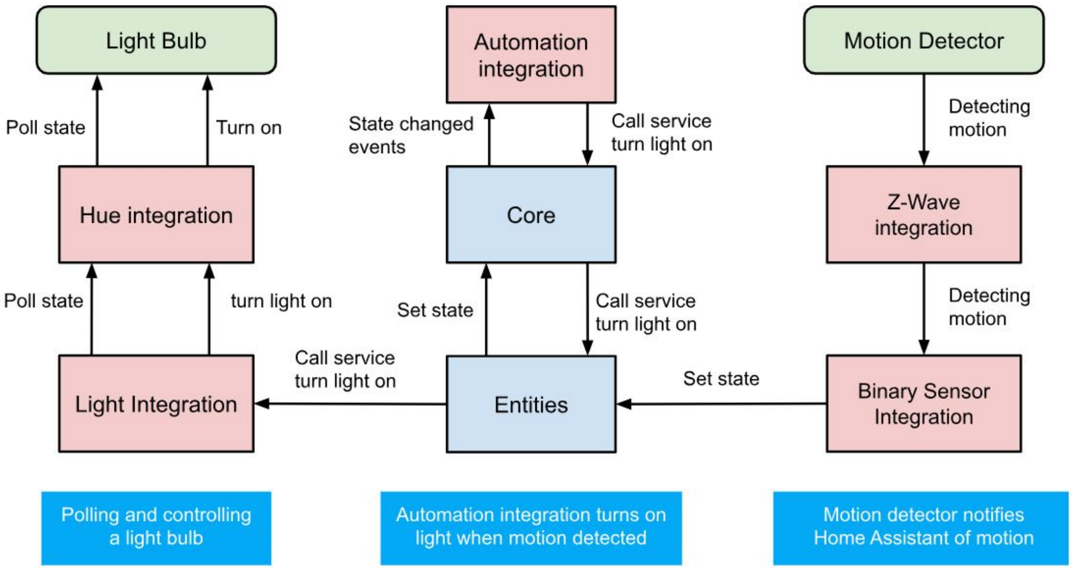

Figure 9.2: HA Integration Architecture

Entities: Integrating devices & services refer Figure 9.3.
Integrations can represent devices & services in Home Assistant. The data points are represented as entities [28]. Entities are standardized by other integrations like light, switch, etc. Standardized entities come with services for control, but an integration can also provide their own services in case something is not standardized.
An entity abstracts away the internal working of Home Assistant. As an integrator you don't have to worry about how services or the state machine work. Instead, you extend an entity class

and implement the necessary properties and methods for the device type that you're integrating.
Configuration is provided by the user via a Config Entry or in special/legacy cases via configuration.yaml.

