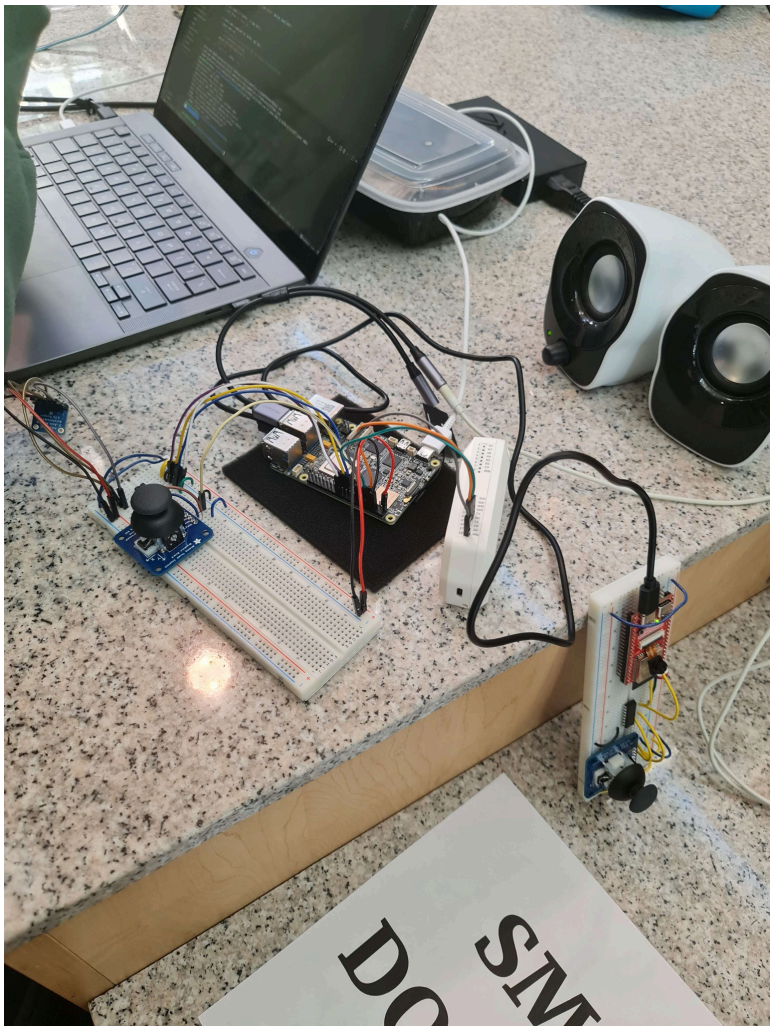
Smart Doorbell & Security System: Project Write-up

1. System Explanation

System Overview

The **Smart Doorbell & Security System** is a distributed embedded application designed to simulate a modern smart home entry solution. It integrates multi-factor access control, real-time video surveillance, and cloud-based alerting into a unified platform. The system is orchestrated by a **BeagleY-AI** single-board computer, which serves as the central control unit. It interfaces with two external subsystems: an **ESP32-CAM** acting as a wireless IP camera and a **Flipper Zero** functioning as an RFID reader. The system operates in real-time, concurrently processing video frames for motion detection, polling sensors for tampering, and listening for user authentication inputs via UART and SPI.

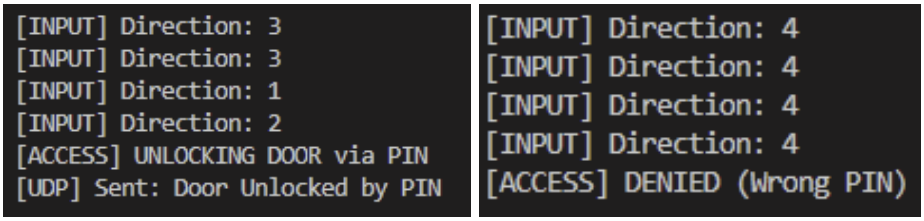


Feature Implementation & Technical Workflow

1. User Interface & PIN Authentication

The primary user interface consists of a **2-axis analog joystick** with an integrated push-button. The BeagleY-AI reads the raw analog values via an SPI-connected ADC (Analog-to-Digital Converter).

