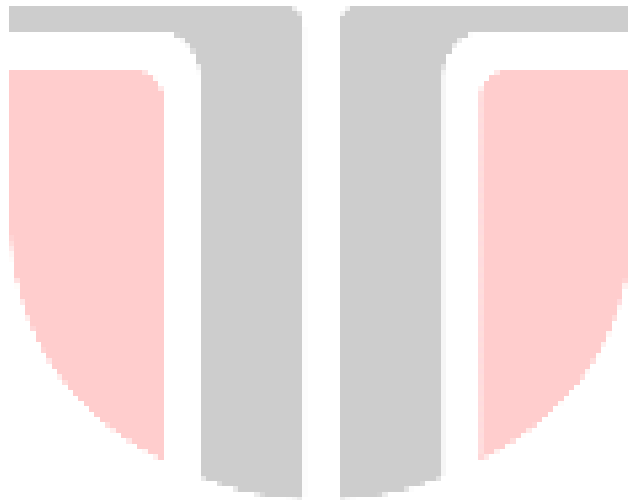


Carucior inteligent pentru cumparaturi.

~Documentatie~



TECHNICAL
UNIVERSITY
OF CLUJ-NAPOCA
ROMANIA

Proiectarea si implementarea unui carucior inteligent
folosind tehnologia RFID si placuta Arduino Mega 2560

Universitatea Tehnica Cluj-Napoca

Facultatea de Calculatoare si Tehnologia Informatiei

Anul III, Seria A, grupa 30231

Ştef Paula-Elena

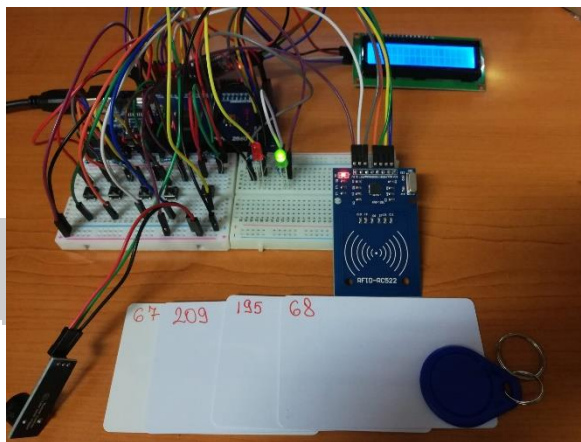
1. INTRODUCERE

1.1. Obiectiv

Implementarea acestui proiect s-a realizat pe baza tehnologiei RFID, prezentata prin suportul placutei Arduino si a altor componente/module necesare.

Ideea de baza a acestui proiect este aceea de a proiecta, si apoi implementa, un carucior inteligent, care sa usureze “scanarea” produselor si evitarea cozilor lungi din magazine. Acest cos de cumparaturi inteligent se va remarca prin marcarea produselor cumparate, vizualizarea informatiilor legate de produse, afisarea produselor si preturilor, cresterea/scaderea numarului de produse si “plata”/generarea facturii pentru obiectele selectate.

Acest cos de cumparaturi poate fi folosit atat de supermarketuri cat si de magazinele alimentare mai mici, usurand metoda de scanare si plata, totodata evitand cozile lungi si astfel micșorand si numărul de personal necesar.



1.2. Ideea de baza

Astfel, ideea de la care am plecat in realizarea proiectului, este aceea de a face posibila “scanarea” produselor in momentul adaugarii lor in cos. Pentru a face acest lucru, proiectul va fi implementat folosind o placuta Arduino si a tehnologiei RFID(Radio frequency identification): RFID cards, tags si RFID readers. Acestea se bazeaza pe o tehnologie unde datele digitale sunt codificate în etichete RFID și decodate de un cititor RFID folosind unde radio.

1.3. Descriere

Atunci cand consumatorul doreste sa adauge un produs in cos, RFID reader il va detecta datorita tag-ului specific produsului. Produsul va fi apoi afisat pe LCD alaturi de pret si va fi adaugat la lista de cumparaturi. Pentru a asigura cumparatorul ca produsele au fost scanate vom adauga un buzzer care va suna atunci cand sunt scanate anumite produse si pentru a evita erori (de ex. adaugare de doua ori a unui produs etc.). Pe masura ce cumparatorul adauga produse acestea vor fi adaugate la pretul final. De asemenea, cumparatorul poate creste/scade numarul de produse pe care doreste sa il adauge la cos. La final, cu ajutorul unui buton se va afisa pretul final de plata si de asemenea se va genera o "factura" in care se specifica toate produsele finale cumparate si cantitatea lor. De asemenea putem adauga optiunea de a revedea produsele adaugate si pretul lor folosindu-ne de butoane.

Pentru a putea exemplifica fizic functionalitatea tehnologiei RFID vom implementa acest carucior utilizand o placuta Arduino pe care o vom conecta la un modul RFID reader. Folosind RFID reader, tags si cards vom crea schema si o vom implementa astfel incat sa ne putem conecta de asemenea si la un laptop (care ar reprezenta server-ul unde se stocheaza baza de date). Pe masura ce cumparatorii vor adauga produse si apoi achizitiona, acestea vor fi trimise la laptop si vor fi scazute din baza de date. De asemenea ne vom folosi de acest lucru si pentru a anunta angajatorul atunci cand se termina unele produse si este nevoie sa se comande altele si de asemenea pentru a pastra evidenta tuturor produselor cumparate si preturilor si pentru a realiza inventarul mai usor.

2. Tehnologia RFID

RFID este un acronim pentru „identificare prin radiofrecvență” și se referă la o tehnologie prin care datele digitale codificate în etichete RFID sau etichete inteligente sunt capturate de un cititor prin unde radio. RFID este similar cu codurile de bare in ideea ca, de asemenea, datele dintr-un tag sau card sunt capturate de un dispozitiv care stochează datele într-o bază de date. RFID, totuși, are mai multe avantaje față de sistemele care utilizează software de urmărire a activelor cu coduri de bare.