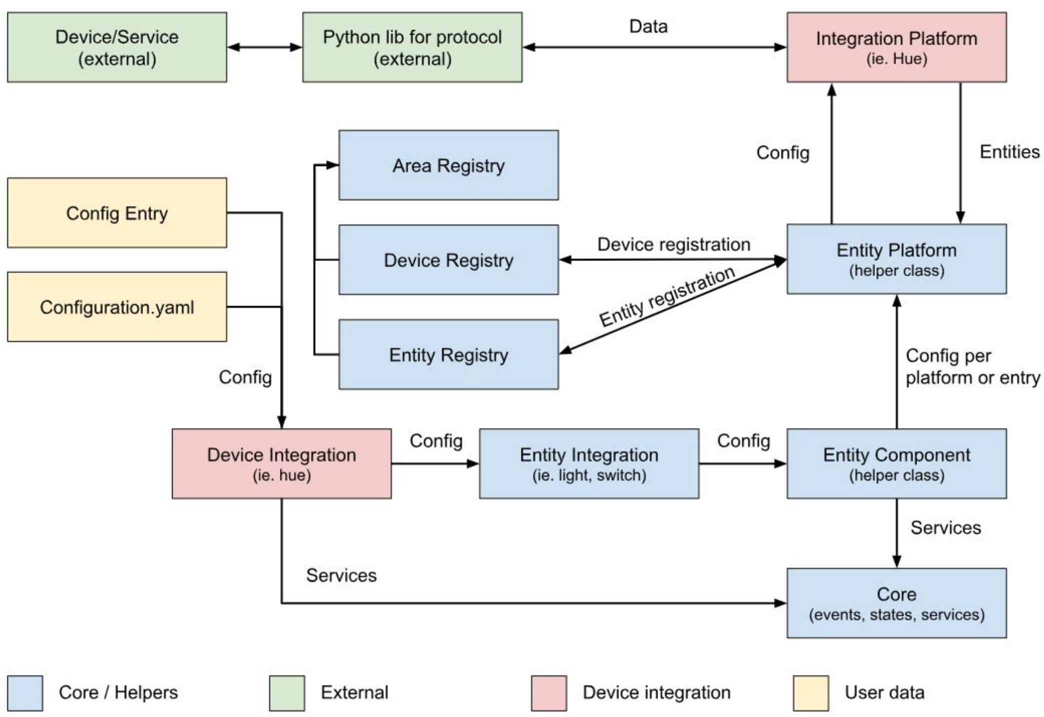

Figure 9.3: Entities: integrating devices & services

Entity interaction with Home Assistant Core refer Figure 9.4.
The integration entity class that inherits from the entity base class is responsible for fetching the data and handle the service calls. If polling is disabled, it is also responsible for telling Home Assistant when data is available. The entity base class (defined by the entity integration) is responsible for formatting the data and writing it to the state machine. The entity registry will write an unavailable state for any registered entity that is not currently backed by an entity object.

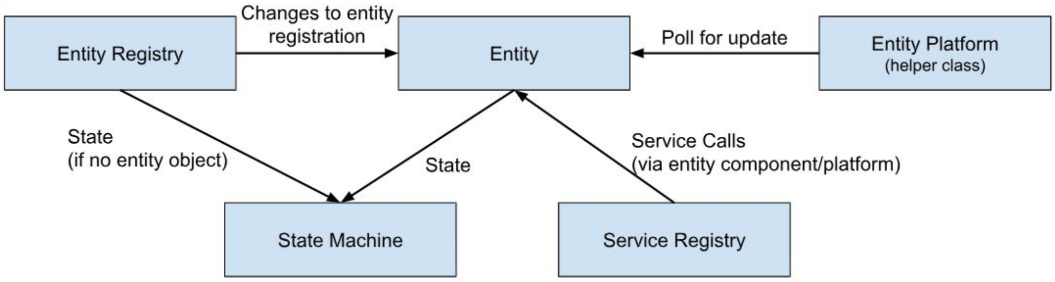

Figure 9.4: Entity interaction with Home Assistant Core

9.2 Appendix B : Practical Recommendation

Configuration of Home Assistant and IoT devices at **HOME**

- The Aqara Presence sensor fp2 and the Tapo Hub can be paired only with 2.4 Ghz of channel.

- The routers at home would generally support both 2.4 Ghz and 5 Ghz bands. However, it is necessary to split the two bands by giving it different SSIDs. To do this, go to the router's web interface and login using the router's credentials. Select the option that says separate bands and give the SSID for the 2.4 Ghz band. Refer Figure 9.5.

- Pair the Aqara Presence sensor fp2 and Tapo Hub using the 2.4 Ghz Wi-Fi band.

- You could use an ethernet cable for initial configuration of the home assistant and then configure Wi-Fi for your home assistant so every time the raspberry pi is powered on, it can automatically connect to the Wi-Fi.

- An important point to be noted is that all the IoT devices and the home assistant need to be connected to the same Home LAN. Refer Figure 9.6 and Figure 9.7.

  Note : The IoT devices were paired using the mobile Apps and the 2.4 Ghz Wi-Fi. Also, the IoT devices were integrated with Home Assistant using this home setup.

