IOT SERVICES

-

CHILD SAFETY MONITORING APPLICATION

Manuel Marceca

mama9469

Filip Silversten Wärn

fisi6585

January 12, 2025

Contents

1 Abstract

This report presents an Internet of Things project aimed at developing a child safety and monitoring system using sensors (ultrasonic sensor and PIR sensors), camera modules and smart devices such as remotely controlled plugs. The system integrates specific devices to monitor child activity in restricted areas and near hazardous objects, delivering real-time alerts and live video feeds via a mobile application. The primary goal is to enhance home safety by automating responses to potential risks and enabling remote monitoring for parents. The report details the system architecture, implementation process, testing results, and future recommendations.

The source code can be cloned from this repository:
https://github.com/MarcecaManu/ChildSafetyApp

Keywords: IoT, Child Safety, Raspberry Pi, MQTT, Mobile Application, Ultrasonic Sensors, PI Cameras, Android Studio

2 Introduction

Ensuring children's safety at home, particularly in environments with potential hazards such as kitchens or rooms containing electrical appliances, is a critical concern for parents. Advances in Internet of Things (IoT) technologies offer practical solutions by enabling interconnected systems that can monitor, alert, and respond proactively to risks.

This project introduces the Child Safety Monitoring System, an IoT-based application designed to enhance child safety by monitoring presence in hazardous areas and controlling dangerous appliances in real-time. The system integrates multiple devices, such as camera modules, smart plugs, and sensors, and uses the MQTT protocol to send real-time notifications to parents and deactivate appliances when necessary. The goal of this project is to demonstrate how IoT technology can provide a safer home environment and offer peace of mind to families.

This report covers the development process, key features of the system, and its potential impact on child safety, illustrating the role of IoT in addressing home safety challenges.

2.1 Background

Child safety remains a significant concern for families, especially in areas of the home that present inherent risks, such as kitchens or living spaces with electrical appliances. Despite various safety precautions, accidents continue to occur, often due to a lack of real-time awareness or control over potentially hazardous appliances. The growing prevalence of Internet of Things (IoT) technologies has created opportunities to address this issue, enabling the development of interconnected systems capable of monitoring, alerting, and responding to potential risks.

The Child Safety Monitoring System was developed to explore the practical application of IoT in improving child safety at home. This system leverages sensors, cameras, and smart devices to create an integrated solution for real-time monitoring and control. By focusing on the practical implementation, this project aims to show how IoT technologies can address ongoing safety concerns, providing a feasible solution for modern households.

2.2 Method

To ensure successful completion of the project within the designated timeframe, we established a detailed project plan that outlined key milestones and deliverables. The project development was organized into distinct phases, each focusing on specific tasks necessary for the realization of the Child Safety Monitoring System. These phases included project planning, requirement analysis, hardware procurement, development, testing, and final deployment. The overall time schedule included the following key stages:

- **Week 1 (Nov 29 - Dec 3)**: Development of the project plan and identification of project goals.
- **Week 2 (Dec 4 - Dec 10)**: Requirement analysis and system architecture design.
- **Week 3 (Dec 11 - Dec 16)**: Hardware procurement and initial sensor and camera integration.
- **Weeks 4-5 (Dec 16 - Dec 31)**: Prototype development and preliminary testing.
- **Week 6 (Jan 1 - Jan 7)**: Application development, final testing, and debugging; report drafting begins.
- **Week 7 (Jan 8 - Jan 15)**: Final adjustments, project report completion, and presentation preparation.

The initial steps of the project involved extensive research into IoT technologies, particularly those related to sensors, cameras, and communication protocols. This knowledge was gained through lectures, tutorials, previous coursework and additional online research. Early in the project, we created a visual prototype of the system to better define the system's components and functionality. This prototype helped us refine the system architecture and create detailed requirements for each functionality of the Child Safety Monitoring System.

The system development utilized various tools and platforms. We worked primarily with Visual Studio Code for Python scripts and Android Studio for application development. We also ensured the development followed modular principles, breaking the system into smaller components, each of which was developed and tested independently.

Remote collaboration played a significant role in our process, as the majority of our work was conducted remotely. We held frequent video calls for team meetings, and employed version control tools, specifically Git, to manage the code collaboratively. To facilitate smooth development, we also utilized pair programming, where both team members worked together on coding tasks. This approach was especially helpful when troubleshooting issues and building a shared understanding of the system's architecture.