RFID aparține unui grup de tehnologii denumite “Identificare automată și captare a datelor” (AIDC). Metodele AIDC identifică automat obiectele, colectează date despre ele și introduc aceste date direct în sistemele informatice cu intervenție umană mică sau deloc. La nivelul tehnologiei RFID, acest lucru se realizează prin unde radio. La un nivel simplu, sistemele RFID constau din trei componente: o etichetă RFID sau etichetă inteligentă, un cititor RFID și o antenă. Etichetele RFID conțin un circuit integrat și o antenă, care sunt folosite pentru a transmite date către cititorul RFID (numit și interogator). Dacă ar fi să analizăm puțin mai în detaliu componentele putem observa că cititorul RFID are nevoie de o antenă construită sub forma unei bobine, un emițător-receptor radio specializat pe standardul RFIF și un circuit “inteligent” pentru interpretarea mesajelor și transmiterea acestora pe cale serială la un

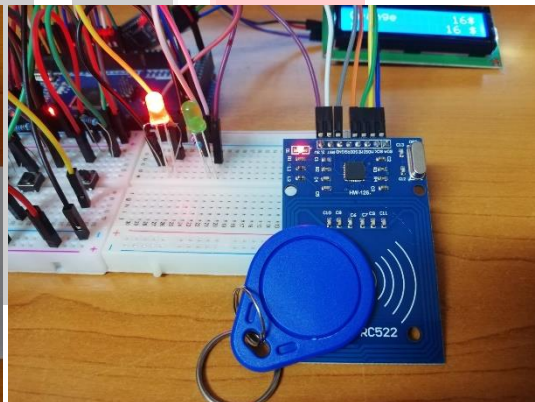
calculator(microcontrolor). Pentru realizarea tag-urilor sau card-urilor RFID este nevoie de un circuit integrat si o bobina pe post de antenă.

Etichetele transmit astfel datele catre citator (numit si interogator), iar acesta convertește aundele radio într-o formă mai utilizabilă de date. Informațiile colectate de la etichete sunt apoi transferate printr-o interfață de comunicații către un sistem computer gazdă, unde datele pot fi stocate într-o bază de date și analizate ulterior. Astfel, in principiu, pentru a se putea realiza un citator RFID este nevoie de un microcontroller (unde se vor pastra/manipula datele) si un circuit de citire/scriere care se leaga serial.

Pentru implementarea acestui proiect am decis sa folosim o placuta Arduino pe post de microcontroller pentru a manipula datele si pentru a le transmite la un server(calculator) si am folosit un RFID module pentru a scana diferitele produse (reprezentate de tags si cards).



RFID TAGS and RFID cards



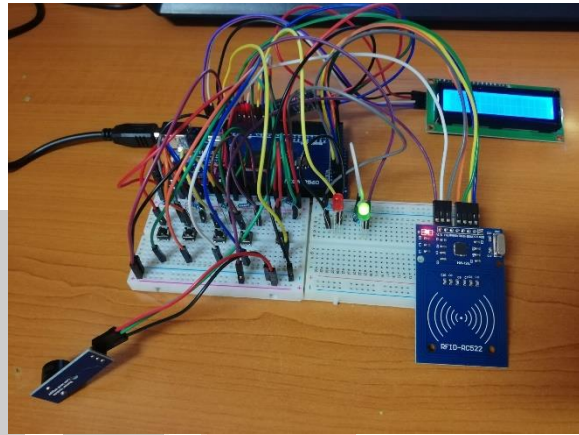
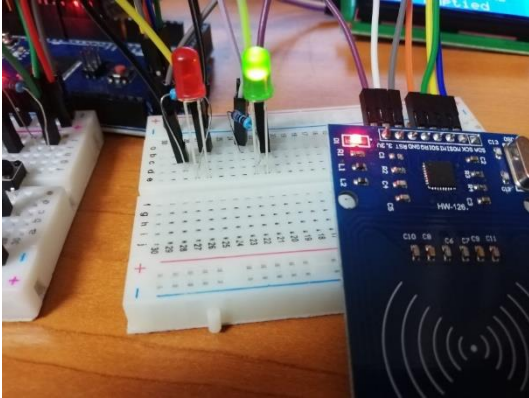
RFID reader

3. Functionalitati

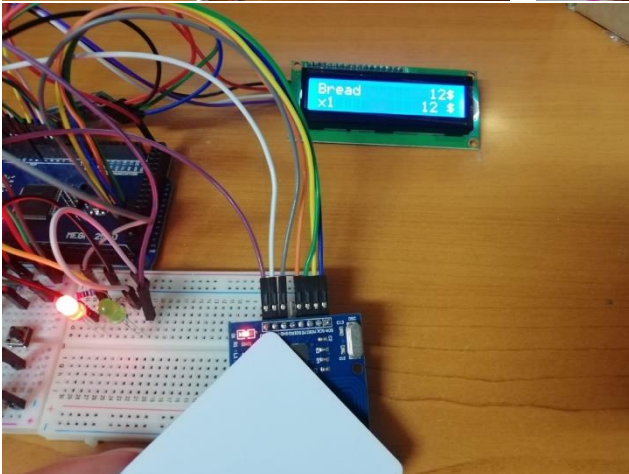
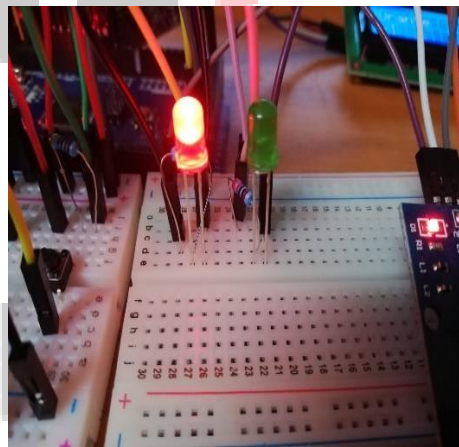
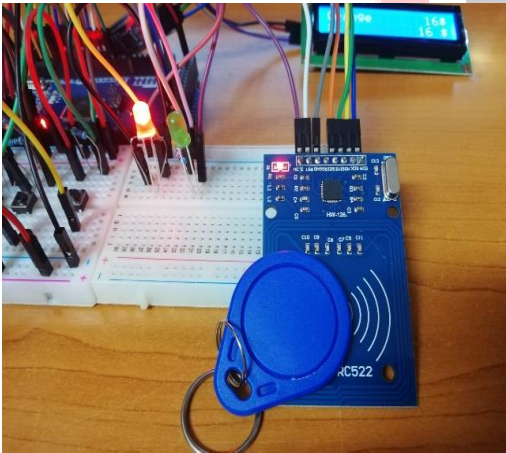
In vederea implementarii proiectului am urmarit realizarea urmatoarelor functionalitati:

