Development of Argus Box: An IoT Smart Safe Monitoring System

Pranshu Bhardwaj
School of Science, Technology, Engineering & Mathematics
University of Washington Bothell
Bothell, WA, USA
Email: pran@uw.edu

***Abstract*—Traditional safes primarily offer physical protection, but in the age of sensor networks and IoT devices, they lack advanced monitoring and security capabilities. This paper presents the development of *Argus Box*, a smart monitoring system for safes that continuously tracks the safe's state and records short videos upon access. The system utilizes commonly available hardware components to implement a cloud-connected camera monitoring system, emphasizing modularity and feature enhancement. The findings demonstrate the feasibility of creating a scalable and cost-effective security solution for safes and lockers of various sizes, with customizable features based on user requirements and budget. Additionally, the development process highlights challenges associated with integrating new hardware, such as the Raspberry Pi 5, due to limited library compatibility and longer implementation times.**

***Index Terms*— AWS, Argus Box, cloud, IoT, monitoring system, Raspberry Pi, safe, security, smart safe**

I. Introduction

THE traditional safe is a very well-known item that is found in many homes. It can be used to store various valuable items that you wouldn’t want anyone to have direct access to like passports, jewelry, guns, essential documents, etc. These safes come in various shapes and sizes for different purposes. The types of safes can include, gun and jewelry safes, portable safes, safes for electronics, fireproof or waterproof safes, and so on [1]. It is very easy to find a safe suitable for one’s use because of the wide variety of options available. All of these safes come with some sort of security measure to keep them secure. The security measure that all safes have is a lock system. Modern safes are usually electronic and conveniently offer access through number pads or touch screens [1]. These electronic locks usually mean better protection than physical key fobs because they help mitigate the risk of lockpicking. Some safes also come equipped with a security system with trigger alarms or delayed access after incorrect attempts and additional measures like 2 Factor Authentication [1]. These systems help make the safe more reliable.

However, despite offering proper safety measures, there seems to be a lack monitoring for the actual access and the contents of the safe itself. If the access to the safe is to be compromised, whether it be due to the password being leaked or the safe’s lock system being hacked, there maybe be no way for the user to know about such an event. The lack of monitoring may not only leave the user in the dark about who accessed the safe and what was taken but also unaware if the safe was accessed at all.

IoT devices have become incredibly popular due to their affordability and access to the internet. In the summer of 2024, it was reported that there have been approximately 16.6 billion IoT devices connected to the internet by the end of 2023 and this number is expected to rise to 18.8 billion by the end of 2024 [2]. This huge increase in usage shows the usefulness these devices offer and also indicated how rapidly everyday devices are being turned into IoT devices. Connecting the devices to the internet allows users to view the information from their devices in real time as long as they are connected to the internet and also enables to the devices to be “smarter” by using edge and cloud computing.

The goal of this paper is to go over the implementation of a smart safe monitoring system called Argus Box. The purpose of the Argus Box system is to provide a reliable monitoring system that offers modularity, cloud connectivity, and scalability.

The Argus Box monitoring system enhances the security capabilities of a safe or a locker by allowing real-time updates to be sent to the user. Argus Box also offers the benefit of keeping track of items. It’s easy to forget when we take items out of their places and find them missing when we return. A monitoring system that records the inside of the safe when it is being accessed allows for real-time updates for the user to keep track of the items they place and remove from the safe. Such systems can also be enhanced by additional sensors other than cameras like a load cell to measure the weight to monitor the state of the items even more accurately.

The development of the Argus Box system involves hardware assembly, software development, and cloud integration. The hardware process involved learning how to work with Raspberry Pi, designing circuitry, and wiring the components effectively. Software development involved research for the components, creating classes for modules, and developing the main script to make everything work together.

II. RELATED WORK

The development of security systems is extremely important and also very common. The Raspberry Pi is also an extremely accessible, affordable, and capable credit card-sized microcomputer that can be used to operate a wide variety of projects from doorbells to robots [3]. Because of this, research is abundant for Raspberry Pi-based security systems. A lot of the libraries and components used in the development of these systems can be applied to the Argus Box system. However, most of these security systems are targeted towards home security/monitoring and non for box/safe monitoring.