Testing was conducted iteratively, starting with the integration of individual components. Each module, such as the sensors and smart plugs, was tested independently to ensure proper functionality. Once we confirmed that each component was working as expected, we began combining them based on the system's design and functionality. This phase focused on ensuring the different components worked together as intended, helping us identify and address any issues early in the process.

Due to not having access to the Pi camera module and one ultrasonic sensor for detecting the presence of an adult or child, we were unable to complete the full integration and testing of the entire system. Although we implemented the camera module and prepared the system for its integration, the absence of the actual camera prevented us from testing this feature fully. While we were able to verify that the fourth ultrasonic sensor would function correctly by temporarily using one of the other sensors for this purpose, we were not able to fully test the system with all components integrated together as intended. Despite these limitations, we conducted thorough testing of the individual components and the partial integration of the system, which enabled us to identify and address several key issues in the system's operation.

By following our planned timeline and maintaining regular communication, we were able to address challenges quickly and develop a working prototype of the Child Safety Monitoring System. This step-by-step approach allowed the system to improve over time, incorporating feedback from each testing phase.

3 Design and Development

In this chapter, we will discuss the design and development process of the Child Safety App, detailing the tools, technologies, classes, and functions used to achieve the final implementation as presented in the evaluation section.

3.1 System Architecture

The Child Safety App is designed as a multi-layered system integrating hardware (a Raspberry PI connected to sensors and actuators) and software (a mobile app and back-end). The architecture ensures real-time monitoring, data processing, and notification delivery to users.

3.2 Hardware Design

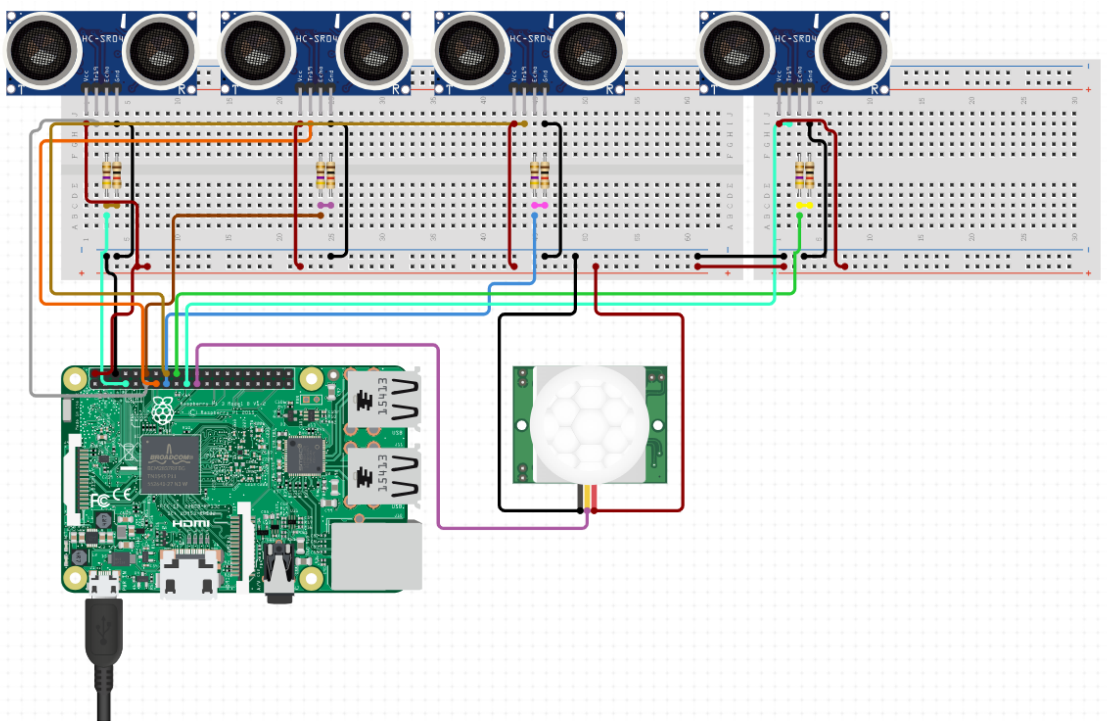


Figure 3.1: Circuit created using circuito.io [1]

3.2.1 Sensors

The sensing components include ultrasonic sensors for distance measurement and PIR sensors for motion detection. These sensors were configured to detect the presence of a child in restricted areas. The data from the sensors is processed by a Raspberry Pi 4 board, which acts as the central control unit for hardware operations.