- Interogare pret si date despre produs
- Aduagare cantitate
- Stergere cantitate
- Adaugare produs la cosul de cumparaturi
- Afisare produse din cosul de cumparaturi
- Posibila renuntare la cosul de cumparaturi
- Check-out / Realizarea platii (aici se vor trimite datele si la server)

In momentul alimentarii componentelor ne vom afla in starea initiala a proiectului : nu vom avea niciun produs in cos, LCD-ul nu va afisa nimic, iar led-ul verde este aprins pentru a specifica faptul ca cosul de cumparaturi este disponibil pentru a fi utilizat.



In momentul in care se va scana primul produs – lucru care se realizeaza prin apropierea acestuia de RFID reader-, led-ul verde se va stinge si cel rosu se va aprinde pentru a specifica ca acesta a fost ocupat.



NAPOCA
NIA

Se va aduga primul produs in cos si se vor afisa datele pe LCD: numele produsului, pretul acestuia, cantitatea de produse, si pretul final (in functie de cantitatea dorita).



Pentru a creste cantitatea de produse adaugate avem butonul 1 (cel din stanga).



Pentru a scadea cantitatea de produse se va apasa butonul 2. Daca nu dorim sa mai adugam acest produs se va scadea cantitatea la 0 si acesta nu va fi adaugat.



Dupa ce am selectat cantitatea deorita se va apasa butonul 3 pentru a adauga produsul in cos.In acest punct pe LCD se va afisa produsul adugat si cantitatea, iar sub se va afisa pretul total actual.



Pentru a vedea lista de cumparaturi se va afisa butonul 4. Astfel , pe LCD vor fi scrise, pe rand, cate doua produse pentru 2 secunda, dupa care se trece la urmatoarele.



La final se va afisa pretul final si se va anunta utilizatorul ca daca doreste sa plateasca trebuie sa apese din nou butonul.



Pentru o a doua apasare com afisa un mesaj pentru a ne asigura ca utilizatorul doreste sa faca plata. In caz afirmativ, este nevoie sa se apese din nou pe buton.



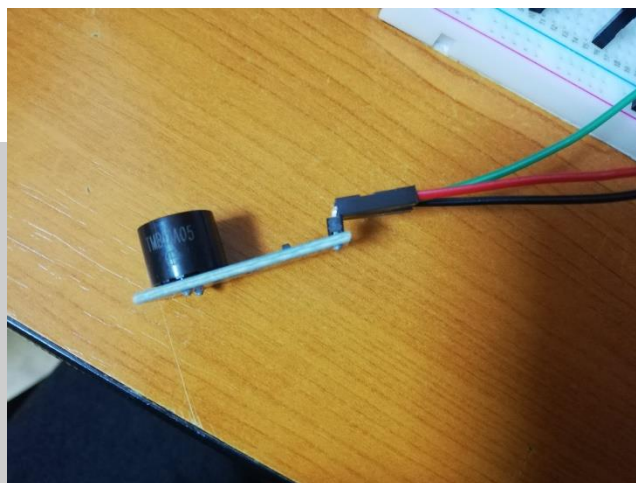
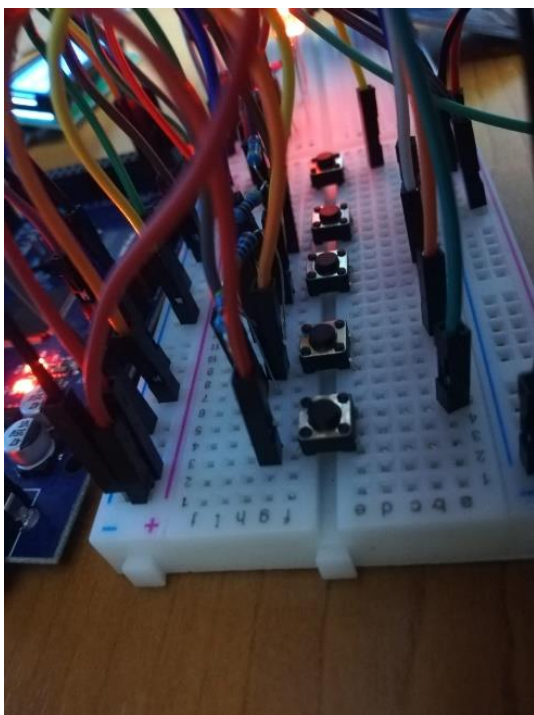
Pentru o a treia apasare se va realiza plata produselor, se va aprinde led-ul verde si se va stinge led-ul rosu pentru a specifica ca cosul este liber. Lista de cumparaturi este trimisa catre server pentru a putea fi scazute obiectele respective din baza de date.



Prin apasarea butonului 5 in orice moment se va renunta complet la toate cumparaturile si se va elibera cosul.



