Gym Attendance System

Autor: Dumitru Bianca-Ștefania
Grupa: 332CA
Email: bianca.dumitru0711@stud.acs.upb.ro

Introducere

Proiectul își propune **verificarea și monitorizarea participanților unei săli de sport**. Acesta are două funcționalități principale: verificarea validității abonamentului personal și contorizarea numarului de prezențe la sală.

Identificarea userilor se face pe baza unui **card personal**, iar sistemul comunică direct cu utilizatorul prin diverse mijloace (audio, vizual) în funcție de validitatea abonamentului. Există și posibilitatea **înregistrării** unui user nou, atunci când el își scanează pentru prima dată cardul.

La fiecare scanare a cardului unui utilizator cu abonament valid, acestuia i se adaugă câte o prezență. Numărul de prezențe al fiecărui user poate fi vizualizat în timp real în cadrul unui site web.

Ideea proiectului este inspirată din sistemele de același tip folosite în prezent de sălile din oraș.

Consider că proiectul aduce beneficii atât pentru operatorii de săli de sport, care automatizează o parte din muncă prin folosirea identificării pe bază de carduri, cât și userilor, cărora li se poate pune la dispoziție cu ușurință un rezumat al vizitelor la sală.

Elementul de noutate pe care l-am introdus, comparand sistemul cu cele pe care le-am intalnit in cadrul salilor de fitness, este afisajul numelui si al contorului de prezente odata cu scanarea cardului.

Descriere generală

Proiectul este controlat de către un modul ESP32 ce suportă comunicație prin Wi-Fi. Userul interacționează cu sistemul prin intermediul cardului de acces, ce este citit de către un modul RFID. Datele cardului sunt transmise ESP-ului, care, folosindu-se de modulul Wi-Fi, comunică cu un server web pentru a determina validitatea abonamentului userului. În funcție de răspunsul primit, modulul ESP acționează un buzzer, un led RGB și un display LCD pentru a-i transmite outputul userului.

Schema bloc a proiectului:

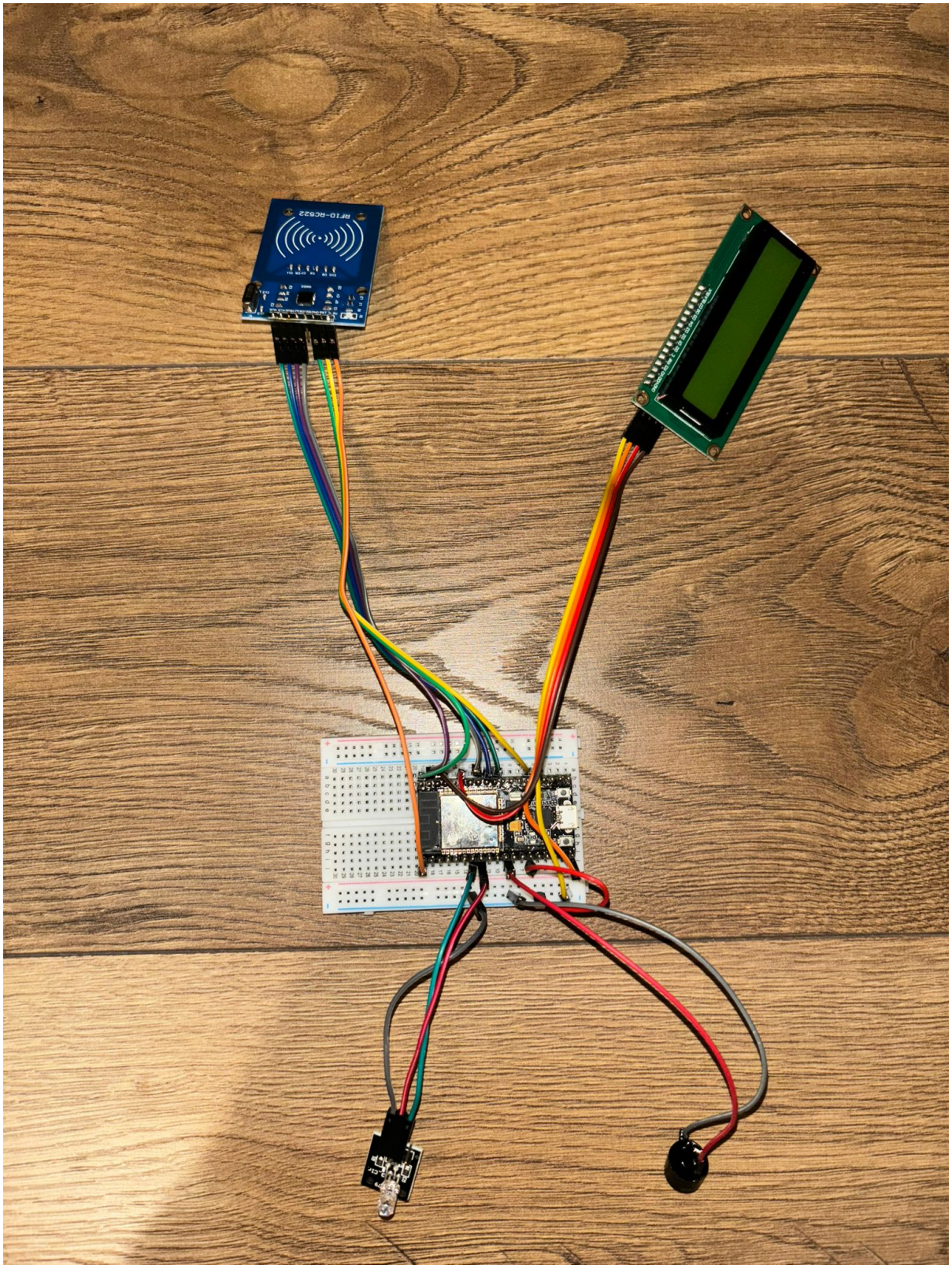
Hardware Design

Lista de componente:

- ESP32 WROOM cu 38 de pini, modul Bluetooth și Wi-Fi
- Led RGB
- LCD 16×2 cu modul I2C
- Modul RFID RC522
- Carduri de acces
- Buzzer
- Half-Breadboard
- Fire
- Sursă de alimentare (baterie externă)

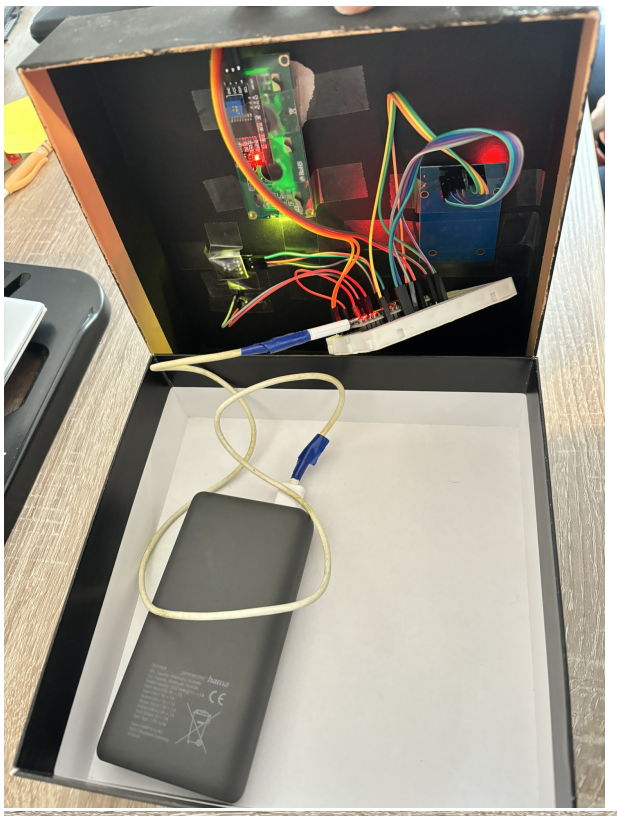
Schema electrică:

Asamblarea componentelor:



După asamblarea inițială a componentelor, le-am testat pe rând folosind câte un exemplu generic găsit în lista de exemple din Arduino IDE, pentru a verifica corectitudinea asamblării.

Pentru a semăna cât mai tare cu un produs real, care ar putea fi folosit într-o sală de fitness am realizat o “carcasă” pentru componente dintr-o cutie de carton, pe care am decupat-o. Am aranjat apoi componentele în carcasă și le-am lipit pentru a le fixa.



Software Design

Stadiul actual al implementării software: În momentul de față, proiectul este finalizat și complet funcțional, înglobând toate funcționalitățile pe care mi-am propus de la început să le aibă.