Ultrasonic sensors

Component: HC-SR04
The system includes ultrasonic sensors to detect movements through the door and establish the number of people in the room; it can also distinguish between an adult and a child. Ultrasonic sensors are also used to detect the close proximity to a plug, so that it can be disabled whenever a dangerous situation occurs.

Motion sensor

Component: HC-SR501
The system includes PIR motion sensors to detect movements inside the room and serves as an additional check for presence detection. Default settings will consider a room empty if no motion is registered for more than 30 seconds.

3.2.2 Devices

Camera

Component: Raspberry PI Camera Module 2
The system includes a camera module from which the stream is processed and shown from the Android application, possible even when the phone is outside the local network.

Communication module

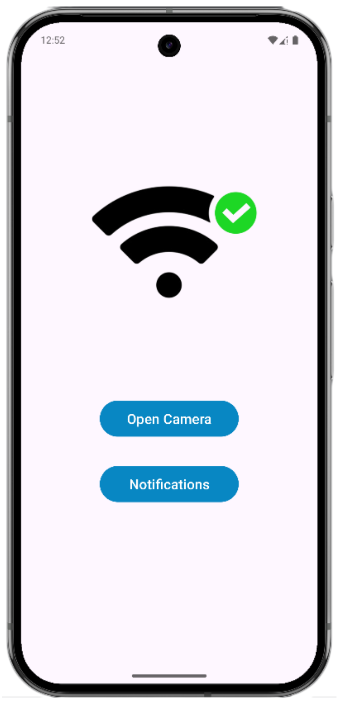
Component: Tellstick ZNet Lite v2
The system includes a communication module that makes communication from the Raspberry to the actuator possible through the Telldus API.

Actuators

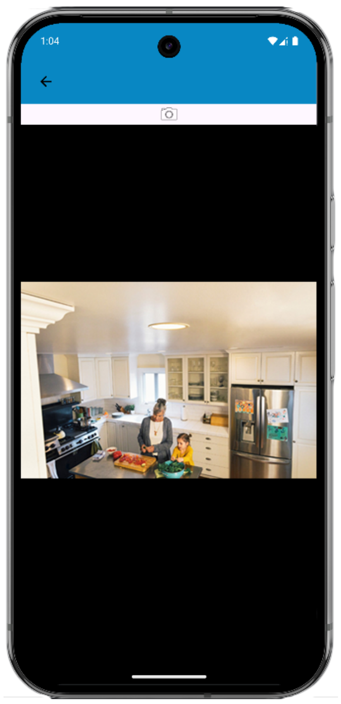
Component: Proove TSR101
The system includes a smart plug that can be turned ON and OFF via Telldus API.

3.3 Software Development

Mobile Application



(a) Homescreen



(b) Video Feed

Figure 3.2: Side-by-side images of the software.

The mobile application demonstrates the core functionality and the design of the user interface. The home screen, as shown in Figure 3.2a, features a minimalistic and user-friendly layout with two primary buttons: a ”Open Camera” button to access the live video feed and a ”Notifications” button to view alerts from the past 24 hours. This minimalist design ensures that users can easily navigate the app. Figure 3.2b shows the video feed screen, which integrates a real-time streaming view.

Mobile Application Development

We developed the mobile application using Android Studio, ensuring compatibility with Android devices. The app features three primary activities: the `MainActivity`, which serves as the home screen; the `CameraActivity`, which handles live video streaming; and the `NotificationsActivity`, which displays notifications from the past 24 hours. The `MainActivity` provides two primary buttons: a ”Open Camera” button that navigates to the `CameraActivity` for real-time monitoring and a ”Notifications” button that redirects to the `NotificationActivity`.

MainActivity

The MainActivity is the landing screen of the application, where the current status of the connection to the broker is shown. The following shows the screen when the connection is lost:

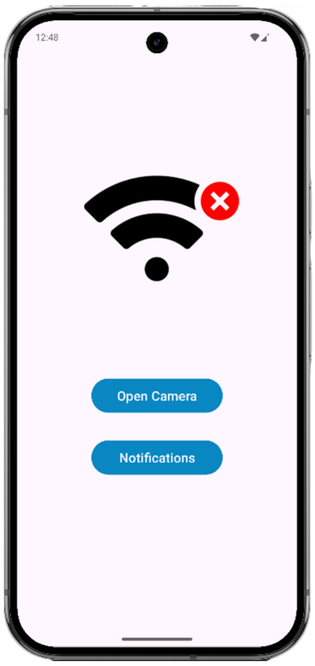


Figure 3.3: Connection refused screen.