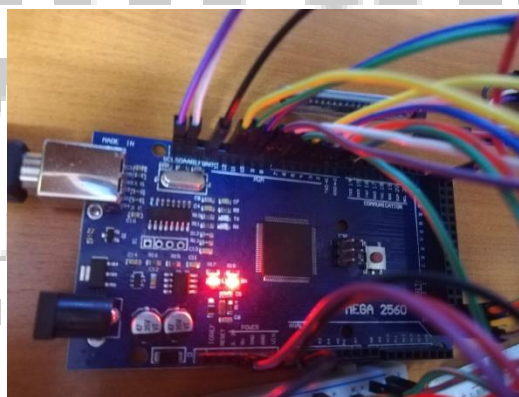
Pentru fiecare apasare de buton, modulul buzzer va emite un sunet scurt pentru a instiinta cumparatorul ca butonul a fost apasat.



4. Proiectare

Pentru inceput voi enunta componentele folosite in realizarea cosului de cumparaturi:

- Placuta ARDUINO MEGA 2560



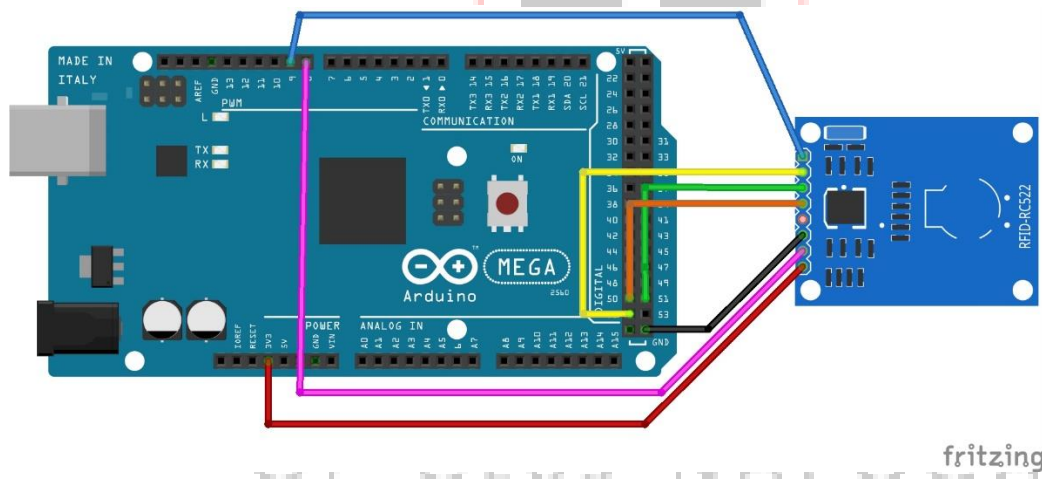
- Modul RFID RC522



- Modul buzzer
- 5 butoane
- 5 rezistente de 4.7 kOhm
- 2 led-uri
- 2 rezistente de 220 kOhm
- Modul LCD I2C
- Fire de legatura

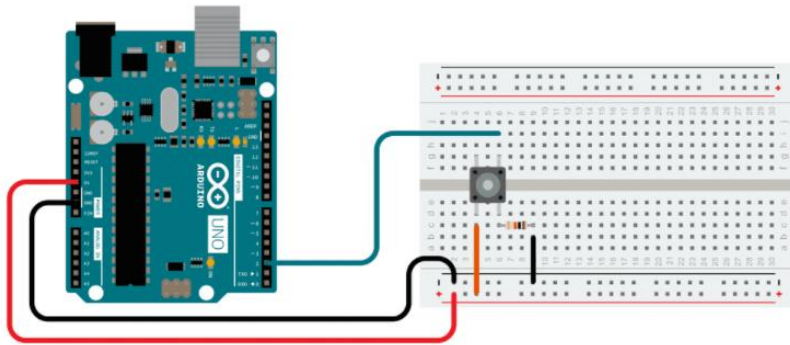
Apoi, pentru inceput am realizat pe rand legarea componentelor la placuta Arduino.

In prima faza am legat modulul RFID ca in schema urmatoare:



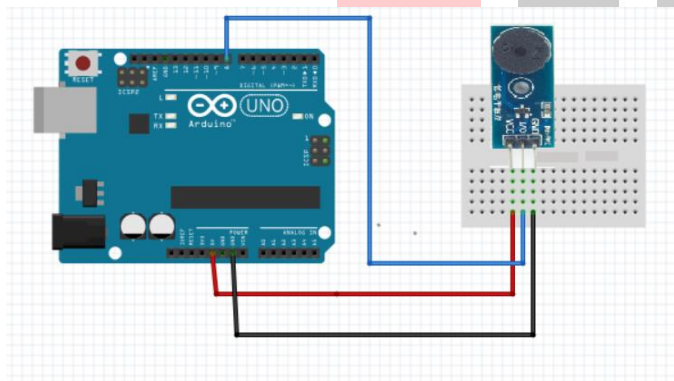
Apoi, am continuat prin legarea butoanelor prin intermediul carora se vor face operatiile principale. Am considerat ca sunt necesare 5 butoane pentru functionalitatile alese. Am folosit modul de conectare

“PULL-DOWN”, in care am folosit rezistente de 4.7 kOhm:



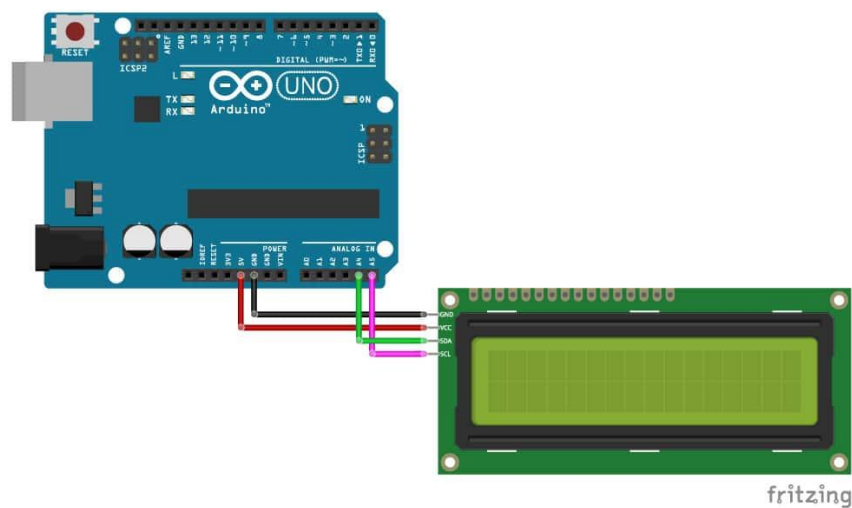
Am repetat acest mod de conectare pentru toate cele 5 butoane.