Repo GitHub: https://github.com/dumitrustefania/Gym-Attendance-System

Mediu de dezvoltare: Arduino IDE

Biblioteci utilizate: WiFi.h, HTTPClient.h, ArduinoJson.h (pentru comunicarea cu serverul web), LiquidCrystal_I2C.h (pentru LCD-ul cu modul I2C), MFRC522.h (pentru cititorul de carduri RFID)

Motivarea alegerii bibliotecilor folosite în cadrul proiectului: Bibliotecile aduc simplitate și mai multă siguranță în cadrul proiectului, facilitând integrarea elementelor hardware cu cele software.

Elementul de noutate al proiectului: Elementul de noutate pe care l-am introdus, comparând sistemul cu cele pe care le-am întâlnit în cadrul sălilor de fitness, este afișajul numelui și al contorului de prezențe odată cu scanarea cardului.

Utilizarea funcționalităților din laborator:

- Laboratorul 0: GPIO - Configurarea și utilizarea pinilor GPIO pentru LED-ul RGB și buzzer
- Laboratorul 1: UART - Comunicarea serială pentru debug
- Laboratorul 3: Timere. PWM - Utilizarea PWM pentru controlul buzzer-ului
- Laboratorul 5: SPI - Comunicarea cu cititorul RFID si modului WiFi
- Laboratorul 6: I2C - Comunicarea cu ecranul LCD

Calibrarea elementelor de senzoristică: Elementele de senzoristică au fost calibrate pentru a garanta o funcționare corectă și precisă a fiecărei componente. În cadrul fișierelor header prezente în arhivă se pot observa metodele de calibrare utilizate pentru RFID, LCD, buzzer și LED, prin inițializarea corectă și prin setarea stării acestora în funcție de outputul propus.

Optimizari:

- folosirea protocolului stateless HTTP in loc de o conexiune continua cu serverul
- oprierea functiei Bluetooth a modulului ESP pentru a economisi energie
- revenirea la starea initiala a LED-ului si LCD-ului doar dupa modificarea acestora datorata de citirea unui card, nu la fiecare rulare a functiei loop

Software design:

Codul este organizat în mai multe fișiere, fiind împărțit între **fișierele header** cu funcții helper pentru inițializare și setare de stări (http.h, lcd.h, rgb.h, buzzer.h, rfid.h) și fișierul **main**, ce conține logica principală a programului.

Funcția de setup se ocupă cu inițializarea componentelor și realizarea conexiunii Wi-Fi:

```
void setup() {
  Serial.begin(9600);

  // Dezactivează complet stiva Bluetooth pentru a economisi energie
  btStop();
  esp_bt_controller_disable();
  esp_bt_controller_deinit();

  connectToWiFi();
  initializeRFID(rfid);
  initializeRGB(PIN_RED, PIN_GREEN);
  initializeBuzzer(PIN_BUZZER);
  initializeLCD(lcd);
```

```
}
```

Funcția loop așteaptă detectarea unui card, moment în care se folosește de funcția sendCheckUserRequest pentru a interoga serverul referitor la cardul scanat. Pe baza răspunsului serverului, programul afișeaza diverse mesaje, revenind ulterior înapoi în starea de așteptare.

```
void loop() {
  // Reset the loop if no new card present on the sensor/reader
  if (!rfid.PICC_IsNewCardPresent())
    return;

  // Verify if the ID has been read
  if (!rfid.PICC_ReadCardSerial())
    return;

  String cardID = "";
  bool validCard = getCardID(rfid, cardID);
  if (!validCard)
    return;

  // Send request to server
  if (sendCheckUserRequest(cardID)) {
    // Delay to allow the user to read the output
    delay(3000);

    // Go back to default state
    noColorRGB(PIN_RED, PIN_GREEN);
    displayScanCard(lcd);
  }

  stopRFID(rfid);
}
```

Funcția sendCheckUserRequest se ocupă de interogarea serverului în legătură cu un card tocmai scanat, folosind un request de tip HTTP POST. Cele trei stări posibile în care se poate afla un card (user) sunt:

- valid - Userul este prezent în baza de date și are un abonament valid
- invalid - Userul este prezent în baza de date și are un abonament invalid
- not registered - Userul nu este încă înregistrat

În funcție de starea returnată, programul afișează userului mesaje sugestive:

- valid
  - LCD - Welcome {nume}! {no_attendances} attendaces
  - LED RGB - verde
  - Buzzer - sunet de confirmare
- invalid
  - LCD - Access denied
  - LED RGB - rosu
  - Buzzer - sunet de refuz

- not registered → Determină apariția unui prompt de înregistrare a userului în cadrul interfeței site-ului web. Programul realizează long polling până la finalizarea înregistrării, așteptând datele nou-introduse despre user (ce se poate afla fie în starea validă, fie cea invalidă). În timpul procesului de long polling, se afișează următoarele:
  - LCD - Not registered! Registering...
  - LED RGB - portocaliu

Funcția returneaza true daca toate requesturile necesare s-au finalizat cu succes.

```
bool sendCheckUserRequest(String cardID) {
  HTTPClient http;

  Serial.println("Sending POST request to server...");
  if (http.begin(serverAddress)) {
    int httpResponseCode = sendPOSTRequest(http, cardID, true);

    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.print("Response Code: ");
      Serial.println(httpResponseCode);
      Serial.print("Response: ");
      Serial.println(response);

      DynamicJsonDocument doc(200);
      deserializeJson(doc, response);

      String membershipStatus = doc["status"];
      if (membershipStatus == "valid") {
        String firstName = doc["first_name"];
        int attendances = doc["attendances"];
        validUser(firstName, attendances);
      } else if (membershipStatus == "invalid") {
        invalidUser();
      } else if (membershipStatus == "not registered") {
        notRegisteredUser();

        // Perform long polling until the user is registered from the UI
        while (membershipStatus == "not registered") {
          delay(3000); // Delay to avoid flooding the server with requests
          Serial.println("Performing long polling...");

          int httpResponseCode = sendPOSTRequest(http, cardID, false);

          if (httpResponseCode > 0) {
            String response = http.getString();
            Serial.print("Response Code: ");
            Serial.println(httpResponseCode);
            Serial.print("Response: ");
            Serial.println(response);

            DynamicJsonDocument doc(200);
```

```
            deserializeJson(doc, response);
            membershipStatus = doc["status"].as<String>();

            if (membershipStatus == "valid") {
              String firstName = doc["first_name"];
              int attendances = doc["attendances"];
              displayRegistered(lcd);
              validUser(firstName, attendances);
            } else if (membershipStatus == "invalid") {
              displayRegistered(lcd);
              invalidUser();
            }
          } else {
            Serial.println("Error on HTTP request");
          }
        }
      }
      http.end();
      return true;
    } else {
      Serial.println("Error on HTTP request");
      http.end();
      return false;
    }
  } else {
    Serial.println("Unable to connect to server");
    return false;
  }
}
```

Pentru a putea testa programul într-un mediu cât mai realist (folosirea programului de către operatorii unei săli de fitness), am realizat un server web minimal (folosind Flask pentru backend și JavaScript pentru frontent), pe care l-am hostat apoi pe Heroku.

Pagina principală afișează detaliile despre toți userii prezenți în baza de date:

Atunci când un card neînregistrat este citit de către modulul RFID, se afișează un prompt de înregistrare al noului user.

Rezultate Obţinute

Demo-ul exemplifică funcționalitatea proiectului prin scanarea unui card înregistrat cu abonament valid, a unui card înregistrat cu abonament invalid și prin înregistrarea unui card nou:

Concluzii

Realizarea proiectului a fost un proces interesant, fiind prima data cand am avut ocazia sa implementez un produs fizic, care include si hardware, nu doar software. Sunt multumita de ceea ce am realizat, si consider ca sistemul este unul complet, care ofera aceleasi capabilitati ca unul real, dintr-o sala de fitness. Singurul lucru la care consider ca ar mai trebui lucrat este aspectul, intrucat “carcasa” este improvizata de mine si nu se preteaza unui produs real.

Download

Arhiva ce contine codul: gym-attendance-system-main.zip

Jurnal

- 14 aprilie - cumparare piese
- 22 aprilie - lipire RFID
- 2 mai - începere documentație
- 7 mai - asamblare componente și testare individuală
- 11-15 mai - scriere și îmbunătățire cod
- 15 mai - realizare schemă electrică în Autodesk Fusion
- 16-17 mai - realizare carcasă și asamblarea componentelor în aceasta
- 22 mai - filmare demo
- 24 mai - finalizare documentație

Bibliografie/Resurse

Datasheet ESP32

Conectare RFID si ESP32

Requesturi HTTP cu ESP32

Export to PDF