

# Smart Home Security System

Team members: Megha, Purab, Niraj

## Introduction

The Smart Home Security System is an advanced and intuitive solution designed to enhance the security and convenience of modern homes. Leveraging the capabilities of motion detection, camera activation, and person recognition, this system provides an automated approach to home security. The core components include a motion sensor, a camera module, a microcontroller (Arduino), and a user feedback display.

Upon detecting motion, the system activates the camera to capture an image and sends it to a server for processing. If the person is recognized as a household member, the door automatically unlocks, providing seamless and secure access. In cases where the person is not recognized, the system rings a doorbell and displays a message indicating that the individual is not identified.

This project aims to address the limitations of traditional security systems by offering an affordable, easy-to-use, and reliable smart home security solution. It emphasizes key design constraints such as ease of installation, cost-effectiveness, durability, and data security.

By integrating cutting-edge technology and user-centric design, the Smart Home Security System represents a significant advancement in home automation and security, providing homeowners with peace of mind and convenience.

# Problem statement

1. Traditional security systems are reactive:
  - They often respond only after a break-in has occurred.
  - They require manual activation/deactivation.
2. Lack of automated access control:
  - Traditional systems do not integrate automated person recognition.
3. Manual user interaction:
  - Users must manually check cameras or peepholes.
  - No immediate feedback is provided to visitors.
4. Increasing security concerns:
  - Home invasion and burglary rates necessitate improved security measures.

## Target Audience

Homeowners looking for enhanced security:

- Individuals concerned about home security
- Tech-savvy users interested in smart home systems
- Families wanting convenient and automated access control
- Offices or small businesses interested in enhancing access control.
- Those who are early adopters of new technologies and innovations in home security.
- Schools, colleges, and universities interested in improving campus security.
- Government buildings requiring secure access for employees and visitors.

# Design Constraints

## Ease of Use:

- The system must be simple to install and operate.

## Cost-Effective:

- Components should be affordable to make the system accessible to a wide audience.

## Reliability:

- The system must function accurately and consistently.

## Security:

- Data transmission and storage must be secure.

## Compatibility:

- The system should integrate with existing home networks and devices.

## Flaws in Existing Solution

Despite significant advancements, present-day smart home security systems still face several flaws that can affect their effectiveness and user experience.