A research article by Al-Rawi, Muhanned, and Abdulhamid proposes a Raspberry Pi-based surveillance system for intruder detection which aims to provide real-time monitoring, remote control, and alert operations [4]. This study involves the development of a remote surveillance system that offers various elements like the alert system and object recognition which can be used in a smart safe system and the detection of strangers may not be very useful for a smart safe. The video streaming can however be added to the Argus Box system as an additional feature.

Another article by Teja, Joe, and Kalist proposes a different Raspberry Pi IoT surveillance system that makes use of infrared sensors and cameras [5]. This IoT system records 7 videos and sends them to its IoT server where it is then stored in a USB drive. The concept of recording a small video of activity and sending it to the cloud server alongside other sensor data is meant to be used in the Argus Box system. This system is similar to what Argus Box does but it is also meant for home security and unlike Argus Box, uses infrared sensors.

Lastly, the study by Nadafa, Hatturea, Bonala, and Naikb researches a smart mirror-based approach to Raspberry Pi surveillance systems [6]. The system is packed into a smart mirror which performs facial recognition and remote monitoring. This system is packed into a smaller device which is how the Argus Box system is also meant to be implemented. Additionally, the system uses facial recognition which can also be optionally used in the Argus Box system for additional features.

Like these works, there is plenty of research done on Raspberry Pi-based IoT security systems. But most of these works are solely based on home security and mostly activity-based, unlike Argus Box which also detects state-based data alongside activity-based data to record the state of the safe when it is opened.

The Argus Box system contributes to the field of Raspberry Pi-based security systems by offering a modular approach to monitoring targeted toward safes and lockers.

III. SYSTEM MODEL, PROBLEM STATEMENT, ANALYSIS

The goal when designing the model for the Argus Box system was to develop a system with components that can fit easily inside a shoebox and communicate directly with AWS IoT through a Wi-Fi connection. The system also needed to stay functioning if power was interrupted.

A. System Model

The system model for the Argus Box Smart Safe Monitoring System is built around the Raspberry Pi 5. The Raspberry Pi powers and directly controls all the hardware components of the safe. The Pi is also directly connected it two cameras. It is also responsible for sending data to AWS based on the signals it receives through the connected modules. The Raspberry Pi is powered by an Uninterrupted power supply (UPS) unit. Figure 1 represents a high-level diagram of this model.

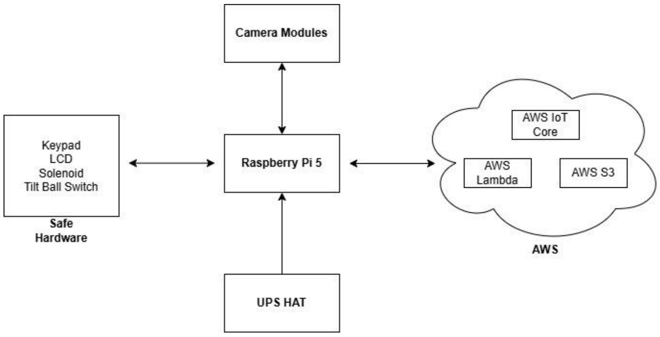

Fig. 1. System Model for the Argus Box Smart Safe Monitoring System

1) *Raspberry Pi 5*

The Raspberry Pi 5 is the latest version of the Raspberry Pi microcomputer which allows for fast processing and up to 8GB of memory [7]. The Pi used in the Argus Box system had 8GB memory which allows for fast video processing and uploading to the cloud. Another reason the Raspberry Pi 5 was a must for this project was because it offers two camera slots on its board compared to only one in all the previous generations. It also has 40 GPIO pins which allow for a large number of electronic components to be connected and enable modularity to install more components if needed. Figure 2 shows the Pi used for this project.