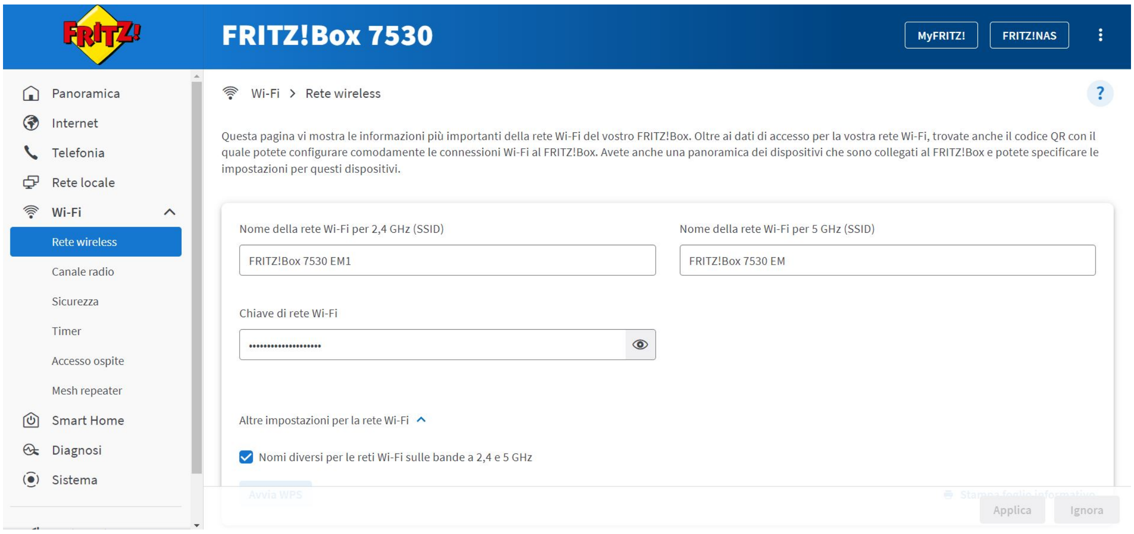

Figure 9.5: Separate SSID for 2.4 and 5 Ghz bands

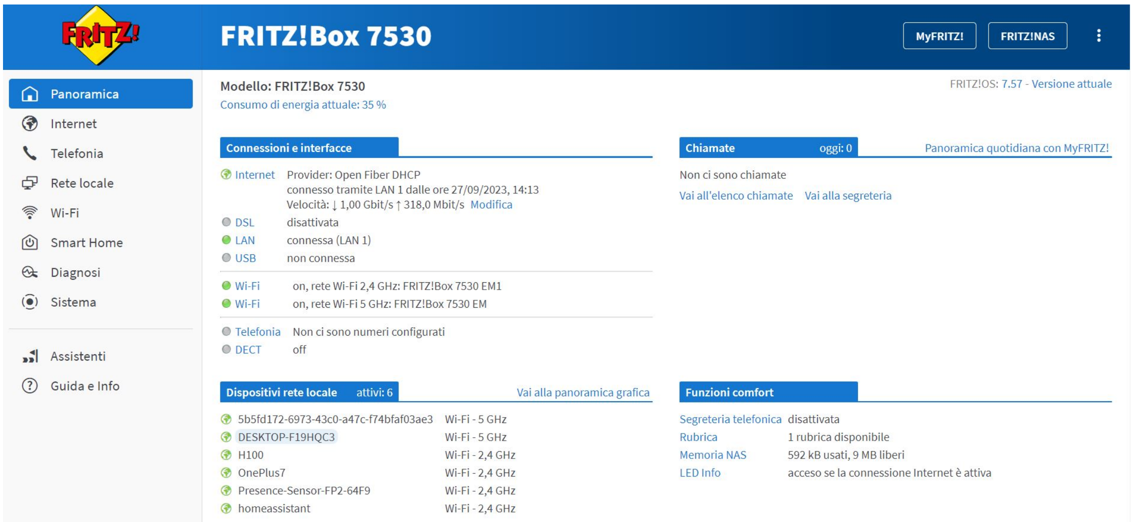

Figure 9.6: HA and IoT devices on the same network

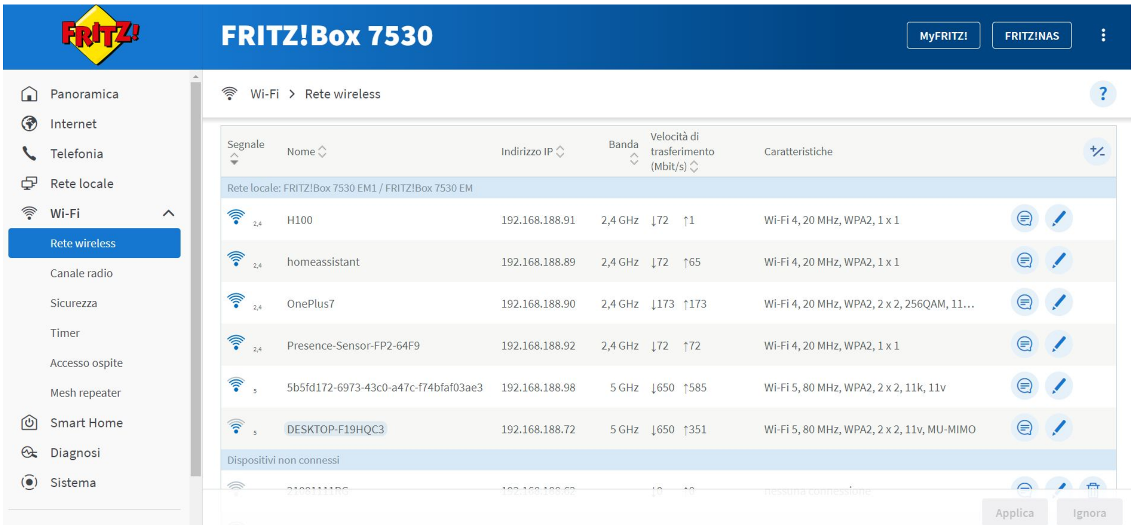

Figure 9.7: HA and IoT devices on the same network with IP addresses

Configuration of Home Assistant and IoT devices at **LAB**

- The Raspberry pi and my desktop were configured with a static IP address
- My desktop was made to act as a hotspot with a 2.4 Ghz band in order to connect the IoT devices.
- A switch was connected to the wall port stud. The Raspberry Pi and my desktop were connected to the switch via ethernet cables.

Note : The IoT devices were paired using the mobile App and the hotspot but the IoT devices were not possible to be integrated with Home Assitant using this lab setup.

The router configuration at home is very much different from the router and access points configuration in lab. Refer Figure 9.8 and Figure 9.9