Am continuat prin a lega un modul buzzer:

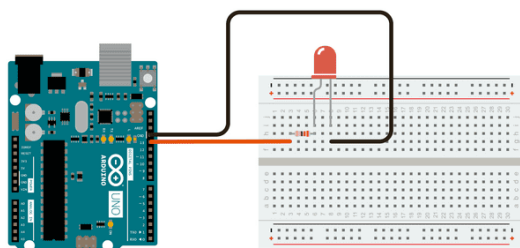


Am adaugat apoi un modul LCD pentru a afisa datele necesare:

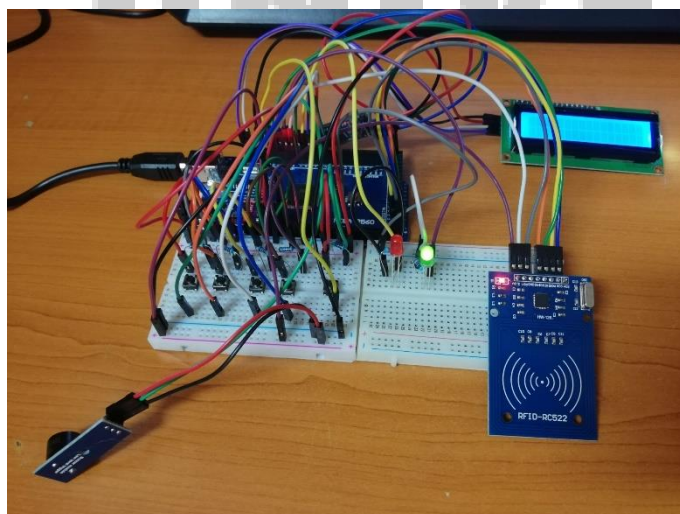
UNIVERSITY
OF CLUJ-NAPOCA
ROMANIA



Apoi, pentru afisare am adaugat 2 led-uri (de culoare rosie si de culoare verde) urmand schema prezentata mai jos si folosind rezistente de 220R:



Astfel, in final dupace am adaugat toate componentele necesare am obtinut urmatoarul amplasament :



5. Implementare

Dupa ce am realizat toate conexiunile necesare, am inceput programarea fiecarei componente pentru a realiza proiectul dorit. Pentru acest lucru am folosit IDE-ul pus la dispozitie de Arduino si anume: **Arduino Software (IDE)**. Pentru a face posibila citirea datelor primite de la RFID si prelucrarea lor am utilizat biblioteca <RFID.h> si de asemenea pentru utilizarea modulului LCD am avut nevoie de biblioteca <LiquidCrystal_I2C.h> si de asemena am folosit biblioteca standard Arduino <SPI.h>. Dupa ce am inclus aceste biblioteci am inceput sa facem setarile necesare. Pentru incaput am declarant toate componentele necesare, am specificat pinii de intrare pentru butoane, pinii de iesire pentru led-uri si buzzer, am declarant sirurile in care vom pastra produsele,pretul lor si Id-ul tag-urilor si toate variabilele necesare.

```
* Include the standard Arduino SPI library */
#include <SPI.h>
#include <SoftwareSerial.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27,16,2);
SoftwareSerial BluetoothSerial(BluetoothRx, BluetoothTx);

/* Define the DIO used for the SDA (SS) and RST (reset) pins. */
// pini pentru RFID reader
#define SDA_DIO 53
#define RESET_DIO 8

// pini pentru modulul bluetooth
#define BluetoothTx 11
#define BluetoothRx 10

// sirul in care am pastrat ID-ul fiecarui card
int productID[5]={209,68,67,195,12};

//un sir in care am specificat pretul fiecarui produs
int productPrice[5]={14,12,15,12,16};

//un sir in care am assignat un nume de produs fiecarui ID
String names[] = {"Apple", "Bread", "Water", "Juice", "Orange"};

//un sir in care pastram fiecare cantitate
int quantity[5]={0,0,0,0,0};

//Pretul final pentru cumparaturi
int Total;
int k = 0;
```

```

//un sir in care vom pune produsele adaugate in cosul de cumparaturi
String addedProducts[5];

//Am creat o instanta a libreriei
RFID RC522(SDA_DIO, RESET_DIO);

//Pini folositi pentru butoane
const int buttonPin1 = 2;
const int buttonPin2 = 3;
const int buttonPin3 = 4;
const int buttonPin4 = 5;
const int buttonPin5 = 12;

//pini folositi pentru led-uri
int LED = 6;
int LED2 = 9;

//pinul pentru buzzer
const int buzzerPin = 7;

// variabile care imi vor specifica starea fiecarui buton - HIGH cand e apasat si LOW cand nu
int buttonState1 = 0;
int buttonState2 = 0;
int buttonState3 = 0;
int buttonState4 = 0;
int buttonState5 = 0;

//o variabila prin care am numarat de cate ori este apasat butonul 4
int checkState;

```

Am continuat apoi prin a initializa valori si a citi stari in functia de setup, aceasta se realizeaza o singura data, in momentul in care se alimenteaza componentele, asa ca aici am decis sa definim care pini sunt de intrare si iesire

```

void setup()
{
    Serial.begin(9600);
    lcd.init();
    lcd.clear();
    lcd.backlight();
    // am initializat pinii de intrare
    pinMode(buttonPin1, INPUT);
    pinMode(buttonPin2, INPUT);
    pinMode(buttonPin3, INPUT);
    pinMode(buttonPin4, INPUT);
    pinMode(buttonPin5, INPUT);
    //am initializat pinii de iesire pentru led-uri
    pinMode(LED, OUTPUT);
    pinMode(LED2, OUTPUT);
    // am setat valorile initiale pentru led-uri
    digitalWrite(LED, LOW); //RED
    digitalWrite(LED2, HIGH); //GREEN
    //am setat pinul pentru buzzer ca pin de iesire
    pinMode(buzzerPin, OUTPUT);
    BluetoothSerial.begin(9600);
    /* Am pornit interfata SPI*/
    SPI.begin();
    /* Am initializat RFID-ul */
    RC522.init();
    //am initializat numarul de apasari ale butonului4
    checkState = 0;
}

```

Am continuat prin scrierea efectiva a programului, adica a functiei loop care se executa constant cat timp componentele sunt alimentate.