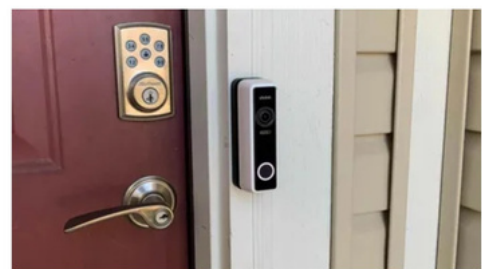
### Vivint smart Home

#### PROS

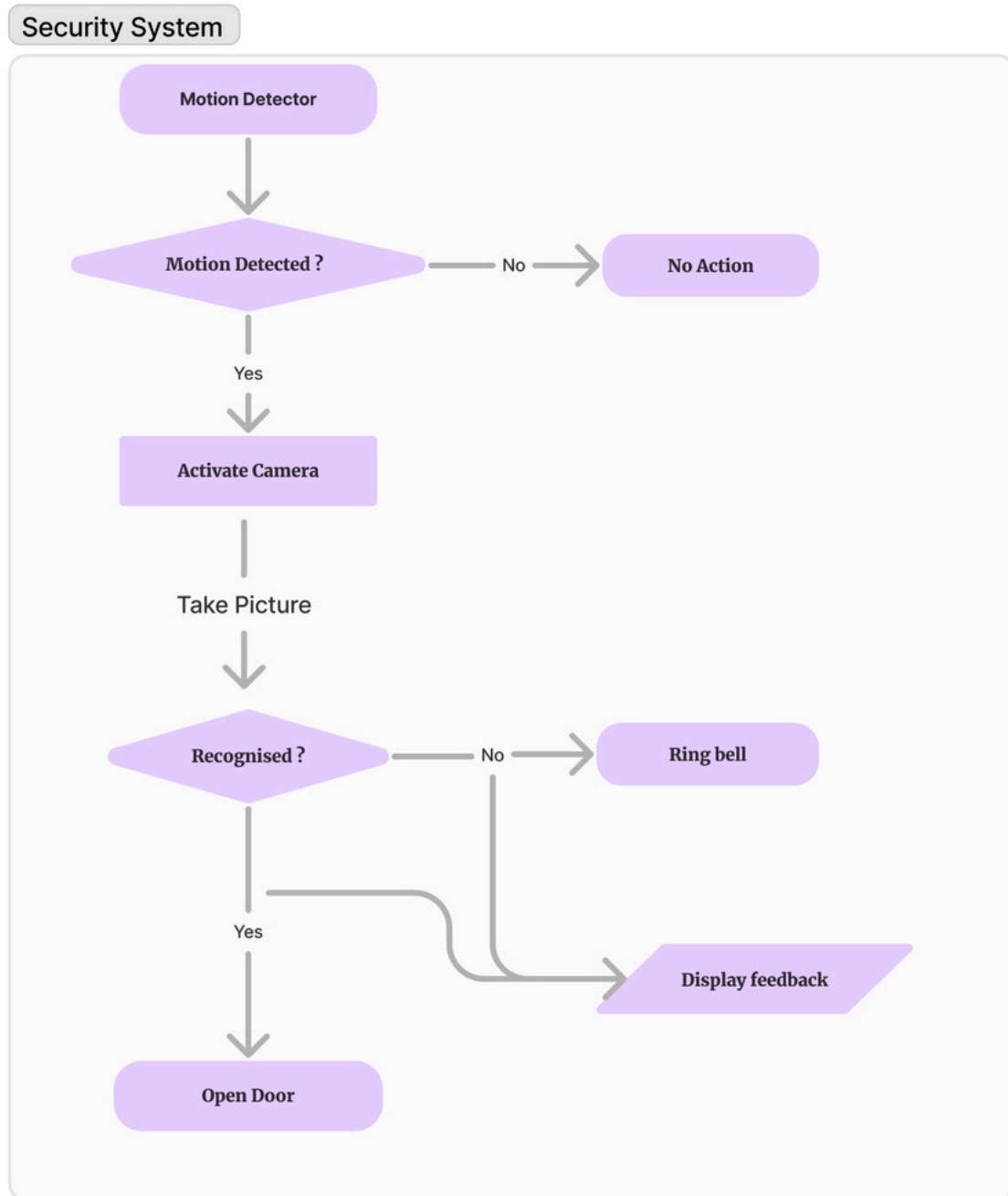
- Seamless integration
- Numerous third-party device integrations
- No contract required

#### CONS

- High upfront charges
- Difficult to decipher pricing without having to give your email address
- Additional Fees
- 24/7 professional monitoring fee, storage fee



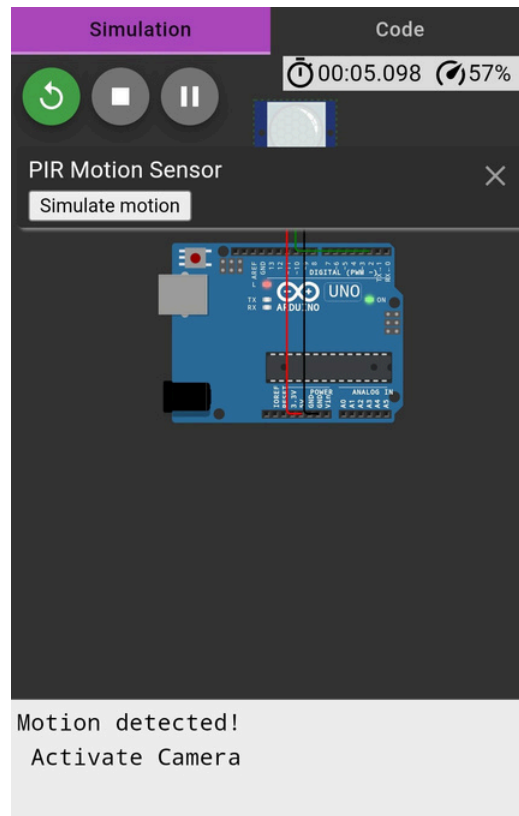
# Work flow of our Project



# Electronics

## Motion detection + Camera activation

<https://wokwi.com/projects/402579658200252417>



- We have created a motion detection system using an Arduino, a PIR sensor and an LED.
- The code reads the output from the PIR sensor to detect motion.
- When motion is detected ,the LED is turned on and a message is printed on the Serial Monitor.
- The LED stays on for 5 seconds to avoid multiple triggers .
- If no motion is detected, LED is turned off.