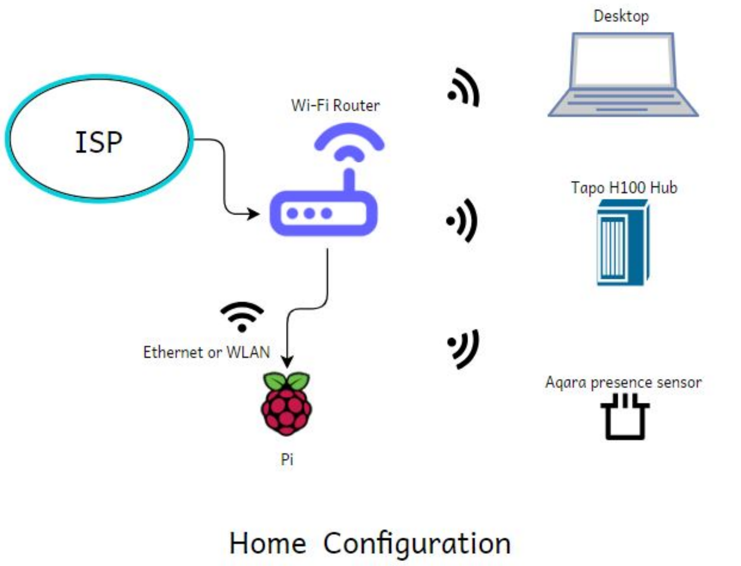

Figure 9.8: Home Setup

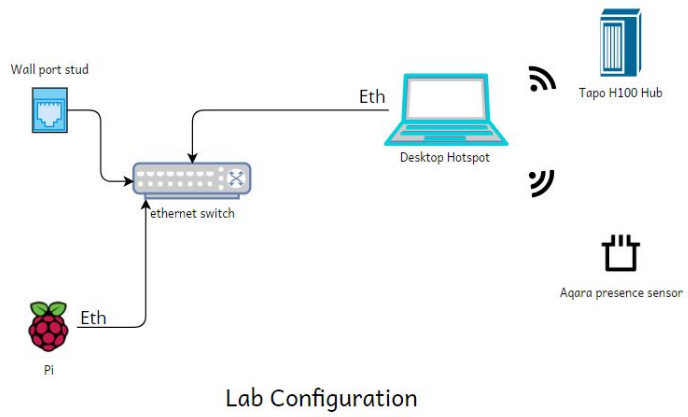

Figure 9.9: Lab Setup

Home Configuration:
Home routers typically use Dynamic Host Configuration Protocol (DHCP) to dynamically allocate IP addresses to devices within the local network. When a device connects to the router, it requests an IP address, and the router assigns an available IP address from its DHCP pool. The

allocation of IP addresses by a home router for devices remains the same even if the devices are disconnected due to the use of a lease-based system within the Dynamic Host Configuration Protocol (DHCP). The router's DHCP server assigns IP addresses to devices for a defined lease duration. This lease duration can vary but is typically several hours or days. During this lease period, the IP address is reserved for the device that received it, even if the device is disconnected.

Lab Configuaration:
In the case of a desktop serving as a hotspot with a static IP address, the desktop itself has a manually configured static IP address. This static IP address remains the same for the desktop at all times. When the desktop serves as a hotspot with a static IP, it can still run a DHCP server to allocate IP addresses to devices that connect to the hotspot. The IP address allocation to these devices is still lease-based. Each connected device receives an IP address for a defined lease duration, just as in a typical home router setup. The lease duration is determined by the DHCP server configuration on the desktop.

In the case of the desktop hotspot, the IP addresses [Figure 9.10] allocated to the IoT devices that were connected to it, constantly kept changing after getting disconnected while in the case router at home, the IP addresses [Figure 9.7] allocated to the IoT devices did not change for days. This maybe one of the possible reasons for not being able to integrate the sensors with Home Assistant at the lab.