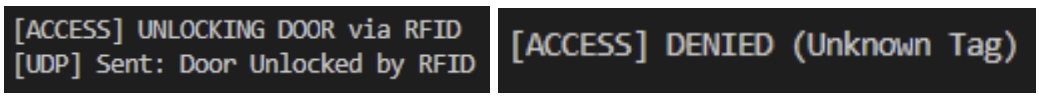
- **Doorbell Function:** Pressing the joystick button triggers the "Ding Dong" audio playback to announce a visitor.
- **Secure PIN Entry:** We mapped the joystick axes to directional values: **+Y (Up) as 1**, **-Y (Down) as 2**, **-X (Left) as 3**, and **+X (Right) as 4**. To unlock the door, the user enters a specific sequence (e.g., Left, Left, Up, Down).
- **Feedback:** The system provides immediate audiovisual feedback.
    - **Success:** If the PIN is correct, the on-board **Green LED** illuminates, and a "Success" sound plays, simulating the door unlocking mechanism.
    - **Failure:** If the sequence is incorrect, the **Red LED** flashes rapidly, and an "Access Denied" sound is played.



2. Contactless RFID Authentication

For secure, contactless entry, the system integrates a **Flipper Zero**. We developed a custom Flipper application (.fap) that utilizes the device's internal 125kHz RFID hardware.

- **Communication Pipeline:** When a user scans an RFID fob, the Flipper Zero extracts the unique Tag UID. It formats this UID into a hexadecimal string and transmits it to the BeagleY-AI via a **UART** (Universal Asynchronous Receiver-Transmitter) connection established on the GPIO headers.
- **Verification:** The BeagleY-AI listens on the serial port (/dev/ttyAMA0) in non-blocking mode. Upon receiving a UID string, it compares the data against a whitelist of authorized keys. A match triggers the same "Unlock" sequence as a correct PIN, while an unauthorized tag triggers an "Access Denied" alert.



3. Surveillance & Motion Detection

The video subsystem relies on an **ESP32-CAM** module running a custom C++ firmware. The ESP32 hosts a lightweight HTTP server that provides a live MJPEG stream.

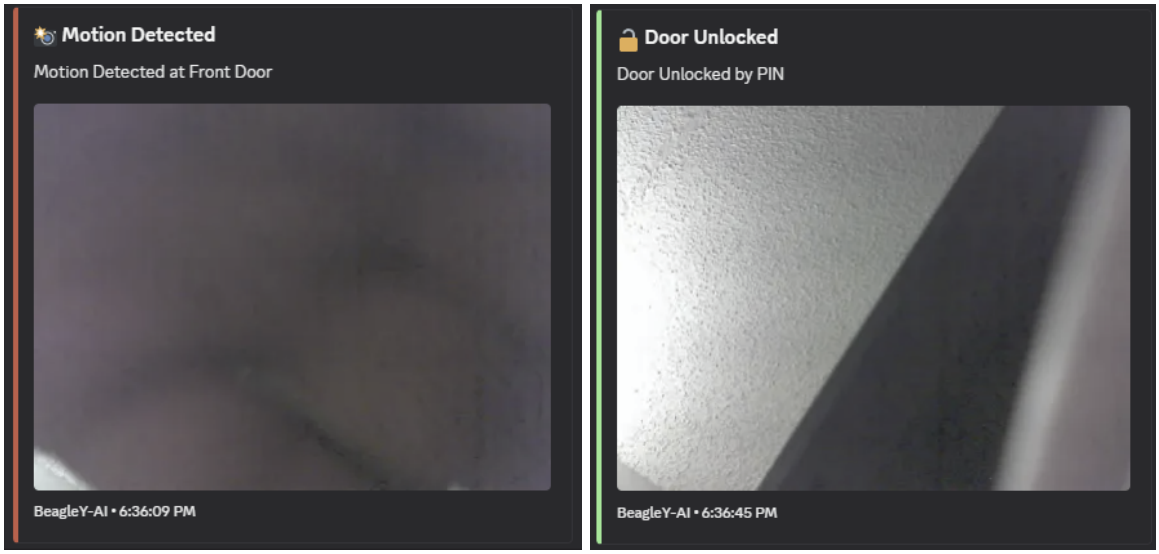
- **Frame Acquisition:** The BeagleY-AI captures high-resolution still images from the stream using wget system calls with strict timeouts to ensure system responsiveness.
- **Motion Algorithm:** We implemented a computer vision algorithm in C. The system decodes the JPEG frames into RGB buffers and uses a **Running Average Background Subtraction** method. By blending 20% of each new frame into a reference background buffer, the system adapts to gradual lighting changes while successfully identifying sudden changes (visitors) as motion events.

```
[MOTION] Movement detected!
[UDP] Sent: Motion Detected at Front Door
```

4. Cloud Integration & Remote Alerts

To enable real-time mobile notifications, we developed a **Node.js middleware server** that runs locally on the BeagleY-AI.

