Sistem Usa Smart

Introducere

Un sistem pentru o usa inteligenta ce va avea următorul comportament:

- Senzorul de miscare va detecta prezenta unei persoane, iar camera va incepe se capteze imagini
- Imaginea persoanei din fata ușii va fi transmisa catre server
- Backend-ul va decide dacă persoana este autorizată iar în caz afirmativ se va deschide usa
- În caz contrar, proprietarii vor primi notificare pe aplicația mobila care va afișa fata persoanei ce așteaptă în fata ușii
- Moment în care se poate aproba deschiderea ușii sau nu
- Dacă usa se deschide cu forța(fără aprobare sau recunoaștere), în spatele acesteia vor exista 3 senzori de miscare ce vor detecta intrarea prin efracție și vor atenționa proprietarii cu o notificare pe mobil și se va porni o alarma

Prezentare

Proiectul consta intr-un sistem de securitate pentru usi bazat pe recunoastere faciala si control remote asupra usii.

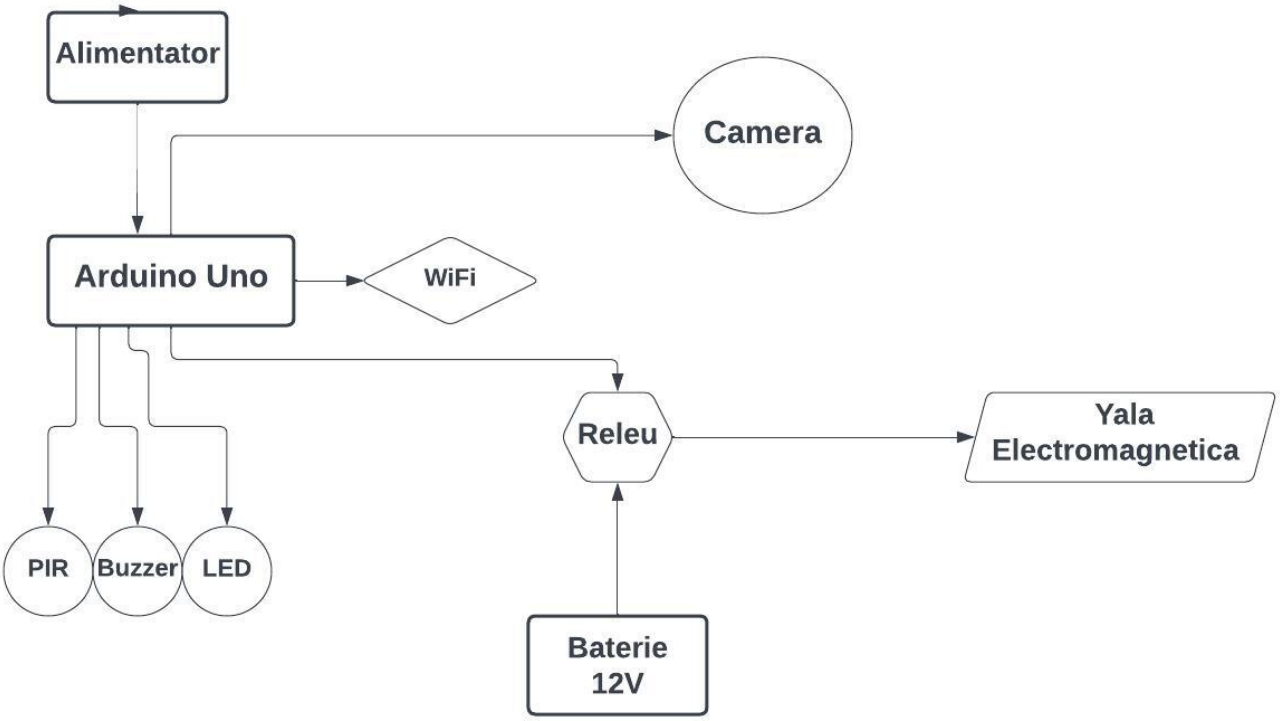
Ideea

Sunt o persoana foarte dezordonata care uita/pierde cheile tot timpul. Asa ca m-am gandit sa-mi usurez viata folosind acest sistem.

Descriere generală

Placuta de Arduino este alimentata la o sursa obisnuita introdusa in priza. Senzorii (de proximitate) sunt conectati la placuta si vor citi mereu datele din mediul inconjurator. Modulul de camera este conectat la placuta si asteapta ca senzorii sa detecteze miscare pentru a transmite imaginile catre server. Un releu conectat la placuta si o sursa pentru alimentarea yalei electromagnetice.