In aceasta functie verificam constant daca este citit un RFID card sau Tag si se verifica daca acel ID se afla printre cele din memorie (adica daca produsul scanat apartine magazinului). In caz afirmativ, led-ul verde se va stinge iar cel rosu se va aprinde, lucru care specifica ca cosul de cumparaturi este ocupat de catre un comparator. De asemenea, buzzer-ul va suna pentru 1 secunda pentru a atentiona cumparatorul ca produsul a fost scanat.

Pe LCD vom afisa pe prima linie numele produsului si pretul unui produs de acest tip, iar pe a doua linie vom afisa cantitatea de produse si pretul final al acestor produse. Acest lucru se realizeaza prin intermediul functiei printItem().

De asemenea se vor verifica constant stările butoanelor prin intermediul functiilor separate pentru fiecare, astfel incat se apasa un buton se vor realiza implementarile necesare.

Prin intermediul variabilei stay ne asiguram ca daca un produs este scanat, este nevoie ca acesta sa fie adaugat in cos inainte de a continua cu scanarea urmatoarelor obiecte, pentru a evita cantitati eronate pentru alte produse.

Daca totusi produsul nu se va gasi, vom afisa pe LCD un mesaj adecvat.

```
void loop()
{
    //citim stările butoanelor 4 si 5
    buttonState4 = digitalRead(buttonPin4);
    buttonState5 = digitalRead(buttonPin5);
    checkGiveUp();
    checkList();
    analogWrite(buzzerPin, 255);
    //digitalWrite(LED, LOW);
    //digitalWrite(LED2, HIGH);

    byte i=0;
    byte j=0;
    int ID;

    //Aici verificam daca a fost citit un card
    if (RC522.isCard())
    {
        buttonState4 = digitalRead(buttonPin4);
        boolean stay = true;
        analogWrite(buzzerPin, 0);
        digitalWrite(LED, HIGH);
        digitalWrite(LED2, LOW);

        /* Daca da, ii preluam ID-ul */
        RC522.readCardSerial();

        /* Verificam daca avem ID-ul */
        ID=RC522.serNum[0];
        for(i=0;i<5;i++)
        {
            buttonState4 = digitalRead(buttonPin4);
            if(productID[i]==ID)
            {
                printItem(i);
```

NICAL
ERSITY
- NAPOCA
ANIA

```

{
    printItem(i);
    if(quantity[i]==0)
    {
        quantity[i]++;
        printItem(i);
    }
    while(stay == true)
    {
        buttonState1 = digitalRead(buttonPin1);
        buttonState2 = digitalRead(buttonPin2);
        buttonState3 = digitalRead(buttonPin3);

        stay = checkButton(buttonState1, 1, i, stay);
        delay(100);
        stay = checkButton(buttonState2, 2, i, stay);
        delay(200);
        stay = checkButton(buttonState3, 3, i, stay);
        delay(100);

    }
    break;
}
else if(i==5)
{
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Product not found");
    break;
}
}
//checkList();
}
else{
    //Serial.print("Reading...");

```

Avem apoi functiile necesare pentru:

- Afisarea produselor pe LCD cand acestea sunt scanate:

```

void printItem(int pos){
    //firstline
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print(names[pos]);
    lcd.setCursor(13,0);
    lcd.print(productPrice[pos]);
    lcd.setCursor(15,0);
    lcd.print("$");

    //secondline
    lcd.setCursor(0,1);
    lcd.print("x");
    lcd.setCursor(1,1);
    lcd.print(quantity[pos]);
    lcd.setCursor(12,1);
    int total = quantity[pos] * productPrice[pos];
    lcd.print(total);
    lcd.setCursor(15,1);
    lcd.print("$");
}

```

- Pentru apasarea butonului 5 – aici se va afisa un mesaj pentru a specifica sa se renunta la cumparaturi si se vor sterge toate datele retinute pentru cumparatorul actual.

```

void checkGiveUp(){
    if( digitalRead(buttonPin5) == HIGH){
        analogWrite(buzzerPin ,0);
        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print("Drooping items");
        lcd.setCursor(0,1);
        lcd.print("List Emptied");
        delay(1000);
        for(int j=0; j<5; j++){
            addedProducts[j] = "";
            quantity[j] = 0;
        }
        Total = 0;
        k=0;
        digitalWrite(LED, LOW);
        digitalWrite(LED2, HIGH);
        //lcd.clear();
        checkState = 0;
    }else{
        analogWrite(buzzerPin,255);
    }
}