NotificationsActivity

In this screen it's possible to see notifications from the last 24 hours, disclose them and eventually delete them. Notifications are stored using an SQLite database, a simple and lightweight solution to use storage.

CameraActivity

The `CameraActivity`, developed based on a project previously completed by Roy WCH. [3], uses a `WebView` to display a video feed through a dynamic DNS service called No-IP [2] (`http://childsafety4iot.ddns.net:8080`), supported by a `ProgressBar` for loading feedback. A forwarding rule must be set from the Raspberry Pi's static IP to port 8080 for the camera connection to work.

Code Snippet 1 - CameraActivity Class

```java
public class CameraActivity extends AppCompatActivity {

    private static final String CAMERA_URL =
        "http://childsafety4iot.ddns.net:8080";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_camera);

        // [...]

        webView.loadUrl(CAMERA_URL);
    }
}
```

The layout for `CameraActivity` was defined in the XML file `activity_camera.xml`. The XML file includes a `VideoView` element with an `id` of `videoView`. To display the live video, the `CameraActivity` class initializes the `VideoView` with the RTSP URL of the video stream. This URL, mentioned before, was hardcoded for testing purposes but can be dynamically updated in future iterations of the app.

3.4 Backend System

The backend, implemented in Python, runs on a Raspberry Pi 4 and utilizes various modules for its functionality:

- **Motion Detection:** Handles input from PIR sensors and ultrasonic sensors to determine activity.
- **Actuator Control:** Manages appliances through relays using the Telldus API.
- **Notification System:** Sends alerts using MQTT to notify users of important events.
- **Video Feed:** Streams live video via Raspberry Pi's camera module.

3.4.1 Core Classes and Functions

Main Manages the overall system states, processes sensor inputs, and coordinates actions like sending notifications or controlling appliances. Example function:

```python
def actuate(self):
        if self.child_in_room and not self.adult_in_room:
            if not self.notification_sent and time.time() -
                self.timestamp_child_alone > self.timeslot_child_alone:
                # Send notification
                mqtt_handler.send_notification("A child has been alone in the
                    room for over " + str(self.timeslot_child_alone) + "
                    seconds!")
```

Motion Sensor Handler Monitors motion using GPIO-connected PIR sensors. Example function:

```python
def monitor_motion():
    pir.wait_for_motion()
    motion_detected = True
    print("PIR - Motion detected!")
```

Ultrasonic Sensor Handler Tracks the presence of individuals near specific areas. Example function:

```python
def monitor_sensor_low():
    self.lower_sensor.wait_for_in_range()
    self.lower_detected = True
    print("Lower sensor - Motion detected!")
```

Actuator Handler Controls appliances through the Telldus API.

MQTT Handler Handles communication with an MQTT broker for real-time alerts. Example function:

```python
def send_notification(self, message):
        """Send a notification to the MQTT broker."""
        print(f"Publishing message to topic {self.pub_topic}: {message}")
        self.client.publish(self.pub_topic, message)
        print(f"Notification sent: {message}")
```

Camera Handler Streams live video from a Raspberry Pi camera to a web client. Example function:

```python
with picamera.PiCamera(resolution='640x480', framerate=24) as camera:
    output = StreamingOutput()
    camera.start_recording(output, format='mjpeg')
    try:
        address = ('', 8080)
        server = StreamingServer(address, StreamingHandler)
        server.serve_forever()
    finally:
        camera.stop_recording()
```

3.5 Testing and Debugging

The system was validated through:

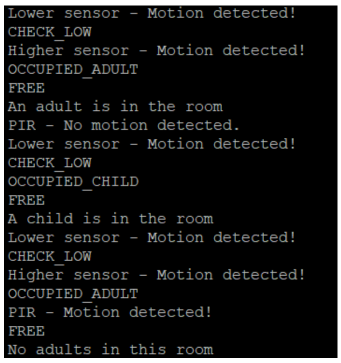
- **Unit Tests:** Ensured each sensor and control function works independently.
- **Integration Tests:** Verified interactions among sensors, controllers, and external systems.
- **Stress Tests:** Assessed system performance under prolonged usage.