Fig. 2. Raspberry Pi 5 with cooling unit attached on top

2) *Cameras*

There are 2 camera modules directly connected to the Raspberry Pi. The Pi Camera Module 3 was chosen to be used as it allows for full HD video at 50 fps and has autofocus [8]. This camera module is ideal for taking 10-second HD videos both from inside and outside the safe. The autofocus feature also helps the camera to not get blurry while recording. Figure 3 shows the Pi Camera Module 3.

Fig. 3. The picture of a Raspberry Pi Camera Module 3

3) *Components for Safe*

The components of the safe itself are essential for the safe to perform its basic function and actually be considered a safe that locks and unlocks. The components for that Argus box system include a keypad module, an LCD module, a solenoid lock, and a tilt ball switch. The keypad and LCD module act as the user interface for the password system. The solenoid lock is responsible for locking and unlocking the safe. Lastly, the tilt ball switch is responsible for recognizing if the safe is opened or closed.

4) *UPS unit*

For the UPS Unit, the Argus Box system uses the Waveshare UPS HAT B which is an expansion board that goes under the Raspberry Pi and supplies uninterrupted 5V power [9]. The UPS HAT uses 2 18650 batteries that power the Raspberry Pi when the power supply is disconnected. The UPS HAT is rated to be able to power the Pi for 2-3 hours [9]. However, this may vary based on the processing and components connected to the Pi.

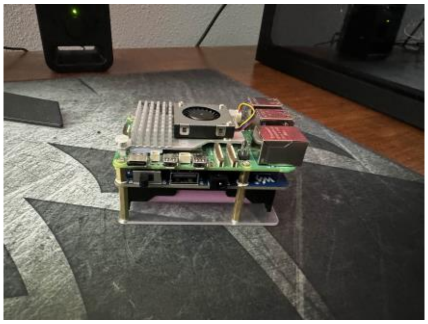

Fig. 4. Waveshare UPS HAT B mounted under the Raspberry Pi 5

5) *AWS cloud architecture*

AWS is the platform being used for the cloud operations of the Argus Box system. The three AWS services used for the system are IoT Core, S3, and Lambda. The IoT Core is used to receive MQTT messages containing the state of the safe and cameras from the safe. An AWS Lambda function is used to compute the time since the safe was last opened based on the MQTT messages. Finally, an S3 bucket is used to store raw messages, processed messages, and videos captured by the Argus Box system.

B. *Problem Statement*

Traditional safes, while effective at providing physical security, cannot monitor and report access or track their contents in real-time. This limitation exposes users to potential risks such as unauthorized access, theft, or tampering, without any way to detect or respond to these events promptly. Current solutions that incorporate advanced security features, such as electronic locks and alarm systems, do not provide comprehensive monitoring capabilities, leaving users unaware of who accessed the safe, when it was accessed, or what was removed [1].

Additionally, the lack of modular and scalable monitoring systems for safes restricts the ability to customize features based on user needs or budgets. Despite the increasing adoption of IoT technologies in home security systems [2], there is no widely available solution that integrates real-time monitoring, cloud connectivity, and modularity for safes.

C. *Analysis*

This research paper aims to address these gaps by offering an IoT-based solution that continuously tracks the safe’s state, records access events through 10-second videos and sends real-time information that is sent to the AWS cloud which the users can access by using dedicated applications. By making use of commonly available hardware, the system provides an affordable and scalable solution for enhancing the security and usability of safes and lockers.

IV. Design and Implementation

The design and implementation of the hardware system were a top priority for this project. It was essential that all hardware components worked together in harmony to have a robust safe monitoring system with a complete password and locking system included. The software development was handled side by side with the addition of each component to the system rather than later. Cloud integration was the final step for the safe after both the hardware and software for the safe were working as a complete system.

A. *Hardware System*

The hardware system was developed with modules added one on top of the other rather than separately to ensure all system components worked together as it was developed.

1) *Camera System*

The camera system was the first and simplest step of the hardware implementation. Because the camera modules are from Raspberry Pi themselves, they have working libraries already installed on the system to get the cameras working very easily. The camera modules were connected to the two camera ports on the Pi 5 and they were good to go.