```

- Functia pentru butonul 4 unde in functie de cate ori acesta a fost apasat se vor realiza alte comenzi : daca a fost apasat o data se va afisa lista de cumparaturi si pretul total, de doua ori va apareea un mesaj de avertizare pentru cumparator ca urmeaza sa plateasca, iar a treia oara se va realiza plata – se vor sterge valorile pastrate pana in momentul actual pentru a pregati cosul pentru noul comparator si vom afisa produsele platite pe Serial Monitor pentru a putea fi trimise apoi la server si a fi scoase din baza de date.

```

void checkList(){
    if ( digitalRead(buttonPin4) == HIGH){
        if(checkState == 0){
            for(int j =0; j<k; j+=2){
                lcd.clear();
                lcd.setCursor(0,0);
                lcd.print(addedProducts[j]);
                lcd.setCursor(0,1);
                lcd.print(addedProducts[j+1]);
                delay(2000);
            }
            lcd.clear();
            lcd.print("Total: " + String(Total) + "$");
            delay(2000);
            lcd.clear();
            lcd.setCursor(0,0);
            lcd.print("Press again for");
            lcd.setCursor(0,1);
            lcd.print("checkout");
            checkState = 1;
        }
        else if(checkState == 1){
            lcd.clear();
            lcd.setCursor(0,0);
            lcd.print("Your total is: ");
            lcd.setCursor(0,1);
            lcd.print(String(Total) + "$");
            delay(2000);
            lcd.clear();
            lcd.setCursor(0,0);
            lcd.print("Press again to");
            lcd.setCursor(0,1);
            lcd.print("pay items");
            checkState = 2;
        }
    }
}

```



```

        checkState = 2;
    }
    else if(checkState == 2){
        analogWrite(buzzerPin ,0);
        lcd.clear();
        int paid = Total;
        lcd.print("You paid " + String(Total) + "$");
        delay(2000);
        for(int j=0; j<k; j++){
            Serial.println(addedProducts[j]);
        }
        for(int j=0; j<5; j++){
            addedProducts[j] = "";
            quantity[j] = 0;
        }
        Total = 0;
        k=0;
        digitalWrite(LED, LOW);
        digitalWrite(LED2, HIGH);
        //lcd.clear();
        checkState = 0;
    }
}
else {
    analogWrite(buzzerPin,255);
}
}
}

```

- apoi avem o functie pentru restul butoanelor – 1, 2 si 3. Am ales sa folosesc o singura functie pentru aceste butoane pentru a evita erori din cauza apasarii butoanelor in acelasi timp si deoarece in momentul cand se scaneaza un produs aceste 3 butoane se vor asigura ca se pot introduce in cos cantitatile corespunzatoare: - butonul 1 creste cantitatea, butonul 2 o scade, iar butonul 3 va aduga in cos produsul. Butonul 4 este blocat pana cand se va apasa butonul 3. De asemenea, daca dorim sa modificam cantitatea la produse mai tarziu vom putea scana produsul din nou si modifica cantitatea sau chiar sa eliminam produsul.

```

boolean checkButton( int buttonState, int buttonNumber, int pos, boolean state){
    if (buttonState == HIGH) {
        analogWrite(buzzerPin ,0);
        //Button1
        if(buttonNumber == 1){
            //Crestem cantitatea
            quantity[pos]++;
            printItem(pos);
        }
        //Button2
        if(buttonNumber == 2){
            //Scadem cantitatea
            quantity[pos]--;
            printItem(pos);
        }
        //Button3
        if(buttonNumber == 3){
            //Se salveaza modificarile si se trece la urmatorul buton
            Total = 0;
            for(int j=0; j<5; j++){
                if(quantity[j] != 0)
                    Total += quantity[j] * productPrice[j];
            }
            lcd.clear();

            //first line
            lcd.setCursor(0,0);
            String message1 = String(quantity[pos]);
            message1 += " x " + names[pos] + " added";
            lcd.print(message1);

            //second line
            lcd.setCursor(0,1);
            String message2 = "Total is: " + String(Total) + "$";

```

```

lcd.setCursor(0,1);
String message2 = "Total is: " + String(Total) + "$";
lcd.print(message2);
//Add item to cart
if(quantity[pos] != 0){
    int flag = 0;
    for(int j=0; j<k; j++){
        if(addedProducts[j].startsWith(names[pos])){
            addedProducts[j] = names[pos] + " x" + String(quantity[pos]);
            flag = 1;
            break;
        }
    }
    if(flag == 0){
        addedProducts[k] = names[pos] + " x" + String(quantity[pos]);
        k++;
    }
}
else{ //if quantity is 0 we want to delete the item
    for(int j=0; j<k; j++){
        if(addedProducts[j].startsWith(names[pos]))
        {
            for(int m=j; m<k; m++)
                addedProducts[m] = addedProducts[m+1];
            k--;
            break;
        }
    }
    state = false;
}
}
else {
    analogWrite(buzzerPin,255);
}
return state;
}

```

6. Conectarea la server

Pentru realizarea acestei functionalitati am creat o noua baza de date relationara in PostgreSQL in care am adaugat datele necesare, denumirea fiecarui produs, pretul acestuia, si cantitatea fiecarui produs. Pentru a realiza conexiunea propriu-zisa intre componente/aplicatia Arduino si aceasta baza de date create, am folosit o aplicatie Java implementata in IntelliJ. In aceasta aplicatie am realizat ata conexiunea cu baza de date cat si conexiunea cu aplicatia Arduino pentru a putea trimite produsele cumparate si cantitatea care s-a achizitionat pentru fiecare produs catre baza de date. Apoi am sczut din baza de date fiecare produs. Pentru a usura observarea bazei de date de catre administratorul magazinului am implementat de asemenea si o interfata prin care dispunem intr-o fereastra toate produsele existente in baza de date si detaliile legate de acest produs.

6.1. Conexiuni

Pentru conexiunea la baza de date a fost nevoie de PostgreSQL JDBC driver, iar pentru a realiza conexiunea intre o aplicatie Java si aplicatia Arduino am avut nevoie sa descarc biblioteca JSerialComm care ne permite citirea unui port serial.