Schema Bloc



Hardware Design

- Arduino Uno
- Senzori de mișcare PIR
- Modul de cameră OV7670
- Modul de comunicație wireless ESP8266
- Modul Releu canal 5V
- Baterie de 12V + conector baterie
- Buzzer
- LED-uri
- Yala electromagnetica
- Alimentator
- Fire mama-mama, mama-tata, tata-tata

Schema Circuit

Aici puneţi tot ce ţine de hardware design:

- listă de piese
- scheme electrice (se pot lua şi de pe Internet şi din datasheet-uri, e.g. http://www.captain.at/electronic-atmega16-mmc-schematic.png)
- diagrame de semnal

- rezultatele simulării

Software Design

Mediu De Dezvoltare

- Arduino IDE - pentru programarea placutei
- IntelliJ - pentru programarea comunicatiei seriale
- VSCode - programarea serverului(Python - Flask) si programarea aplicatiei mobile(Flutter)

Biblioteci

- Adafruit_ST7735
- Am preluat un cod scris sub forma de biblioteca pentru interactiunea cu modulul de camera OV7670

Organizarea Codului

Arduino

```
#include "setup.h"

// defines pins numbers
const int trigPin = 9;
const int echoPin = 10;
const int outPin = 11;
// defines variables
long duration;
int distance;

void setup() {
  CLKPR = 0x80; // enter clock rate change mode
  CLKPR = 0; // set prescaler to 0. WAVGAT MCU has it 3 by default.

  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  pinMode(outPin, OUTPUT);
  Serial.begin(9600); // Starts the serial communication

  initializeScreenAndCamera();
}
```

```
void loop() {

  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance = duration * 0.034 / 2;

  if(distance < 20){
    for(int i = 0; i < 10; i++){
      char serverResponse = processFrame();
      if(serverResponse == 1){
        digitalWrite(outPin, HIGH);
        delay(25000);
        digitalWrite(outPin, LOW);
        break;
      }
    }
  }
}
```

- Se asteapta pana cand sensorul infra-rosu detecteaza ceva la 20cm distanta, camera incepe sa inregistreze, iar in cazul in care un frame contine fata unei persoane autorizate serverResponse va deveni 1 iar incuietoarea se va deschide pentru cateva secunde.
- In cadrul acestui cod, functia “initializeScreenAndCamera()” a fost lasata intacta, in schimb ceea ce am modificat din aceasta “biblioteca”, a fost functia “processFrame()” unde am impus asteptarea unui raspuns pe portul serial

```
char processFrame() {
  processedByteCountDuringCameraRead = 0;
  commandStartNewFrame(uartPixelFormat);
  noInterrupts();
  processFrameData();
  interrupts();
  frameCounter++;
  commandDebugPrint("Frame " + String(frameCounter)/* + " " +
String(processedByteCountDuringCameraRead)*/);

  // wait for the server to check the frame
  while(!Serial.available()){}
  char serverResponse = Serial.read();

  return serverResponse;
}
```

- Aici, fiecare frame este trimis pe portul serial unde ajunge sa fie procesat de o aplicatie in Java si ulterior trimis catre backend

Java

Functia adaugata de mine este urmatoarea:

```
  private void sendImageToServer(BufferedImage image, File toFolder) {
    try {
      // Save image to PNG file
      File newFile = new File(toFolder.getAbsolutePath(), getNextFileName());
      ImageIO.write(image, "png", newFile);

      // Read the image file as bytes
      byte[] imageData = Files.readAllBytes(newFile.toPath());

      // Encode the image bytes as Base64
      String base64Image = Base64.getEncoder().encodeToString(imageData);

      // Set up the URL and connection
      URL url = new URL("http://localhost:5000/upload");
     HttpURLConnection connection = (HttpURLConnection) url.openConnection();
      connection.setRequestMethod("POST");
      connection.setRequestProperty("Content-Type", "application/json");
      connection.setDoOutput(true);

      // Create the JSON payload
      JSONObject jsonPayload = new JSONObject();
      jsonPayload.put("image", base64Image);

      // Write the JSON payload to the request body
      OutputStream outputStream = connection.getOutputStream();
      outputStream.write(jsonPayload.toString().getBytes());
      outputStream.flush();

      // Read the response from the server
      int responseCode = connection.getResponseCode();
      if (responseCode == HttpURLConnection.HTTP_OK) {
        System.out.println("Server response: OK");
        serialReader.sendResponse(1);
      } else {
        System.out.println("Unexpected server response code: " +
responseCode);
        serialReader.sendResponse(0);
      }

      // Close the connection
      connection.disconnect();
```

```
      saveCountLabel.setText(" (" + (++saveCounter) + ")");
    } catch (IOException e) {
      System.out.println("Sending file failed: " + e.getMessage());
    }
  }
```