2) Password System

The password system was the first to be implemented. The hardware components for this system involved the keypad module and the LCD module both connected to the Pi. The goal when implementing this system was to ensure that the LCD registered the letters that were typed on the keypad and also that the typed password was able to be both entered and deleted using the ‘*’ and ‘#’ keys. Figure 5 below demonstrates the two components connected to the Pi with a breadboard.

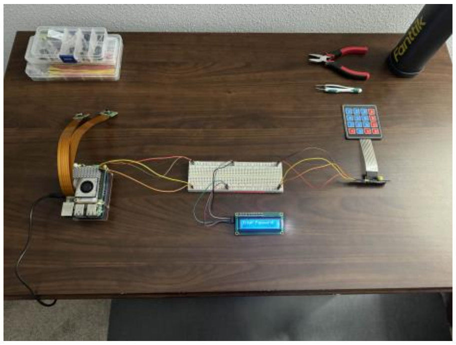

Fig. 5. Keypad module and LCD module connected to Raspberry Pi with breadboard.

The membrane keypad module shown in Figure 5 was later replaced with a mechanical one. The mechanical keypad had 13 pins for communication. To save the GPIO pins from being overused, the keypad module was connected to an I2C module (shown in Figure 6) that reduced the number of pins used by the keypad from 13 to 2 excluding the 2 power pins used by the I2C module.

Fig. 6. Waveshare I2C MCP23017 IO Expansion Board

3) Locking System

The next step of the system involved connecting a locking system to the Argus Box system to work with the password system. A small 5V solenoid was used with this system so it could be directly powered by the Pi. A bigger and more powerful solenoid lock can be installed on a modified with an external power source for the solenoid. To control the solenoid, a MOSFET (Figure 7) was added to it could take a signal from the Raspberry Pi GPIO and enable power to the solenoid to unlock it.

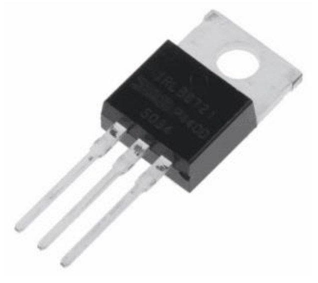

Fig. 7. MOSFET used to control power to the solenoid

However, there was an issue with the compatibility of the MOSFET and Raspberry Pi GPIO. To give a signal to the MOSFET to open the drain and let current through, it needed a signal of around 5V while the Raspberry Pi GPIO is only capable of supplying a maximum voltage of 3.3. To solve this issue, a level shifter (Figure 8) was added to the circuit to convert the 3.3V signal from the Pi to a 5V signal.

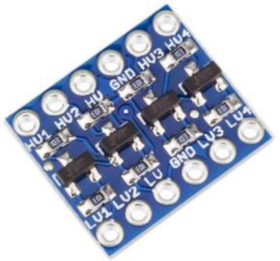

Fig. 8. Bidirectional 3.3v to 5v Level Shifter used in the circuit

After adding the level shifter, the MOSFET was able to receive a high enough signal to open the drain and allow current to flow through the solenoid to unlock it. Figure 9 shows the complete circuit for the solenoid.

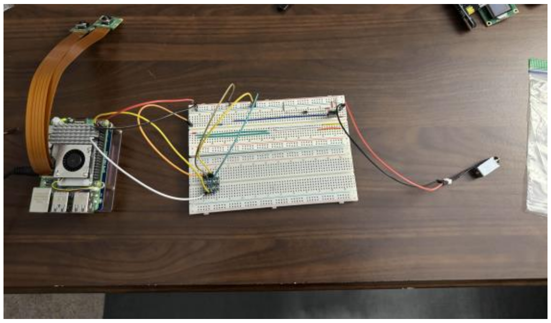

Fig. 9. The circuit of the Raspberry Pi controlling the 5v solenoid with a level shifter

4) Breadboard Circuit

The two circuits for the password system and the lock system were combined to make a single big circuit to allow the Pi to easily access all the components easily though the breadboard and make the assembly easier. Figure 10 shows the password system circuit and lock system circuits combined into a single breadboard circuit.