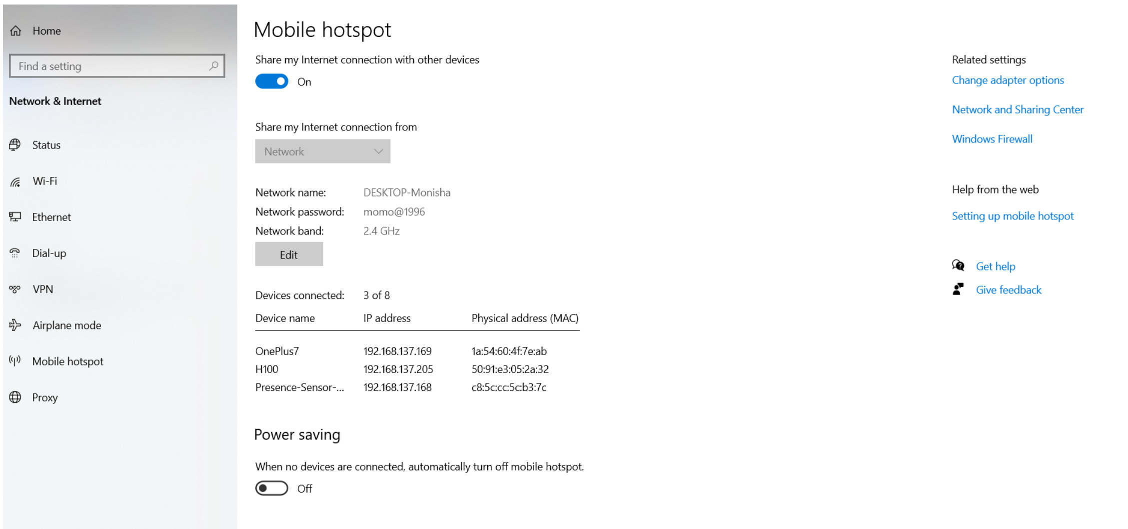

Figure 9.10: IP Addresses of IoT devices in the lab

In summary, both a home router and a desktop serving as a hotspot can utilize DHCP for IP address allocation to connected devices. The key difference is that the desktop serving as a hotspot uses a static IP for the desktop itself, while the router manages dynamic IP allocation for both the desktop and the connected devices. The **lease duration** for IP addresses allocated to connected devices is probably **more i.e days/months** with respect to a **home router** while the **lease duration** is **very short i.e changes every time the IoT devices get disconnected** in the case of the desktop hotspot. The hotspot was unstable and hence the IoT devices were automatically disconnecting and reconnecting to the hotspot network followed by changes in the IP Addresses. Therefore the Home Assistant was unable to discover the Aqara Presence sensor and when I tried to integrate the Tapo Hub using its IP address, it threw an error.

9.3 Appendix C : Yaml codes

 # To create a database using InfluxDB on Home Assistant

```
influxdb:
  host: a0d7b954-influxdb
  port: 8086
  database: homeassistant
  username: homeassistant
  password: momo@1996
  max_retries: 3
  default_measurement: state
```

 # To create an Alarm control panel on Home Assistant Dashboard

```
alarm_control_panel:
  - platform: manual
  name: Home Alarm
  code: 1234
  code_arm_required: false
  arming_time: 30 # time of the arming before a state change
```

```
  delay_time: 20 # time of the pending state before triggering the alarm
  trigger_time: 60 # time of the triggered state
  disarm_after_trigger: false
  disarmed:
    trigger_time: 0
  armed_home:
    arming_time: 0
    delay_time: 0
```

 # To send a WhatsApp notification when the alarm is triggered

```
wake_on_lan:

notify:
  - name: WhatsApp
  platform: rest
  resource: https://api.callmebot.com/whatsapp.php
  data:
    source: HA
    phone: +39XXXXXXXXXX #enter your phone number here
    apikey: XXXXXXX #enter your apikey here
```

References

[1] “ Need for Fall detection ,” https://www.forbes.com/health/medical-alert-systems/fall-detection-devices, accessed: 2023-11.

[2] “Low Cost IoT sensors ,” https://www.electronicsforu.com/technology-trends/economical-iot-sensors-for-the-developing-world, accessed: 2023-11.

[3] “ Advantages of Sensor-Based Fall Detection Systems ,” https://www.linkedin.com/pulse/6-advantages-sensor-based-fall-detection/, accessed: 2023-11.

[4] C. P. M. E. Diana Yacchiremaa, Jara Suárez de Pugaa, *Fall detection system for elderly people using IoT and Big Data*. Elsevier B.V, 2018.

[5] E. J. Wang X and A. G, *Elderly Fall Detection Systems:A Literature Survey*. Front. Robot. AI, 2020.

[6] F. Montenegro and S. Vélez, *Domotics in inclusion and health: Descriptive Study*. utn.edu.ec, 2023.

[7] K. O. G. S. R. J. C. L. B. O. Gutierrez, L.J.; Rabbani, *Internet of Things for Mental Health: Open Issues in Data Acquisition, Self-Organization, Service Level Agreement, and Identity Management*. MDPI, 2021.

