# ECEN 361 Project Definition

PHREAKS 4 SECURITY

NICHOLAS BASTION, JOEL BRAUTIGAM, KEIN LIKA, JAMES GREEN

## Project Scope

***What It Is:***

*This project is a **smart home security system** built using embedded systems technology. It combines hardware sensors, a microcontroller, and a user interface to monitor and protect residential space. The system will be capable of detecting unauthorized entry, monitoring environmental conditions, and providing real-time feedback and control to the user.*

***What It Will Do:***

- ***Monitor Entry Points:***
  *Use magnetic reed switches to detect when doors or windows are opened.*
- ***Detect Motion:***
  *Use PIR (Passive Infrared) sensors to detect movement within the home.*
- ***Trigger Alerts:***
  *Activate alarms (buzzer, LED, or notifications) when suspicious activity is detected.*
- ***User Interaction via UI:***
  *Provide a graphical user interface (GUI) or mobile/web dashboard where users can:*
  - *Arm/disarm the system.*
  - *View real-time sensor data.*
  - *Receive alerts and logs.*
  - *Configure system settings.*
- ***Optional Features:***
  - *Environmental monitoring (e.g., temperature, smoke, gas).*
  - *Remote access via Wi-Fi or cloud services.*
  - *Battery backup or low-power operation.*

***Why It Is Important:***

- ***Real-World Relevance:***
  *Home security is a critical concern, and this project simulates a practical application of embedded systems in everyday life.*

- ***Skill Development:***
  *Students will gain hands-on experience with:*
    - *Sensor integration.*
    - *Microcontroller programming.*
    - *Communication protocols (e.g., UART, I2C, Wi-Fi).*
    - *UI/UX design and development.*
    - *System debugging and testing.*
- ***Interdisciplinary Learning:***
  *Combines knowledge from electronics, software engineering, networking, and human-computer interaction.*
- ***Scalability and Innovation:***
  *The project can be expanded with features like facial recognition, voice control, or AI-based threat detection, encouraging creativity and deeper learning.*

# Lessons to Learn

*To successfully complete this project, the team will need to explore and understand several core concepts and technologies related to embedded systems, software development, and system integration. These include:*

***1. Sensor Integration and Signal Processing***

- *Learn how to interface various sensors (PIR, reed switches, temperature, smoke) with a microcontroller.*
- *Understand how to read and process digital and analog signals.*
- *Implement debouncing and filtering techniques to ensure reliable sensor readings.*

***2. Microcontroller Programming***

- *Gain proficiency in programming microcontrollers (e.g., ESP32, Arduino) using C/C++.*
- *Learn how to manage GPIO pins, interrupts, timers, and power modes.*
- *Implement real-time logic for monitoring and responding to sensor inputs.*

***3. Communication Protocols***

- *Learn how to use serial communication protocols such as UART, I2C, and SPI for sensor and peripheral interfacing.*
- *Explore wireless communication methods like Wi-Fi or Bluetooth for data transmission to the UI.*
- *Understand how to implement MQTT or HTTP protocols for cloud connectivity and remote access.*

***4. User Interface Development***

- *Learn how to design and build a responsive GUI using tools like:*
    - ***Web technologies*** *(HTML/CSS/JavaScript) for browser-based dashboards.*
    - ***Mobile frameworks*** *(e.g., Flutter, React Native) for app development.*
    - ***Desktop environments*** *(e.g., Python with Tkinter or PyQt).*
- *Understand how to structure UI for usability, real-time updates, and secure control.*

***5. Power Management***

- *Learn how to implement power-efficient designs using sleep modes and interrupts.*
- *Explore battery backup systems and energy-saving techniques for embedded devices.*

***6. System Integration and Testing***

- *Understand how to integrate hardware and software components into a cohesive system.*
- *Learn how to test embedded systems for reliability, responsiveness, and fault tolerance.*
- *Develop debugging skills using tools like serial monitors, logic analyzers, and simulation environments.*

***7. Security and Data Protection***

- *Learn basic principles of embedded system security, including:*
    - *Secure communication (e.g., encryption for Wi-Fi).*
    - *Authentication for UI access.*
    - *Safe handling of user data and logs.*

# Roles and Responsibilities

*Define roles and responsibilities for each team member and report them here. Note the team leader is required, but the others are provided just as examples.*

| Role | Name | Responsibilities |
|---|---|---|
| **Team Leader** | Joel Brautigam | • Set up project in Monday.com and invite team<br>• Manage project tasks in Monday.com<br>• Submits group assignments<br>• Establishes meeting schedule (1-2 times per week) |
| **Hardware Lead** | Nicholas Bastion<br>James Green | • Final decision maker on hardware selection<br>• Hardware block diagram owner<br>• Gathers necessary hardware (purchase or loan) |

| | | |
|---|---|---|
| **Software Lead** | Kein Lika | • Final decision maker on software architecture<br>• Software block diagram owner<br>• Set up Git repository and share with team |

# Schedule

*Create initial set of tasks in Monday.com and assign a person and dates and durations. Export main table to Excel and Gantt chart to PDF and turn in with this sheet.*

# Derived Requirements

*List the requirements for your project. Derived requirements are typically broken down into three categories: general, interface, and functional.*

## General Requirements

**Form Factor**

- Small, contained units.
- Sensors are individual.
- Wired connections

**Power**

- Runs off 120v AC.