- Aici are log un request http in urma caruia se transmite inapoi la placuta raspunsul serverului

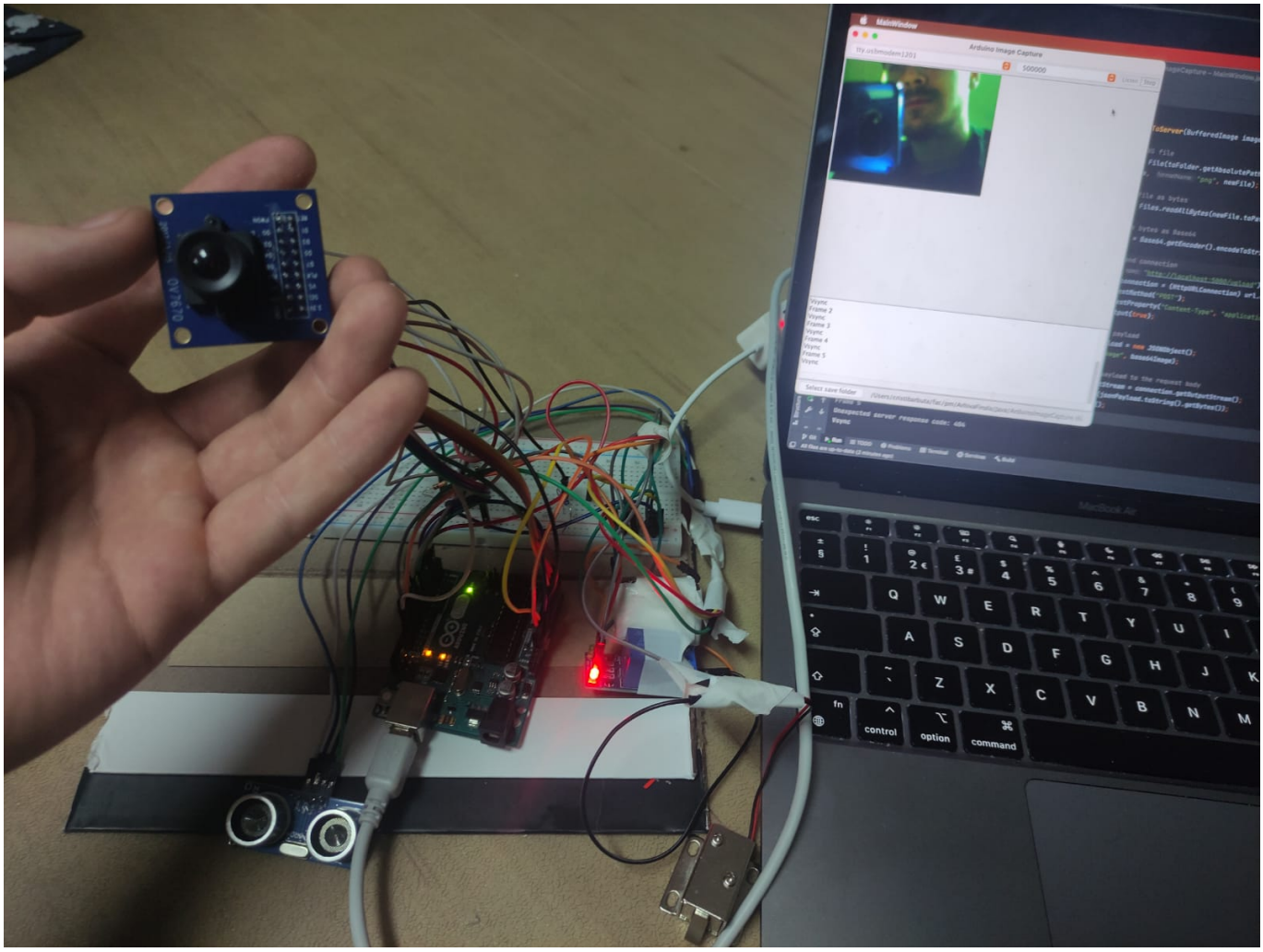
Flask - Flutter

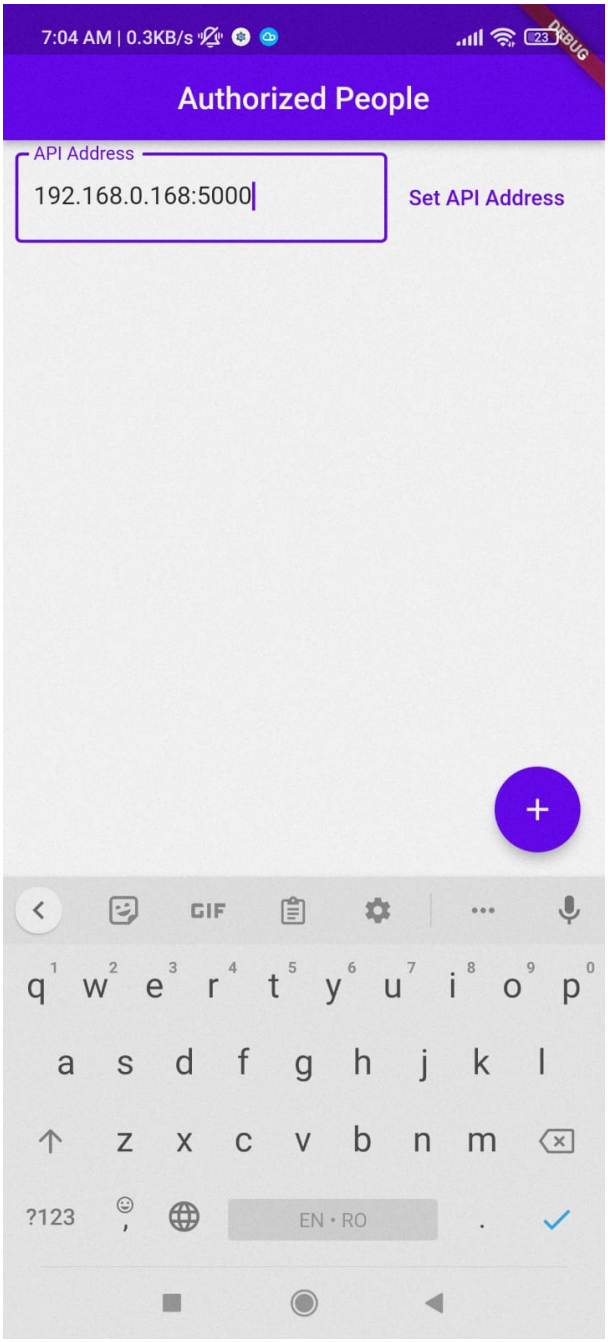
Am facut o aplicatie full-stack minimalista care are