# User Authentication and Management System

<https://wokwi.com/projects/402730525426607105>



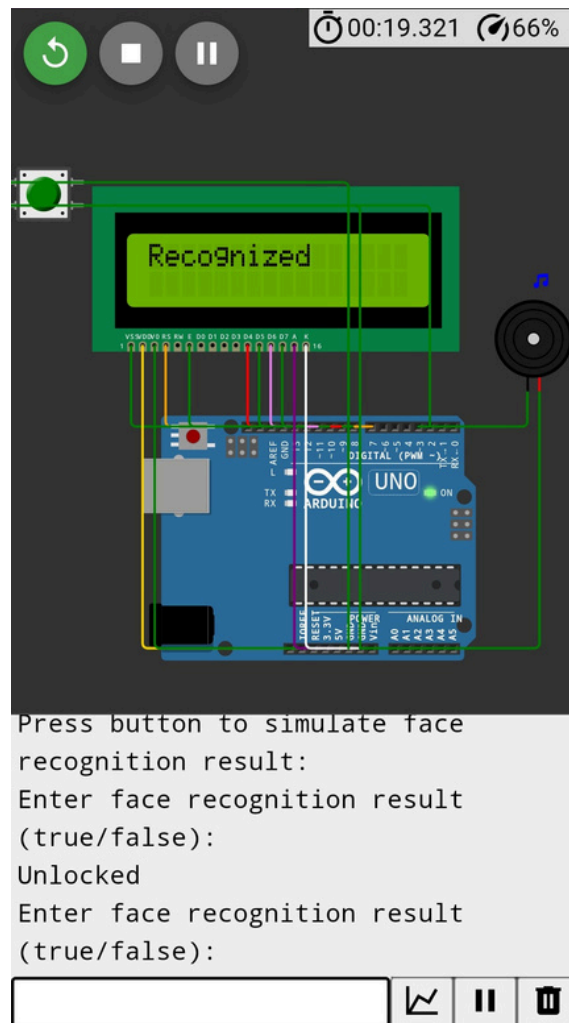
- We have created a user authentication system using an Arduino where certain users can be verified and only admin can add new users or change the password after verification.

## User Interaction:

- The user is prompted to enter their name.
- If the name matches an existing identity, a welcome message is displayed.
- If the name does not match, an "**Unknown identity**" message is displayed.
- If User (Megha) is identified, she can choose to add a new identity or change the password by entering the current password

# Unlocking/Locking the door

<https://wokwi.com/projects/402659252685206529>



- We have used Serial Monitor to simulate receiving the output from the facial recognition system.
- Based on the input, the Arduino will either "unlock" (print"unlocked"), display "Recognized" on the screen, or ring a buzzer and display "Unrecognized".
- If the input is neither true nor false the screen displays invalid input and rings the buzzer

# Software

## Face Recognition System

**Project files :** [click to access files](#)

This document outlines the steps and code used to create and implement the **facial recognition** component of our **smart home security system**. The system includes setting up a database of known faces, capturing images from a camera, and matching them against the known faces to identify individuals.

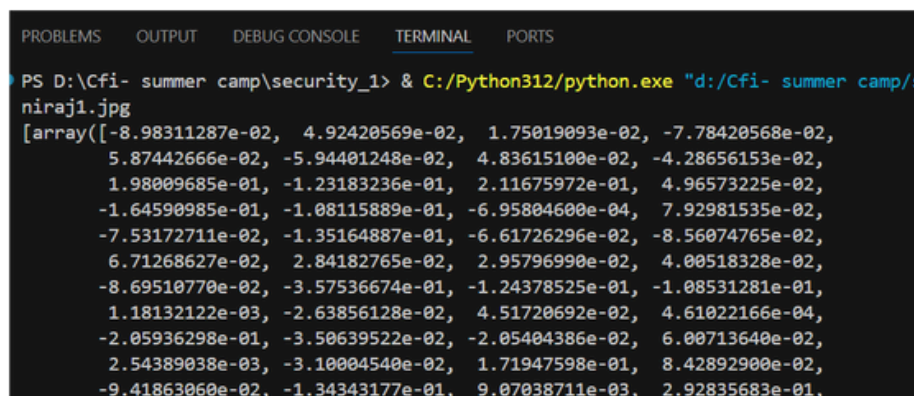
### Setting Up the Database

Setting up the database for person recognition involves collecting images of household members, encoding these images into numerical representations (face encodings), and storing them for future comparisons