**Hardware**

- Currently we want the system to run on a Raspberry Pi. For sake of our software lead.

**Interface**

- Some sort of applet could be web based, but it runs locally on the pi.

## Interface Requirements

*Interface requirements detail the interfaces of the system and cover items such as communication standards, protocols, baud/bit rates, cloud connectivity, security, etc. Use these to help guide hardware selection and software design.*

**Communication Protocols**

- **UART** (9600–115200 bps): For serial communication with sensors and actuators.
- **I2C** (100–400 kHz): For low-speed sensor interfacing.
- **SPI** (up to 10 MHz): For high-speed peripherals.
- **Wi-Fi (802.11 b/g/n)**: For wireless connectivity to cloud and UI.

**Cloud & Remote Access**

- **Protocols:** MQTT (port 1883/8883) and HTTPS (port 443).
- **Platform:** AWS IoT, Firebase, or similar.
- **Features:** Real-time data, alerts, remote control via web/mobile dashboard.

**Security**

- **Authentication:** Login required for dashboard access.

**Hardware Interfaces**

- **GPIO:** For sensor and actuator connections.
- **Power:** 120V AC with optional battery backup.

## Functional Requirements

*Functional requirements detail the features and overall functionality of the system and cover items such as specific capabilities and behaviors required to fulfill the scope of the project.*

**Security Monitoring**

- **Entry Detection:** Magnetic reed switches detect door/window openings.
- **Motion Detection:** PIR sensors monitor movement inside the home.
- **Alert System:** Triggers alarms (buzzer, LED, or notifications) upon suspicious activity.

**User Interaction**

- **Dashboard Access:** Users can arm/disarm the system, view sensor data, receive alerts, and configure settings via a local web or mobile dashboard.

**Environmental Monitoring (Optional)**

- **Sensors:** Monitor temperature, smoke, and gas levels.
- **Alerts:** Notify users of abnormal environmental conditions.

**Remote Access**

- **Wi-Fi/Cloud Integration:** Enables system control and monitoring from remote devices.

**Power Management**

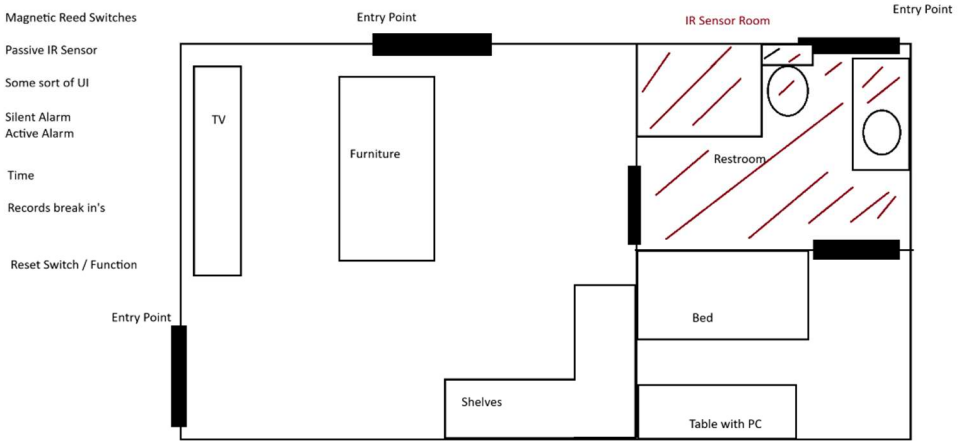
- **Primary Power:** Operates on 120V AC.
- **Backup:** Optional battery support for power outages.
- **Efficiency:** Uses sleep modes and interrupts to conserve energy.