- **Trigger:** When the C application detects motion, a tamper event, or a doorbell press, it saves the current video frame to a RAM disk (/tmp) and sends a lightweight **UDP packet** to the local Node.js server.
- **Dispatch:** The Node.js server listens for these UDP triggers. Upon receipt, it fetches the image from RAM, constructs a multipart/form-data payload, and posts it to a **Discord Webhook**. This delivers a rich notification to the user's smartphone, complete with a timestamp, event title (e.g., "Motion Detected"), and a snapshot of the event.



5. Tamper Detection

To prevent physical tampering, the system continuously polls an **Accelerometer**. The application calculates the absolute difference (delta) of the X, Y, and Z acceleration vectors between loop iterations. If the device is shaken or forcibly removed (delta > TAMPER_THRESHOLD), the system enters an alarm state, playing a loud siren sound and flashing the Red LED to deter the intruder and alert nearby individuals.

Challenges & Resolutions

During development, we faced two significant technical hurdles:

1. **Motion Detection "Ghosting":** Our initial algorithm compared each frame only to the previous one, causing excessive false alarms from minor lighting flickers. We switched to a "Running Average" background model to smooth out noise. However, this introduced a new challenge: if a person stands still, they slowly blend into the background. When they leave, the algorithm detects their absence as a new motion (a "ghost"). We mitigated this by fine-tuning the learning rate to 0.2, balancing responsiveness with stability.
2. **Flipper Zero Thread Crashes:** Our custom Flipper RFID app initially caused the device to crash and reboot upon exit with a furi_check failed error. This was due to improper resource management where we attempted to destroy the RFID worker thread while its callback was still executing. We resolved this by implementing an Event Loop pattern with a message queue, allowing the main thread to safely stop the worker and clean up resources only after all events were processed.

2. Feature Table

Description	Host/Target	Comp	Code	Author(s)	Notes
RFID Authentication	Other (Flipper)	5	C (Furi)	Palash	Custom Flipper app (.fap) reads 125kHz tags and sends Hex UID over UART to BeagleY-AI.
PIN Code Entry	Target (Beagle)	5	C	Sam	Reused as1 code, reads a string of input on joystick
Motion Detection	Target (Beagle)	4.5	C	Palash	Software-based comparison of JPEG frames. Works well
Video Stream	Other (ESP32)	4.5	C++ (Arduino)	Palash	ESP32-CAM hosts MJPEG server. Auto-adjusts resolution based on PSRAM availability.
Tamper Alarm	Target (Beagle)	5	C	William	Reused code from as3. Triggers alarm if shaking.
Remote Alerts	Target (Beagle)	5	JavaScript	William	Listens for UDP, grabs temp image, sends Rich Embed to Discord Webhook.
UART	Target (Beagle)	5	C	Palash	Custom HAL module for non-blocking serial communication with Flipper Zero.
Audio Feedback	Target (Beagle)	5	C	Sam	Plays distinct WAV files (DingDong, Alarm, Success) using aplay system calls.
Visual Feedback	Target (Beagle)	5	C	Sam	GPIO control of Red/Green LEDs for status indication (Access Granted/Denied).

3. Extra Hardware & Software Used

Hardware

- **Flipper Zero:** Used RFID reader and serial bridge. It reads and decodes the uid of the RFID fob and sends it to beagle board for authentication.
- **ESP32-CAM:** A low-cost Wi-Fi microcontroller module with a camera sensor, used for video generation.
- **Analog Joystick:** A standard 2-axis potentiometer joystick with a push button, used for PIN code entry and doorbell ringing.
- **Accelerometer:** Used for detecting physical tampering/shaking of the device.
- **Status LEDs:** Red and Green LEDs for visual user feedback and as door emulation for the project.
- **Speaker:** Audio out for various actions like doorbell sound, correct/incorrect pin entered, and tamper alarm

Software Libraries & Tools

- **libjpeg (C):** Used on the BeagleY-AI to decompress JPEG images into raw RGB buffers for pixel-level motion analysis.
- **Furi SDK (C):** The firmware development kit for the Flipper Zero, used to access the lfrfid and furi_hal_serial (UART) subsystems.
- **Arduino ESP32 Core:** Used to program the ESP32-CAM, utilizing the esp_camera driver and WiFi libraries.
- **Node.js Packages:**
    - axios: For making HTTP POST requests to the Discord API.
    - form-data: For constructing multipart payloads to upload images.
    - dgram: For implementing the UDP server listener.
- **Linux System Tools:**
    - wget: For efficient non-blocking downloading of images from the camera.
    - aplay: For playback of WAV audio files via the ALSA sound architecture.