4 Results and Outputs

In this chapter we will go through the application to present and evaluate the resulting output.

When the application is executed, it will show the main screen. Let's create a scenario:

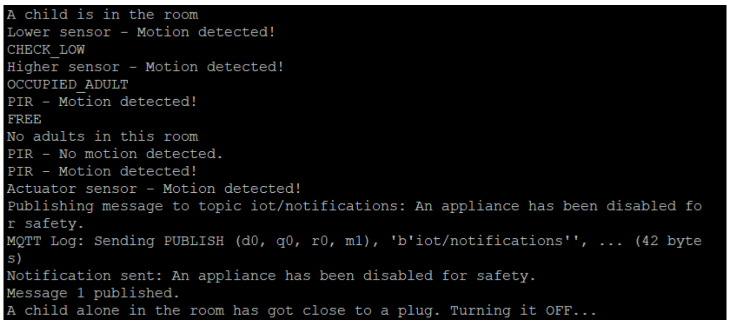
An adult is in the kitchen when a child also enters the room. After a few moments, the adult leaves the room, leaving the child alone. The system detects the following events:



```
Lower sensor - Motion detected!
CHECK_LOW
Higher sensor - Motion detected!
OCCUPIED_ADULT
FREE
An adult is in the room
PIR - No motion detected.
Lower sensor - Motion detected!
CHECK_LOW
OCCUPIED_CHILD
FREE
A child is in the room
Lower sensor - Motion detected!
CHECK_LOW
Higher sensor - Motion detected!
OCCUPIED_ADULT
PIR - Motion detected!
FREE
No adults in this room
```

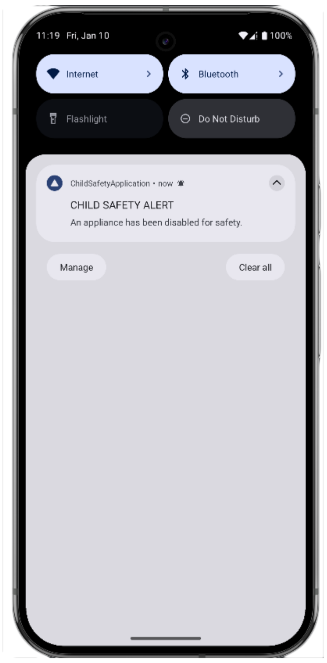
Figure 4.1: An adult and a child enter the room, until the adult leaves

After a while, the child gets too close to a plug. The system will detect this, disable the plug and send a notification:



```
A child is in the room
Lower sensor - Motion detected!
CHECK_LOW
Higher sensor - Motion detected!
OCCUPIED_ADULT
PIR - Motion detected!
FREE
No adults in this room
PIR - No motion detected.
PIR - Motion detected!
Actuator sensor - Motion detected!
Publishing message to topic iot/notifications: An appliance has been disabled fo
r safety.
MQTT Log: Sending PUBLISH (d0, q0, r0, m1), 'b'iot/notifications'', ... (42 byte
s)
Notification sent: An appliance has been disabled for safety.
Message 1 published.
A child alone in the room has got close to a plug. Turning it OFF...
```

(a) The system detects the hazard.



ChildSafetyApplication • now

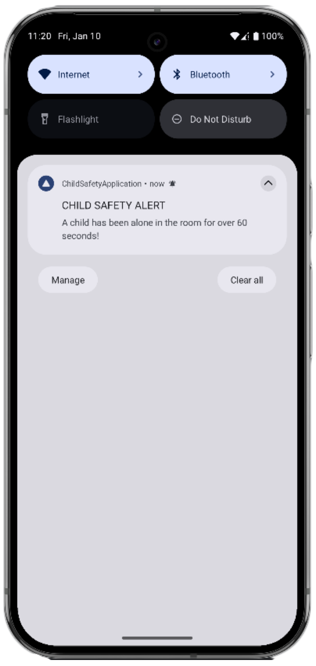
CHILD SAFETY ALERT

An appliance has been disabled for safety.

(b) A notification is sent.

Figure 4.2: The system detects a child approaching an electrical plug without supervision.