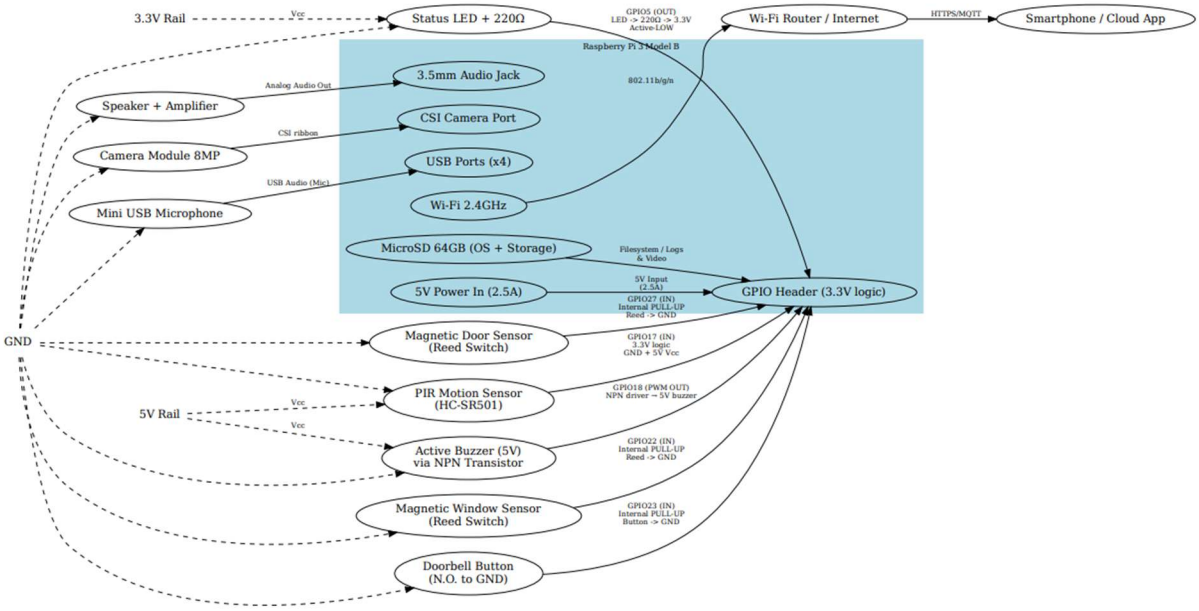
**System Reliability**

- **Real-Time Response:** Immediate reaction to sensor inputs.
- **Data Logging:** Stores event logs for review.
- **Testing & Debugging:** Includes tools like serial monitors and logic analyzers.

# Hardware Block Diagram

*Create an initial, rough hardware block diagram showing the major hardware components, how they connect, how they are powered, etc. Use a professional tool such as LucidChart, draw.io, or Miro for this. Export a PDF and turn in with this sheet.* ***Note that this is not final and will be updated weekly.***

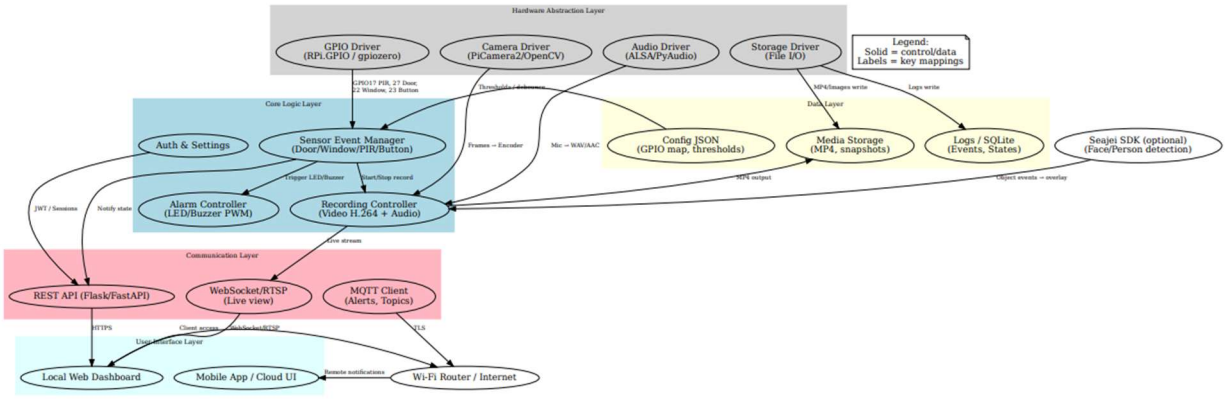


HardwareDiagram_v2.pdf

# Software Block Diagram

*Create an initial, rough software block diagram showing the major software modules, how they connect, how they interface with hardware, etc. Use a professional tool such as LucidChart, draw.io, or Miro for this. Export a PDF and turn in with this sheet.* ***Note that this is not final and will be updated weekly.***




https://drive.google.com/file/d/1B9XFPiyofJJlrwn9pkn7Civo2eum8l0D/view?usp=sharing

SoftwareDiagram_v2.pdf