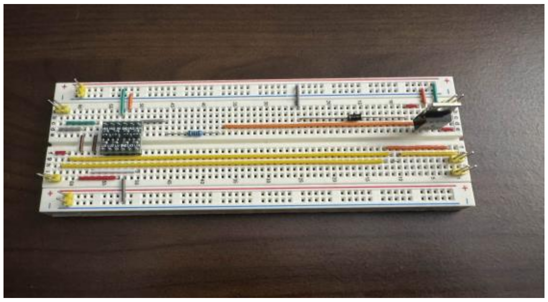

Fig. 10 Breadboard circuit to connect the Raspberry Pi to all the modules controlling the safe

5) *Tile Ball Switch*

The last component that was added to the system was the tilt-ball switch (Figure 11). This switch is to indicate whether the box is closed or not. The original plan for this feature was a reed switch but due to time constraints, a tilt switch was used. This switch is more situational but works to demonstrate a working prototype.

Fig. 11 A tilt ball switch used to indicate if the box is closed or not based on its orientation

6) *Box Assembly*

After the Argus Box system hardware was working as a complete system. I integrated the system into a cardboard box. This was both due to limitations of not being able to create a custom metal safe and to show the argus box system could be applied to any box as long as it opened, closed, and locked. Figures 12 and 13 show the inside and outside of the shoe box in which the argus box system was integrated.

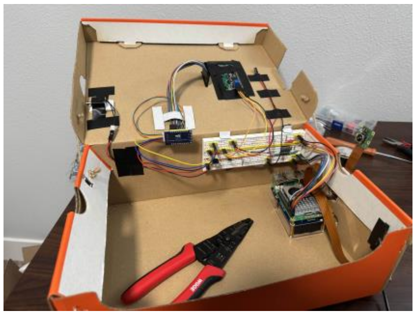

Fig. 12 Argus Box System integrated inside a regular shoebox

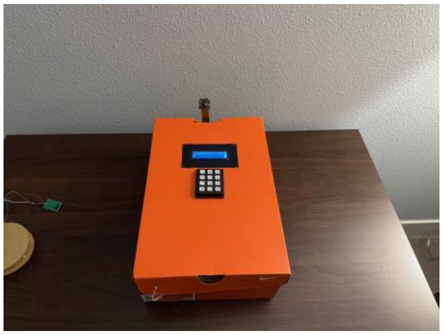

Fig. 13. The user interface for the Argus Box System integrated outside of a shoebox

B. *Software System*

The software system for the Argus Box was developed as the hardware was assembled.

1) *SmartSafe class*

The main software component that handled everything in the software was the smart safe class. This class was responsible for managing the controls of all the components based on signals received from them. The main components of this class were the password system and the camera monitoring system. The camera system made use of the system-integrated picamera2 class which allowed easy access to the cameras. Every time the safe was accessed the outside camera would record a 10-second video and if it continued being accessed the camera kept recording until the user stopped pressing buttons. When the safe is opened, both the inside and outside cameras constantly record 10-second videos until the safe is closed.

2) *Classes for modules*

All the attached components had classes that allowed them to send and receive signals from the Pi. The LCD module was the only module that had a complicated interface, but a tutorial website provided the code to write and erase text on the LCD [10]. All of these classes were imported into the SmartSafe class for everything in the system to work together.

C. *Cloud Integration*

For the AWS APIs to install and run on the Raspberry Pi, a virtual environment had to be created since direct system modification is restricted on the Raspberry Pi. Figure 14 shows how the data flows in this system.

1) *AWS IoT Core*

The Pi being used in the Argus Box had to be registered as a thing in the IoT Core to allow for MQTT communication. After the Thing was created, the AWS connection kit was installed to establish a connection with AWS IoT Core, and pubsub.py in the installed SDK from the connection kit allowed the Pi to send MQTT messages to AWS successfully. Pubsub.py was then modified to run the SmartSafe class and send safe access data including the time accessed. The current time, the camera status (recording or standby), and the status of the box (open or closed).