As the unsupervised child continues to roam the room, the system sends another notification after one minute:

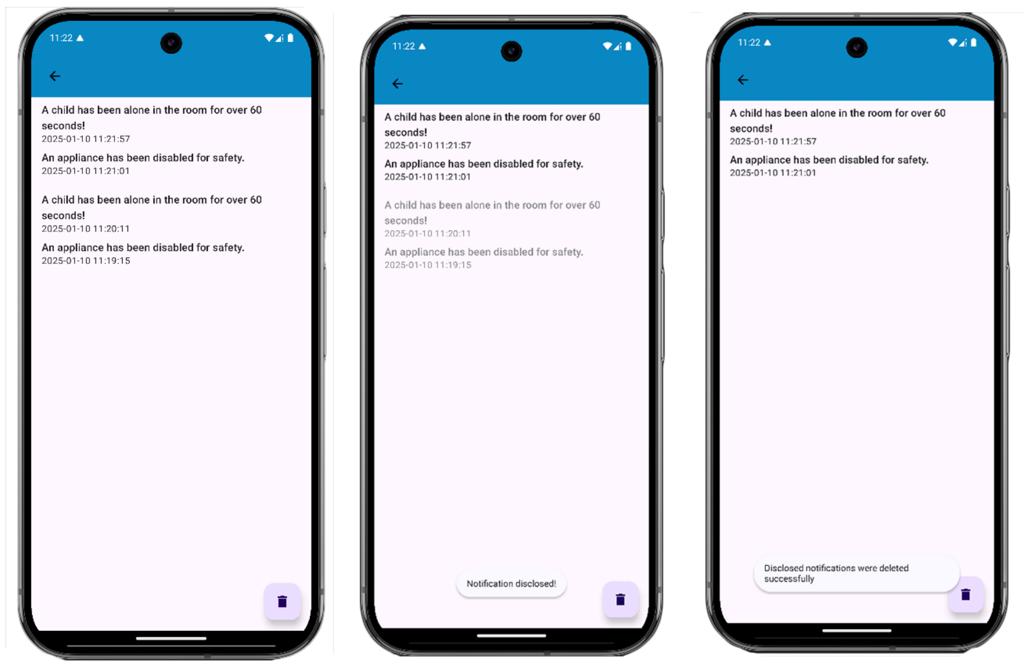


CHILD SAFETY ALERT

A child has been alone in the room for over 60 seconds!

Figure 4.3: A notification saying that the child has been left unsupervised for a minute is sent.

The user can interact with the notifications sent and eventually look at what is happening inside the room through the camera.


11:22

A child has been alone in the room for over 60 seconds!
2025-01-10 11:21:57

An appliance has been disabled for safety.
2025-01-10 11:21:01

A child has been alone in the room for over 60 seconds!
2025-01-10 11:20:11

An appliance has been disabled for safety.
2025-01-10 11:19:15

11:22

A child has been alone in the room for over 60 seconds!
2025-01-10 11:21:57

An appliance has been disabled for safety.
2025-01-10 11:21:01

A child has been alone in the room for over 60 seconds!
2025-01-10 11:20:11

An appliance has been disabled for safety.
2025-01-10 11:19:15

Notification disclosed!

11:22

A child has been alone in the room for over 60 seconds!
2025-01-10 11:21:57

An appliance has been disabled for safety.
2025-01-10 11:21:01

Disclosed notifications were deleted successfully

(a) User opens the notifications screen. (b) User discloses notifications. (c) User deletes disclosed notifications.

Figure 4.4: The user interacts with notifications.

When the adult finally comes back, the plug is re-enabled:

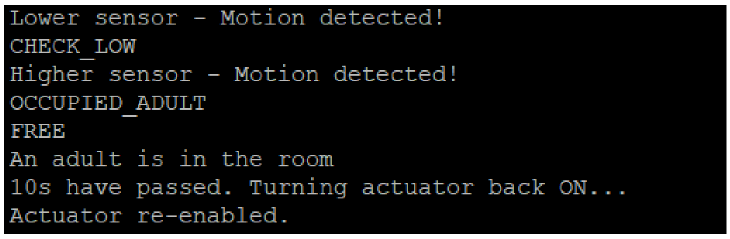
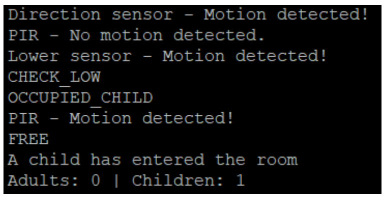

```
Lower sensor - Motion detected!
CHECK_LOW
Higher sensor - Motion detected!
OCCUPIED_ADULT
FREE
An adult is in the room
10s have passed. Turning actuator back ON...
Actuator re-enabled.
```

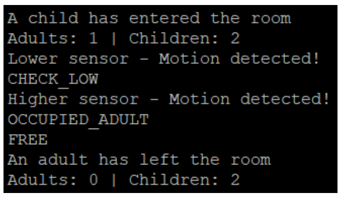
Figure 4.5: An adult enters the kitchen to supervise the child.