urmatoarele functionalitati in cadrul proiectului

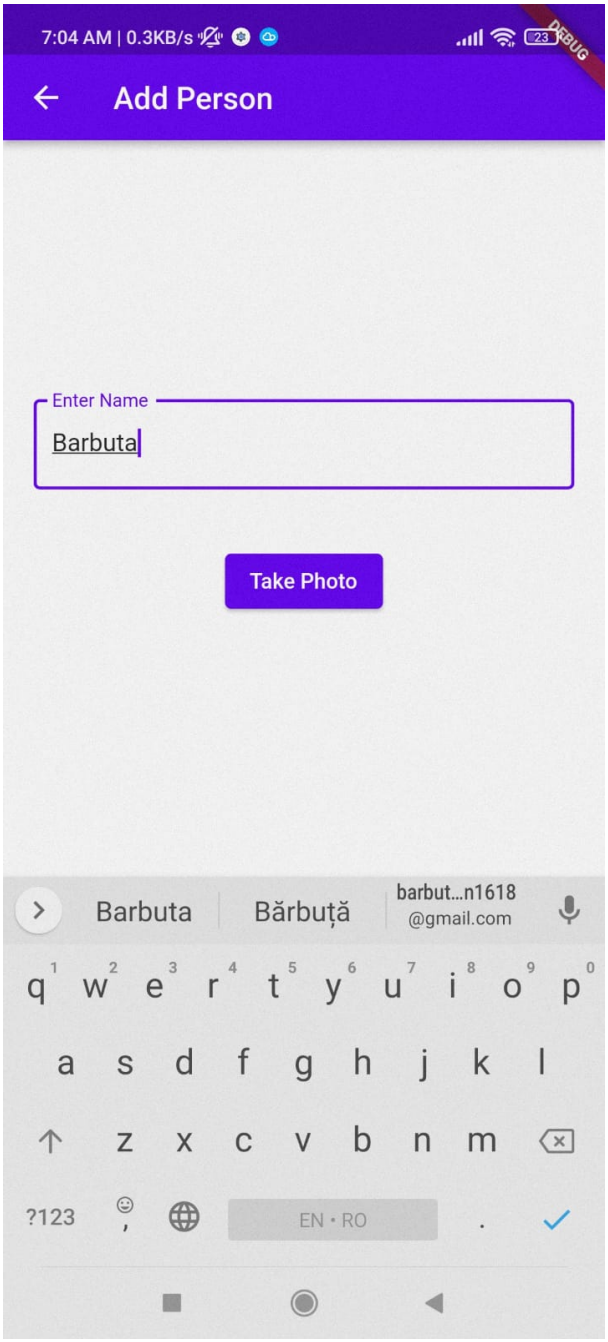
- listarea persoanelor autorizate
- adaugarea de persoane autorizate: Nume - Selfie
- stergerea de persoane autorizate
- recunoasterea persoanelor autorizate