2) AWS S3

A dedicated AWS S3 bucket was created to store data from the Argus Box system. The SmartSafe class was modified to send the videos it recorded directly to its dedicated S3 bucket. To enable this, an IAM user had to be created for the Pi to give permissions to access the S3 buckets and generate an access key (and secret key) for the Pi to log into the AWS account. To perform such tasks, AWS CLI and boto3 had to be installed in the virtual environment to allow the Pi to access and upload videos to the S3 bucket.

3) AWS Lambda

Lastly, a Lambda function was created in AWS to take the raw data from the MQTT messages sent by Argus Box and process it by subtracting the last access time from the current time creating a duration, and then storing both the processed data and raw data in the dedicated S3 bucket. This function can later be modified to perform more complex tasks if need be.

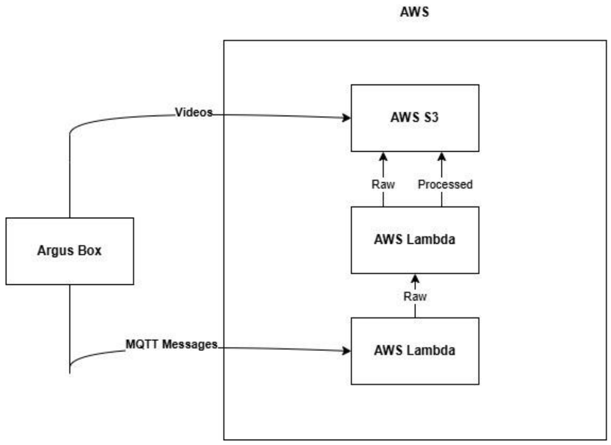

Fig. 1. System Model for the Argus Box Smart Safe Monitoring System

V. EVALUATION

The testing of the Argus Box system demonstrated that the safe performs its intended functions effectively. The integration of hardware and software components worked cohesively to deliver the desired features, although a few limitations were observed during testing.

A. System Performance

The cameras reliably recorded 10-second videos during access events. The outside camera began recording the moment the keypad was accessed, while both cameras operated when the safe was opened. The videos were successfully sent to AWS S3 almost instantly after recording, ensuring near real-time monitoring. However, the system encountered a potential issue when internet connectivity was slow. If the video upload to AWS was delayed, subsequent video recordings could not start until the previous upload was completed. This delay could compromise the system's ability to consistently monitor extended activity.

The MQTT messages functioned as expected, sending data regarding the state of the safe every second. This ensures that users receive timely updates on whether the safe is locked or unlocked, as well as the status of the cameras. The messaging system proved to be robust and reliable during all tests.

B. Password and Locking System

The password and locking mechanisms also performed their tasks as designed. The keypad successfully communicated with the Raspberry Pi to control the locking system. However, a minor delay was noted between pressing a button on the keypad and the corresponding character appearing on the LCD. While this delay does not affect functionality, it may slightly impact the user experience, particularly for users accustomed to instantaneous feedback from digital devices.

The solenoid lock functioned as intended and reliably locked and unlocked the safe based on input from the keypad. However, due to the prototype being constructed from a cardboard box secured with tape, the safe could still be forced open despite the lock being engaged. This limitation is inherent to the prototype materials and would not exist in a production model with a better more solid body made out of metal.

Overall, the Argus Box system successfully integrates IoT technologies to enhance traditional safe monitoring. The cameras, password system, lock system, and cloud integration worked together to deliver a functional and efficient solution. While the prototype's hardware design introduces some limitations, such as the box's fragility and video upload bottlenecks during slow internet conditions, these issues are specific to the testing environment and can be addressed in future iterations. Despite these challenges, the core system demonstrates strong potential as a scalable and secure IoT-enabled safe.

The evaluation highlights that while the Argus Box achieves its primary goals, there is room for refinement to improve user experience, ensure consistent monitoring, and enhance physical security.

