Smart Home box

Introducere

Prezentare succinta proiect:

Proiectul consta in realizarea unui sistem de control al luminilor si al prizelor. Monitorizare consum curent, temperatura, umiditate, calitate aer. Deschiderea ușii se va realiza prin introducerea unei parole pe un keypad sau folosind un card rfid. Controlul si monitorizarea se va face de pe o aplicația realizata in flutter si folosind o telecomandă ir. Va conține o cameră esp-cam, care va fi controlata cu doua servomotoare. Alarma se va activa atunci când pătrunde cineva in casa fără sa introducă parola/cardul sau când calitatea aerului este foarte rea. Pentru controlul intensității luminii am creat un dimmer ce folosește pwm, luminile pot fi controlate si bătând din palme, de la telecomanda sau din aplicație. De asemenea putem seta o anumita ora/interval sau atunci când intensitatea luminii scade sub un anumit nivel când sa se aprindă/stingă luminile.

De ce este util un astfel de proiect ?

Prin simplificarea excesivă a proceselor obositoare și obișnuite întâlnite în casele tradiționale, aceasta aduce un nivel de confort și eficiență care poate să nu fie experimentat în altă parte. De exemplu, prin integrarea tehnologiilor inteligente și a automatizării, sarcinile precum controlul temperaturii, securitatea locuinței sau gestionarea energiei sunt gestionate în mod automat și convenabil. Aceste facilități nu numai că economisesc timp și efort, dar și oferă un sentiment de confort și siguranță, lăsându-i pe locatari să se concentreze pe alte aspecte ale vieții lor.

Descriere generală

Main Box :

Aceasta este cutia principala care se va conecta la automatele din casa pentru controlul prizelor, becurilor, masurare curent, interactionare cu baza de date.

Box 1 Aceasta este prima cutie care se va monta intr-o camera si care contine mai multi senzori pentru monitorizare si control. Ea va comunica direct cu placuta principala utilizand pinii GPIO.

Box 2

Este la fel ca prima cutie numai ca comunica prin wifi.

Door Lock Box

Usa va putea fi deschisa folosind cardul sau introducand parola pe un keypad. Din meniul acestuia putem schimba parola, schimba limba, adauga/sterge carduri. Datele sunt salvate in EEPROM. Pentru a detecta daca usa a fost deschisa voi folosi un Sensor Hall. Interfonul va comunica cu placuta principala prin UART pentru a trimite semnale de alarma.

Esp-cam

Este o camera directia careia va fi controlata cu 2 servo-motoare.

Hardware Design

Lista piese:

- Esp32 x2
- Esp-cam
- Arduino Nano
- Placa PCB prototipare
- Relee x4
- Header pini
- Conectori XH2.54
- Conectori screw-in x20
- Convertor 220V-5V
- Current sensor CT 013
- LEDs x3
- Cablu UTP ~1m
- Cablu eletric 2.5mm ~2m
- Power button
- Fotorezistori x2
- Telecomanda IR
- Sensor IR x2
- Sensor DHT11 x2
- Buzzer x2
- Sensor MQ-2 x2
- Microfon x2
- Acumulator 18650 x2
- Convertor Step-Up
- Modul incarcare
- LCD 16×2
- Keypad
- Servo-motor x3
- Sensor Hall
- Magneti

- Modul RFID
- MOSFET
- Punte redresoare
- Condensator 100uF
- Dioda Zenner 15V
- Dioda
- Resistori 100k, 10k, 330, 220
- Optocoupler

Scheme electrice

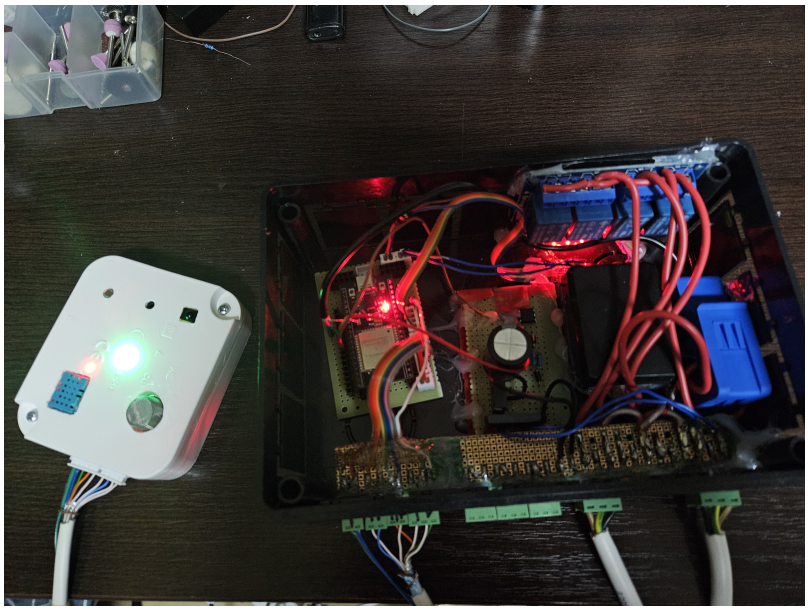
Schematics:

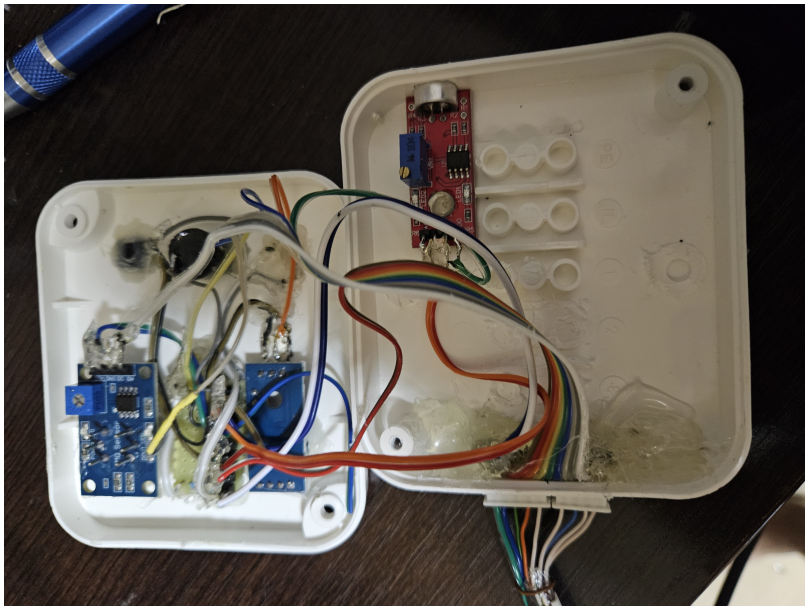
Main_box + Box1

Interfon + Camera

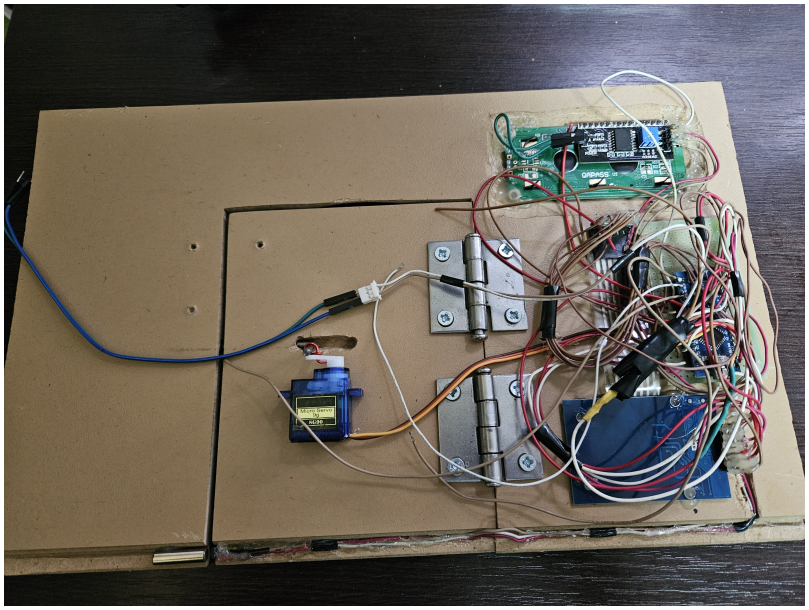
Box 2

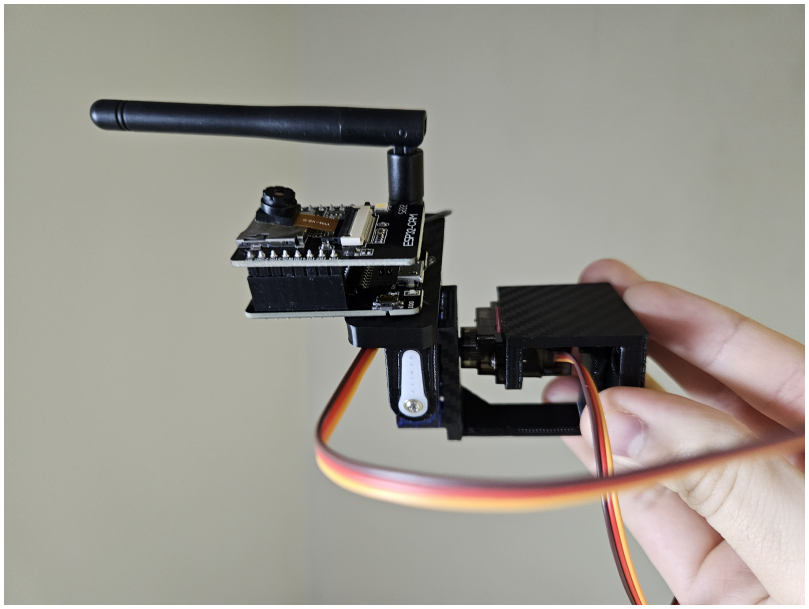
Main Box + Box 1



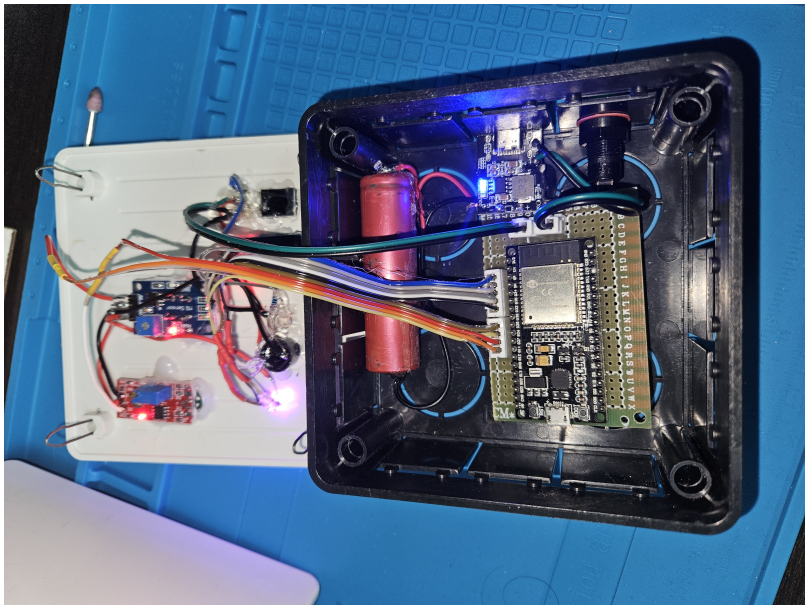


Interfon + Camera





Box 2



Software Design

Descrierea codului aplicaţiei:

Medii de dezvoltare:

```
VS Code - Platformio : Programare placute
Android-Studio: Aplicatie telefon
```

Biblioteci utilizate:

```
WiFi.h
Adafruit_Sensor.h
EmonLib.h
driver/adc.h
Firebase_ESP_Client.h
addons/TokenHelper.h
IRremote.h
SPI.h
MFRC522.h
EEPROM.h
```

Baza de date

Baza de date folosita pentru a stoca informatia si pentru a comunica cu telefonul este Firebase

Comunicare esp cu baza de date:

```
FirebaseAuth auth;
FirebaseConfig config;
FirebaseData fbdo;
```

```
.............. (autentificare si initializare)
```

Trimitere BD:

```
Firebase.RTDB.setInt(&fbdo, "Main_box/power_measure", mainBox.power_measure)
```

Citire BD:

```
Firebase.RTDB.readStream(&fbddo_light_intensity)
```

```
if (fbdo_light_intensity.streamAvailable()) {
   if (fbdo_light_intensity.dataType() == "int")
   {
     mainBox.light_intensity = fbdo_light_intensity.intData();
   }
 }
```

Comunicare aplicatie cu baza de date:

```
private DatabaseReference database =
FirebaseDatabase.getInstance().getReference();
database.addValueEventListener(new ValueEventListener() {
          public void onDataChange(final DataSnapshot dataSnapshot) {

              // Citire BD
              Long temperature = (Long)
dataSnapshot.child("Room2").child("Temperature").getValue();
              Long gassensor = (Long)
dataSnapshot.child("Room2").child("Gas_sensor_value").getValue();

              // Scriere BD

dataSnapshot.getRef().child("Room2").child("Relay1").setValue(true);
}
```