This short demonstration shows a prototype that supports a scenario in which only one adult and one child are considered. This was due to the limited amount of ultrasonic sensors available to us. We developed a solution that considers a total of four ultrasonic sensors, and uses the fourth new one to detect the direction in which the person is walking through the entrance door. This enables the system to understand the actual number of people in the room. Here is an example:



```
Direction sensor - Motion detected!
PIR - No motion detected.
Lower sensor - Motion detected!
CHECK_LOW
OCCUPIED_CHILD
PIR - Motion detected!
FREE
A child has entered the room
Adults: 0 | Children: 1
```

(a) A child enters the room.



```
A child has entered the room
Adults: 1 | Children: 2
Lower sensor - Motion detected!
CHECK_LOW
Higher sensor - Motion detected!
OCCUPIED_ADULT
FREE
An adult has left the room
Adults: 0 | Children: 2
```

(b) The adult leaves.

Figure 4.6: Example of the system in a scenario where multiple people are considered.

The system's behaviour is the same as before, but now works for a more realistic scenario.

5 Discussion

5.1 Achievements

The core functionality of the Child Safety Monitoring System is working as intended, providing a solution for monitoring a child's presence in potentially dangerous areas. The system is able to detect and distinguish if a child or an adult enters the area and subsequently tracks if the child is in the proximity of a dangerous appliance or a plug. When the child approaches the plug/appliance, the system is able to disable the plug to prevent the potential hazard/accident. Furthermore, notifications are sent to the user when a child is left alone in the dangerous area for a specific amount of time. This feature works as intended, alerting parents to the situation and providing an opportunity to intervene.

The functionality of the user interface in the application works effectively, allowing the user to interact with the notifications by opening the "notifications" screen on the homepage. There the user is able to view details and delete notifications once they have been addressed. Additionally, the camera module, when added, will allow users to monitor the dangerous area in real time, improving the general purpose of the system.

The system can detect the movement of people in the room, even with a limited number of ultrasonic sensors. However, the fourth ultrasonic sensor, designed to improve detection of individuals in the room and determine who is present, is integrated but not active because we did not have access to a fourth sensor.

5.2 Notes

While the system works well in simple scenarios with one child and one adult, missing components such as the camera module and fourth ultrasonic sensor, limit full functionality testing. This also prevents evaluation of scenarios with multiple people in the room. While individual testing confirmed the fourth sensor's ability to track room presence, the full integration testing remains incomplete.

5.3 Future works

While our current prototype of the Child Safety Monitoring System illustrates the capabilities of using IoT technologies to enhance child safety within homes, there are several improvements and expansions of the system that can be made to enhance its functionality and usability. One immediate step would be to acquire the missing hardware components, such as the Pi camera and ultrasonic sensor to allow full integration and testing of the system. Integrating the camera module would enable full monitoring of the area, while the additional ultrasonic sensor would improve the system's accuracy. To further improve the system, a login page could be implemented to enhance security, making sure that only authorized users can access the application and its features. Additionally, the system could be scaled by adding more sensors and cameras in multiple rooms, making it more adaptable for larger households. More smart plugs could also be added, allowing the system to control multiple potential dangerous appliances. In future versions of the system, the architecture could be expanded to support multiple families, each with its own child safety monitoring system. By implementing a cloud-based solution, the system would allow each family to register and manage their own child safety monitoring system securely. Each user would log in to their personal account, which would link them to their unique child safety system. This approach would allow the system to scale for a larger user base while maintaining the privacy of each household's information.

Bibliography

[1] circuito.io. *Circuit Design App for Makers*. URL: `https://www.circuito.io/`.

[2] No-IP. *Create an easy to remember hostname and never lose your connection again.* URL: `https://www.noip.com/`.

[3] Roy WCH. *WiFi live streaming Raspberry Pi camera (Android)*. URL: `https://roywchpi.blogspot.com/2020/04/8-live-streaming-pi-camera-android.html`.