VI. CONCLUSION AND FUTURE WORK

The Argus Box system successfully demonstrates the potential of integrating IoT technologies into traditional safes to enhance security and usability. The system functioned as intended, with reliable real-time monitoring through cameras, secure locking mechanisms, and cloud integration. Despite some minor delays, such as in the keypad-to-LCD feedback and video uploads during slow internet conditions, the prototype proved to be effective in meeting its design goals. However, the prototype's use of a cardboard box limited its ability to showcase the full potential of the Argus Box system.

To realize its complete capabilities, the Argus Box needs to be integrated into an actual safe. Testing with a proper enclosure would address the physical vulnerabilities of the current prototype and provide a more accurate assessment of its reliability and security under real-world conditions.

Initially, the Argus Box system was designed to include a load cell for monitoring the weight of items within the safe. Unfortunately, due to the Raspberry Pi 5's adoption of updated GPIO libraries, many existing libraries for load cells are currently outdated or incompatible. However, with enough time, this feature could still be implemented, as new libraries or workarounds could become available.

The modular design of the Argus Box allows for extensive future enhancements. A large number of unused GPIO pins and the abundance of software libraries available for the Raspberry Pi 5 opens the door for additional features, such as facial recognition, advanced sensors, or even biometric authentication. These upgrades could further increase the system's security and functionality.

In conclusion, the Argus Box represents a promising step toward the modernization of traditional safes. By leveraging the capabilities of IoT and the modularity of its design, the system offers a scalable platform for creating smarter and more secure storage solutions. Future work will focus on integrating the system into a robust physical enclosure and exploring advanced features to enhance its capabilities further.

References

[1] claire koeppel, "Security Safes: Protecting Valuables And Important Documents," StaySafe.org, Sep. 05, 2023. https://staysafe.org/home-safety/security-safes-protecting-valuables/ (accessed Dec. 04, 2024).

[2] S. Sinha, "State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally," IoT Analytics, Sep. 03, 2024. https://iot-analytics.com/number-connected-iot-devices/ (accessed Dec. 04, 2024).

[3] R. A S, "What Is Raspberry Pi? Here's The Best Guide To Get Started | Simplilearn," Simplilearn.com, Jul. 23, 2014. https://www.simplilearn.com/tutorials/programming-tutorial/what-is-raspberry-pi (accessed Dec. 04, 2024).

[4] Al-Rawi, Muhanned & Abdulhamid, Mohanad & Sheshai, Singoee. (2019). Design of Security System Based on Raspberry-PI. The Scientific Bulletin of Electrical Engineering Faculty. 19. 56-61. 10.1515/sbeef-2019-0022.

[5] P. A. Teja, A. A. F. Joe and V. Kalist, "Home Security System using Raspberry PI with IOT," 2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), Greater Noida, India, 2021, pp. 450-453, doi: 10.1109/ICACITE51222.2021.9404551.

[6] Raju A Nadafa, S.M. Hatturea, Vasudha M Bonala, Susen P Naikb, Home Security against Human Intrusion using Raspberry Pi, Procedia Computer Science, Volume 167, 2020, Pages 1811-1820, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.03.200. (https://www.sciencedirect.com/science/article/pii/S1877050920306657)

[7] Raspberry Pi Ltd, "Buy a Raspberry Pi 5," Raspberry Pi. https://www.raspberrypi.com/products/raspberry-pi-5/ (accessed Dec. 04, 2024).

[8] R. P. Ltd, "Raspberry Pi Camera Module 3," Raspberry Pi. https://www.raspberrypi.com/products/camera-module-3/ (accessed Dec. 04, 2024).

[9] "UPS HAT (B) - Waveshare Wiki," Waveshare.com, 2024. https://www.waveshare.com/wiki/UPS_HAT_(B) (accessed Dec. 05, 2024).

[10] "Tutorial 1 - LCD1602," NerdCave, Jan. 25, 2022. https://nerdcave.xyz/raspberrypi/module-and-sensors/tutorial-lcd-1602/ (accessed Dec. 04, 2024).