Functie stingere/aprindere lumina batand din palme

Functia verifica daca au fost exact 2 batai intr-un anumit interval de timp

```
void claps_function(int sensorPin, bool *signalToRelayPin) {
  int soundValue = digitalRead(sensorPin);
  currentNoiseTime = millis();
  if (soundValue == 1)
  { // if there is currently a noise
      if (
          (currentNoiseTime > lastNoiseTime + 200) &&
          (lastSoundValue == 0) &&
          (currentNoiseTime < lastNoiseTime + 800) &&
          (currentNoiseTime > lastLightChange + 1000)
         )
      {
        \*signalToRelayPin = !(*signalToRelayPin);
        lastLightChange = currentNoiseTime;
      }
      lastNoiseTime = currentNoiseTime;
  }
  lastSoundValue = soundValue;
}
```

Citire date telecomanda IR

Primeste comenzi de la telecomanda si in functie de acestea seteaza variabilele necesare Pentru a nu citi prea rapid comenzi si nu a trimite date prea des la baza de date an facut un delay pentru fiecare folosind functia millis()

```
  if (IrReceiver.decode()) {
  IrReceiver.resume();
  irCommand = IrReceiver.decodedIRData.command;
  if (millis() - debouncing_delay > 200){

    // up
    if (irCommand == 64) {
      room1.relay1 = !room1.relay1;
      relayStatus = true;
    }
    // ok
    if (irCommand == 70) {
      room1.relay2 = !room1.relay2;
      relayStatus = true;
    }
    // #
    if (irCommand == 74) {
      room1.alarm = !room1.alarm;
      digitalWrite(BUZZER_PIN, room1.alarm);
      if (mainBox.deactivate_alarm) {
        digitalWrite(BUZZER_PIN, LOW);
      }
      update_db_buzzer = 1;
    }

    ................

    debouncing_delay = millis();
    irCommand = 0;
  }
  update_db_time = millis();
}
```

Citire si scriere EEPROM pe Arduino Nano

Parola si cartelele sunt salvate in eeprom pentru a nu pierde informatia la resetare.

```
if (EEPROM.read(EE_START_ADDR) == KEY_DATA)
{
  savedTags = EEPROM.read(EE_START_ADDR + ADDRESS_TAGS);
} // Verify if EEPROM contain key data, clear the memory if not
else
{
  for (int i = 0; i < EEPROM.length(); i++)
  {
    EEPROM.update(i, 0x00);
```

```
  }
  for (int i = 0; i < 6; i++)
  { // Writing a default password when first time uploading
    EEPROM.update(i + 6, '1');
  }
}
EEPROM.update(EE_START_ADDR, KEY_DATA);
```