[8] J. B. N. S. R. C. P. R. S. Failing, J.M.; Abellán-Nebot, *A Tool Condition Monitoring System Based on Low-Cost Sensors and an IoT Platform for Rapid Deployment*. MDPI, 2023.

[9] “ Home Assistant,” https://en.wikipedia.org/wiki/Home_Assistant, accessed: 2023-07.

[10] “ Aqara Presence sensor FP2,” https://www.aqara.com/en/product/presence-sensor-fp2/, accessed: 2023-07.

[11] “ Tapo hub and motion sensors,” https://www.tapo.com/en/product/smart-sensor/tapo-t100/, accessed: 2023-07.

[12] " Configure Home Assistant to export data to InfluxDB," https://blog.sandydrew.com/2022/08/home-assistant-influxdb-config, accessed: 2023-10.

[13] " Installation and configuration of grafana on Home Assistant," https://community.home-assistant.io/t/home-assistant-community-add-on-grafana/54674, accessed: 2023-10.

[14] "Notification alarm system ," https://www.dpstele.com/network-monitoring/alarm-notification-system.php, accessed: 2023-11.

[15] " Whatsapp notifications using Callmebot," https://www.callmebot.com/blog/whatsapp-text-messages-from-homeassistant/, accessed: 2023-10.

[16] " Setup Home Assistant using Raspberry Pi," https://www.home-assistant.io/installation/raspberrypi, accessed: 2023-05.

[17] " Configure WiFi settings on Home Assistant," https://community.home-assistant.io/t/guide-connecting-pi-with-home-assistant-os-to-wifi-or-other-networking-changes/98768, accessed: 2023-05.

[18] " Set up Aqara Presence fp2 sensor," https://www.aqara.com/wp-content/uploads/2023/06/Presence-Sensor-FP2_User-Manual.pdf, accessed: 2023-08.

[19] " Set up Aqara on HA," https://community.home-assistant.io/t/aqara-new-people-presence-sensor-fp2/545374, accessed: 2023-08.

[20] " Set up Tapo," https://www.tapo.com/en/product/smart-hub/tapo-h100/#tapo-product-setup, accessed: 2023-08.

[21] " Set up Tapo on HA," https://github.com/petretiandrea/home-assistant-tapo-p100/issues/354, accessed: 2023-09.

[22] " Tapo Support," https://github.com/petretiandrea/home-assistant-tapo-p100, accessed: 2023-10.

[23] " Alarm Panel card on Home Assistant," https://www.home-assistant.io/dashboards/alarm-panel/, accessed: 2023-10.

[24] " YAML for Home assistant," https://www.home-assistant.io/docs/configuration/yaml/, accessed: 2023-10.

[25] " Home Assistant automations," https://www.home-assistant.io/docs/automation/basics/, accessed: 2023-10.

[26] " Home Assistant Core for developers," https://developers.home-assistant.io/docs/development_index, accessed: 2023-10.

[27] " Home Assistant Core for developers about events," https://www.home-assistant.io/docs/configuration/events/, accessed: 2023-10.

[28] " Home Assistant Core for developers about entites," https://developers.home-assistant.io/docs/core/entity, accessed: 2023-10.

Acknowledgments

I would like to take this opportunity to express my heartfelt gratitude to all those who have been a part of my academic journey and have contributed to the completion of this thesis.

Firstly, I would like to express my gratitude to my supervisor, Professor Andrea Zanella. His unwavering support, guidance, and mentorship have been invaluable throughout this project. Professor Zanella's expertise and dedication to my work have been a constant source of inspiration. His insightful feedback and suggestions have played a crucial role in shaping this thesis into its final form.

I would also like to acknowledge my family and friends who have been a continuous source of encouragement and support. I want to thank them for their patience, understanding, and belief in my abilities. My colleagues have created a stimulating academic environment. Engaging in discussions and sharing ideas with them has been both enriching and enjoyable.

This thesis is the culmination of the collective efforts of many, and it is with humility and appreciation that I acknowledge your roles in this academic endeavor. Thank you for being a part of this journey.