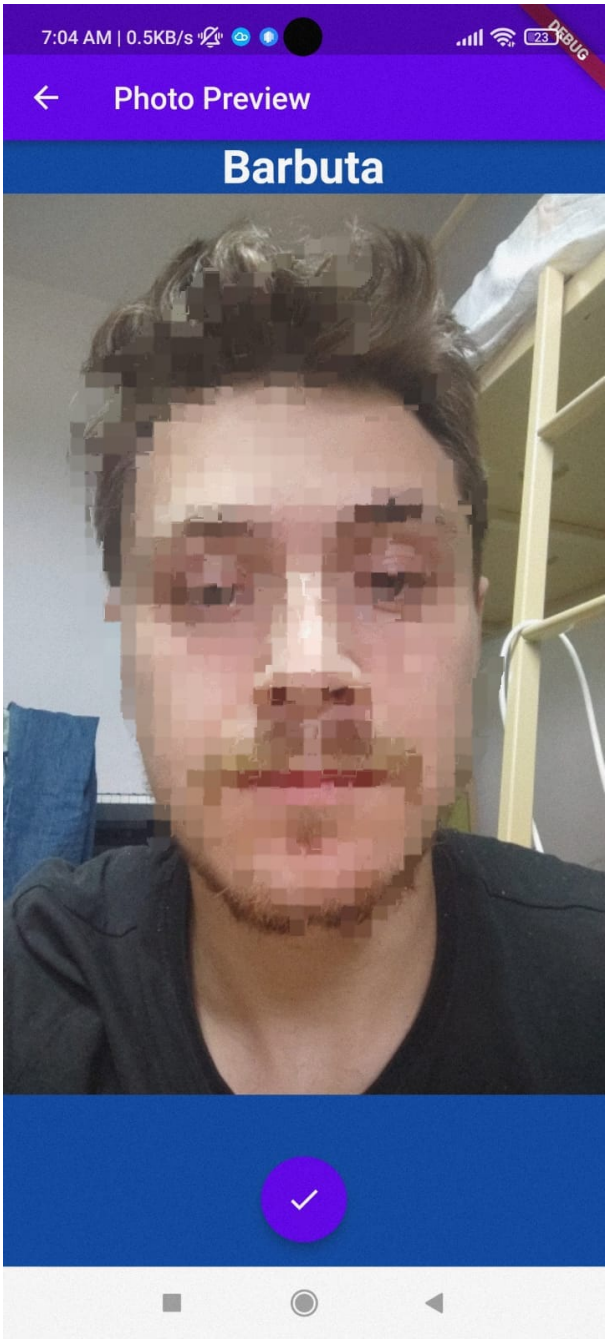
Rezultate Obţinute

In urma proiectului, am obtinut o usa inteligenta care permite intrarea persoanelor autorizate si o aplicatie mobila pentru gestionarea usii.


MainWindow
Arduino Image Capture
MacBook Air
esc
control
option
command


7:04 AM | 0.3KB/s
DEBUG
Authorized People
API Address
192.168.0.168:5000
Set API Address
GIF
?123
EN • RO


7:04 AM | 0.3KB/s
DEBUG
Add Person
Enter Name
Barbuta
Take Photo
Barbuta
Bărbuță
barbut...n1618
@gmail.com
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123
EN • RO



- videoclip cu functionalitatea
- <a>https://drive.google.com/file/d/1ctDuG0gpRxGoAZwBggntORs-Z0veVhDa/view?usp=sharing

Concluzii

A fost un proiect interesant in care am intampinat foarte multe probleme din care am putut invata. Totodata, pe langa cunostintele practice obtinute, am dobandit si o intelegere teoretica asupra diferitelor concepte utilizate.

Download

barbuta_app_final.zip
barbuta_arduino.zip
barbuta_java_final.zip

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe *Resurse Software* şi *Resurse Hardware*.

Export to PDF

From:
https://ocw.cs.pub.ro/courses/ - **CS Open CourseWare**

Permanent link:
https://ocw.cs.pub.ro/courses/pm/prj2023/apredescu/barbuta.cristian

Last update: **2023/05/29 05:32**