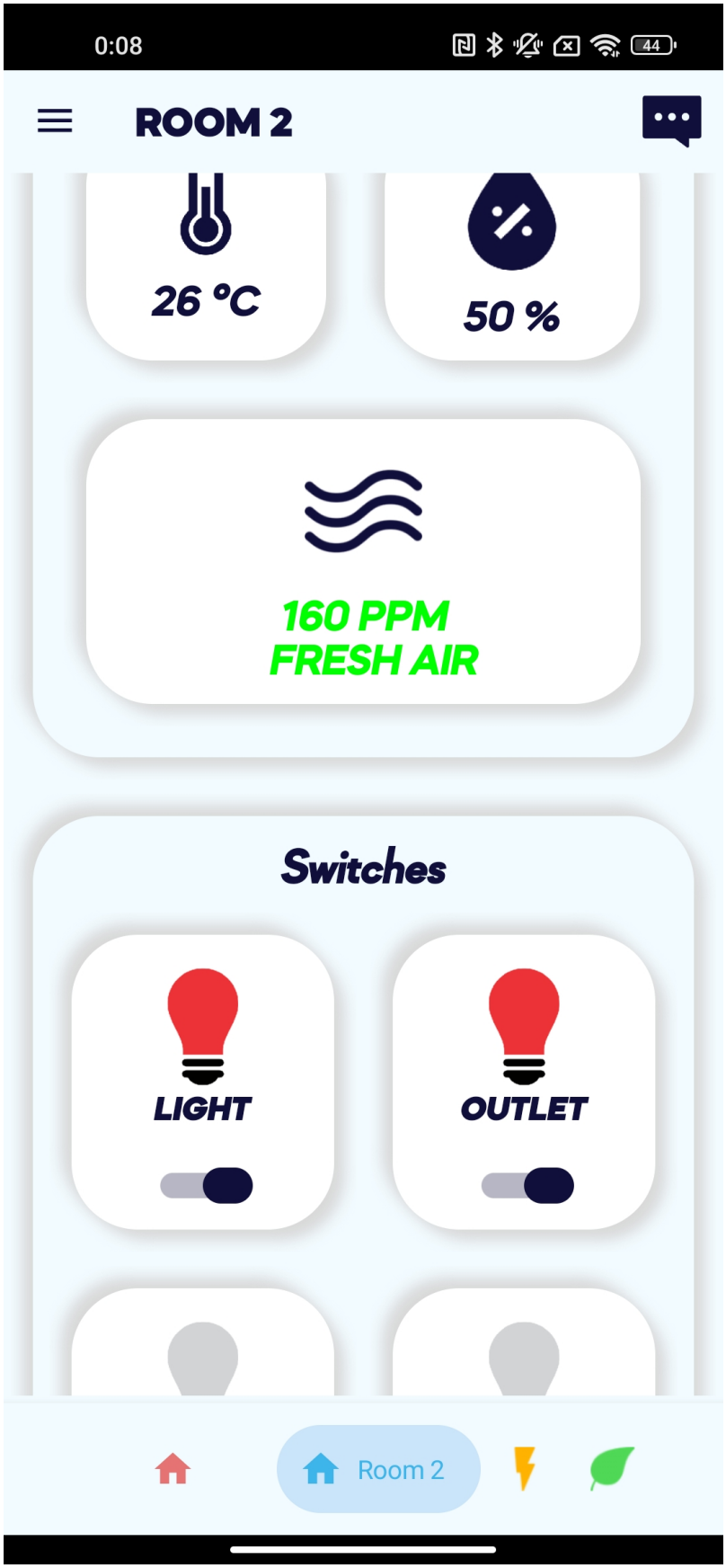
```
for (int i = 0; i < 6; i++)
{
  compPassword[i] = EEPROM.read(i + PASSWORD_ADDRESS); // Reading the
password from EEPROM
  Serial.print(compPassword[i]);
}
```