### Directory Structure

The database of known faces is organized into a directory where each subdirectory represents a person.

```
known_faces/
├── person_1/
│   ├── image1.jpg
│   ├── image2.jpg
│   └── ...
├── person_2/
│   ├── image1.jpg
│   ├── image2.jpg
│   └── ...
└── ...
```



The screenshot shows a terminal window with the following content:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\Cfi- summer camp\security_1> & C:/Python312/python.exe "d:/Cfi- summer camp/
niraj1.jpg
[array([-8.98311287e-02,  4.92420569e-02,  1.75019093e-02, -7.78420568e-02,
        5.87442666e-02, -5.94401248e-02,  4.83615100e-02, -4.28656153e-02,
        1.98009685e-01, -1.23183236e-01,  2.11675972e-01,  4.96573225e-02,
        -1.64590985e-01, -1.08115889e-01, -6.95804600e-04,  7.92981535e-02,
        -7.53172711e-02, -1.35164887e-01, -6.61726296e-02, -8.56074765e-02,
        6.71268627e-02,  2.84182765e-02,  2.95796990e-02,  4.00518328e-02,
        -8.69510770e-02, -3.57536674e-01, -1.24378525e-01, -1.08531281e-01,
        1.18132122e-03, -2.63856128e-02,  4.51720692e-02,  4.61022166e-04,
        -2.05936298e-01, -3.50639522e-02, -2.05404386e-02,  6.00713640e-02,
        2.54389038e-03, -3.10004540e-02,  1.71947598e-01,  8.42892900e-02,
        -9.41863060e-02, -1.34343177e-01,  9.07038711e-03,  2.92835683e-01,
```

### Encoding Known Faces

To encode the known faces, we use the `face_recognition` library. The following script processes each image in the `known_faces` directory, encodes the face, and saves the encodings in a pickle file.



## Facial Recognition Script

The script recognition.py captures an image from the webcam, encodes the face, and matches it against the known faces in the database.

### Script Functionality

- Capture and Save Unknown Face: Open the webcam, capture an image, and save it in the unknown\_faces directory.
- Serialize Captured Image: Serialize the captured image using pickle.
- Match Encoding with Database: Compare the encoding of the captured image with the known face encodings stored in the pickle file.
- Output: Print the matched name and confidence score or indicate if no match was found.

Samples of outputs

If visitor recognised

```
PS D:\Cfi- summer camp\security_1> & C:/Python312/python.exe "d:/Cfi- summer camp/security_1/recognition.py"
Captured unknown face saved: unknown_faces\captured_unknown_face.jpg
Matched name: niraj
Confidence Score: 0.71
Person recognized: True
```

If visitor is not recognised

```
PS D:\Cfi- summer camp\security_1> & C:/Python312/python.exe "d:/Cfi- summer camp/security_1/recognition.py"
Captured unknown face saved: unknown_faces\captured_unknown_face.jpg
No matching face found in the database.
Person recognized: False
```

If issue with detecting face

```
PS D:\Cfi- summer camp\security_1> & C:/Python312/python.exe "d:/Cfi- summer camp/security_1/recognition.py"
Captured unknown face saved: unknown_faces\captured_unknown_face.jpg
No face found in the captured image.
Person recognized: False
```

### To run the programme...

1. Ensure the known\_faces directory is structured correctly with subdirectories for each person containing their images.
2. Run the database setup script to encode and save the known faces.
3. Run the recognition.py script to capture an image from the webcam, encode it, and match it against the known faces.

**Project files can be accessed here:**

<https://drive.google.com/drive/folders/1UZRV271na8Fd1EdqIEokBYUdKIUXXCv3?usp=sharing>

## Possible Flaws, need to ponder over (for our project)

- What happens, when there is no one in home?
- The system might be tricked by using high-resolution images or videos of an authorized person
- Aging: Over time, as people age, their facial features may change.
- The system might struggle to distinguish between identical twins.
- If the system does not have a backup power source, it will be unusable during power outages.

## References

- python documnetation
  - <https://docs.python.org/3/>
- Wokwi documentation
  - <https://docs.wokwi.com/>
- arduino documentation
  - <https://docs.arduino.cc/>