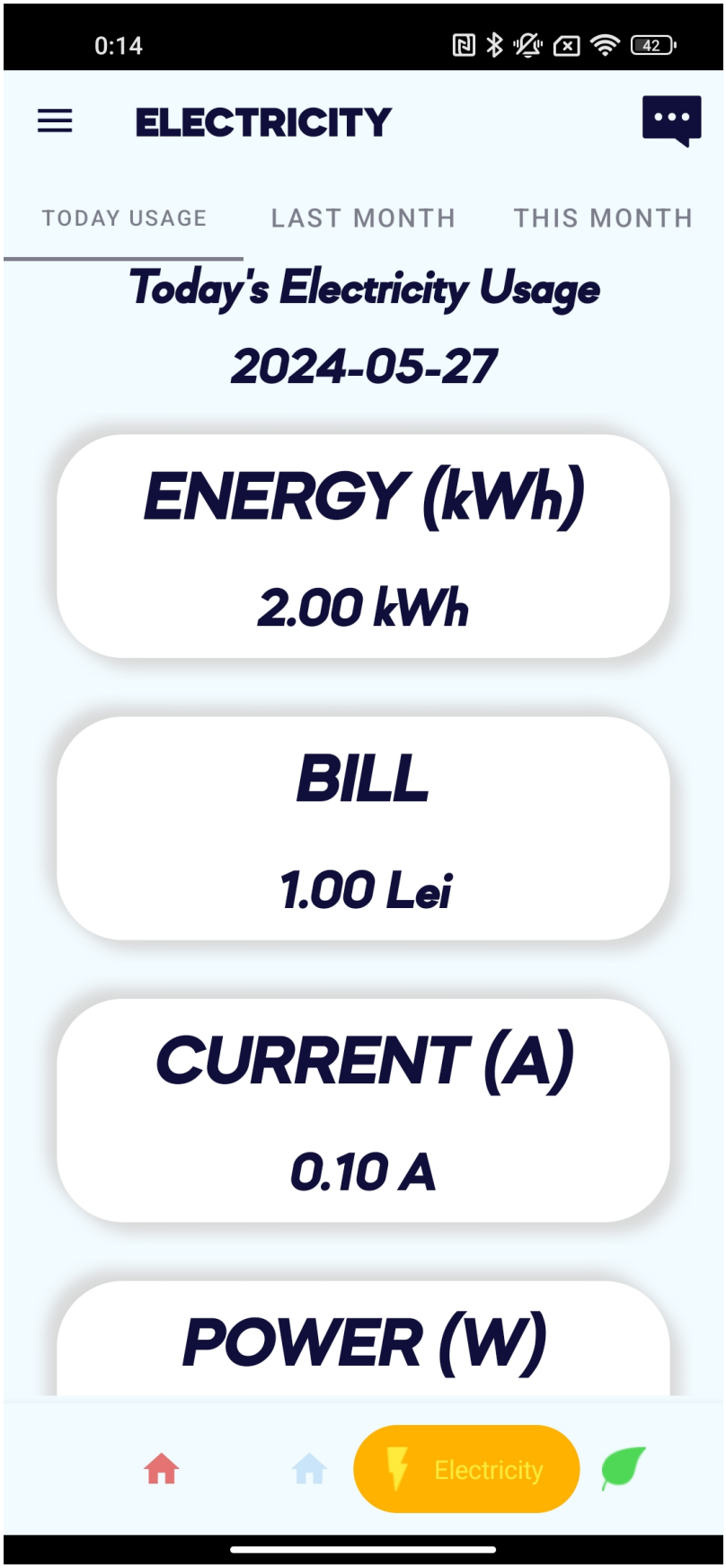
Stergere/Scriere cartele

```
// Adding and deleting new tag in/from EEPROM
bool saveOrDeleteTag(uint8_t *tag, uint8_t size)
{
   int16_t tagAddr = foundTag(tag, size);                      // search the
tag
   uint16_t newTagAddr = (savedTags * 8) + EE_START_ADDR + ADDRESS_TAGS; //
last tag addres
   if (tagAddr >= 0)
   { // deleting the tag
      for (uint8_t i = 0; i < 8; i++)
      {
        EEPROM.write(tagAddr + i, 0x00);                           //
deleting old tag
        EEPROM.write(tagAddr + i, EEPROM.read((newTagAddr - 8) + i)); //
writing new tag
        EEPROM.write((newTagAddr - 8) + i, 0x00);
      }
      EEPROM.write(EE_START_ADDR + 1, savedTags --); // write in EEPROM
number of tags
      return false;
    }
    else if (savedTags < MAX_TAGS)
    {
      for (uint16_t i = 0; i < size; i++)
      EEPROM.write(i + newTagAddr, tag[i]);       // write new tag
      EEPROM.write(EE_START_ADDR + 1, ++savedTags); // write number of tags
    }
    else
    {
      return false;
    }
```

```
    return true;
}
```

Poze aplicatie


0:08
ROOM 2
26 °C
50 %
160 PPM
FRESH AIR
Switches
LIGHT
OUTLET
Room 2



Rezultate Obţinute

Inchidere Usa:

```
  Introducere parola pe numpad pentru a deschide usa;
  Apropiere cartela pentru a deschide usa;
  Cand usa este deschisa tastare "***" pentru a intra in meniul de setari
  unde putem schimba parola, limba(RO, EN, FR), adauga/sterge cartele;
  Usa se incuie automat dupa cateva secunde daca este inchisa.
```

Cutie 1 si 2:

```
  Afisare temperatura, umiditate, calitate aer pe telefon;
  Activare/dezactivare becuri, prize, alarma de la telecomanda sau telefon;
  Aprinde/stinge luminile, batand de doua ori din palme;
  Aprinde lumina daca intensitatea luminii este mai mica de un prag;
  Activare/Dezactivare becuri, prize, alarma intrun anumit interval;
  Setare intensitate lumina de la telecomanda sau din aplicatie;
  Afisare curent consumat in aplicatie;
  Activare alarma daca cineva a patruns in casa sau daca calitatea aerului
este rea;
  Afisare status cutii utilizand ledurile montate.
```

Video 1 Video 2 Video 3

Download

dragos.coscodan.smart_home.zip

Bibliografie/Resurse

Resurse Software:

https://randomnerdtutorials.com/esp32-pwm-arduino-ide/

https://medium.com/@khantalha7367/uart-communication-on-esp32-28fd3df3b6eb

https://randomnerdtutorials.com/esp32-esp8266-firebase-authentication/

https://developer.android.com/

Resurse Hardware:

https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

https://www.instructables.com/AC-PWM-Dimmer-for-Arduino/

https://diyprojectslab.com/measure-ac-current-using-arduino-and-sct-013/

Export to PDF