Dupa ce am adaugat acestea, am inceput scrierea codului efectiv. In aplicatia de Java am realizat doua clase, una pentru conexiunea cu aplicatia Arduino si una pentru conexiunea cu baza de date.

```

public void connect() {
    //first connect the Database
    this.dbConnection = new DbConnection();
    dbConnection.connect();
    //Get the wanted port;
    this.ports = SerialPort.getCommPorts();

    System.out.println("Select a port:");
    int i = 1;
    for (SerialPort portI : ports) {
        System.out.println(i + " " + portI.getSystemPortName());
        i++;
    }
    Scanner s = new Scanner(System.in);
    int chosenPort = s.nextInt();
    this.port = ports[chosenPort - 1];
    //open the right port
    if (this.port.openPort()) {
        System.out.println("Port opened.");
    } else {
        System.out.println("Unable to open the port");
    }
    //wait for data
    this.port.setComPortTimeouts(SerialPort.TIMEOUT_READ_SEMI_BLOCKING, newReadTimeout: 0, newWriteTimeout: 0);
}

```

Conexiunea cu serial port. Am implementat si o metoda prin care vom primi datele si le vom prelucra si pastra intr-o lista de produse

```

public Connection connect() {
    Connection connection = null;
    {
        try {
            connection = DriverManager.getConnection(jdbcURL, username, password);
            System.out.println("Connected to PostgreSQL server.");
        } catch (SQLException throwables) {
            System.out.println("Error in connecting to PostgreSQL server.");
            throwables.printStackTrace();
        }
    }
    return connection;
}

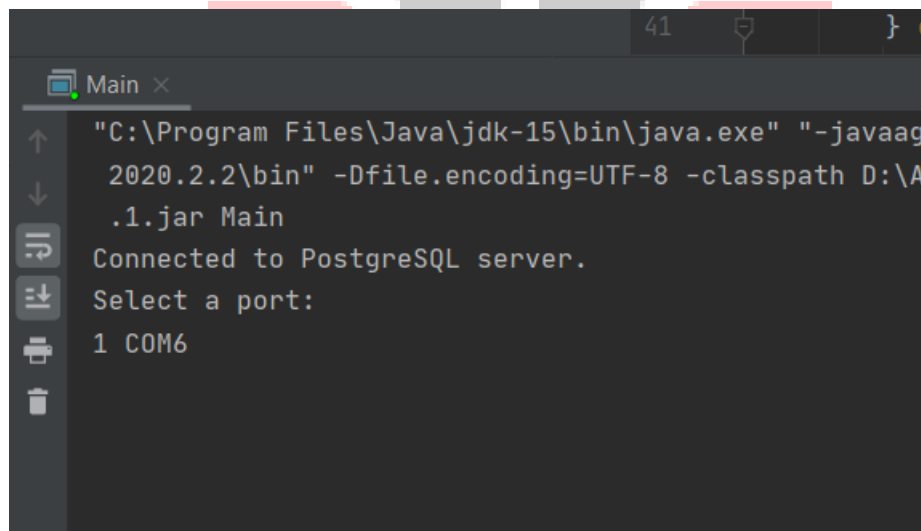
```

Conexiunea cu baza de date. Aici am implementat si o metoda care realizeaza stergerea efectiva a produselor cumparate din baza de date

Restul claselor realizeaza implementarea interfetei.

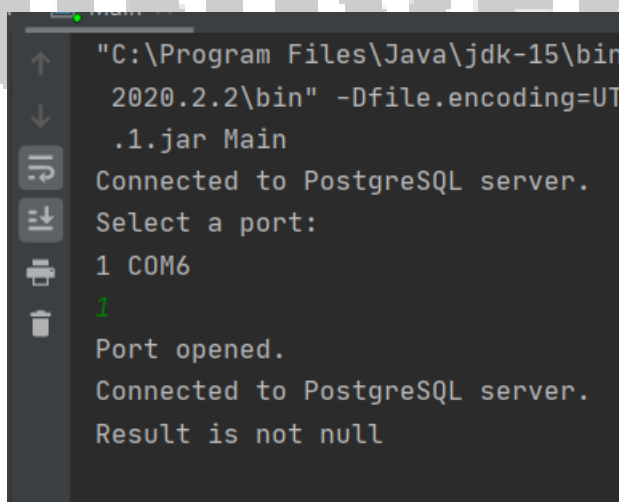
6.2. Utilizare

Dupa ce am conectat placuta si restul componentelor la lapto [rin intermediul cablului, vom rula aplicatia java.In prima faza, aceasta o sa ne afiseze toate porturile conectate la laptop si va astepta sa selectam cel dorit.Acest lucru se realizeaza prin introducerea numarului corespunzator.



```
"C:\Program Files\Java\jdk-15\bin\java.exe" "-javaag
2020.2.2\bin" -Dfile.encoding=UTF-8 -classpath D:\A
.1.jar Main
Connected to PostgreSQL server.
Select a port:
1 COM6
```

Dupa ce s-a realizat aceasta conexiune vom primi un mesaj de confirmare si ni se va deschide fereastra prin care putem observa produse.



```
"C:\Program Files\Java\jdk-15\bin
2020.2.2\bin" -Dfile.encoding=UTF
.1.jar Main
Connected to PostgreSQL server.
Select a port:
1 COM6
1
Port opened.
Connected to PostgreSQL server.
Result is not null
```

id_product	product_name	quantity	price
2	Apple	28	14
6	Orange	8	16
5	Juice	10	12
4	Water	31	15
3	Bread	7	12

Refresh

In momentul in care produsele au fost achizitionate (prin apasarea butoanelor) acestea vor fi trimise la aplicatie, vor fi prelucrate si apoi vor fi scazute din baza de date. In momentul in care este apasat butonul de refresh, datele vor fi actualizate si se va scadea cantitatea produselor.

- > Functions
- > Materialized Views
- > Procedures
- > Sequences
- > Tables (1)
- products
 - > Columns
 - > Constraints
 - > Indexes
 - > RLS Policies
 - > Rules
 - > Triggers
 - > Trigger Functions
 - > Types
 - > Views
- > Subscriptions
- > Login/Group Roles
- > Tablespaces (2)
 - pg_default
 - pg_global

Data Output	Explain	Messages	Notifications
id_product	product_name	quantity	price
1	2 Apple	28	14
2	3 Bread	7	12
3	4 Water	31	15
4	5 Juice	10	12
5	6 Orange	8	16

Baza de date cu tabelul in care am retinut datele.

7. Concluzie si posibile imbunatatiri.

O buna imbunatatire a proiectului ar fii aceea de a realiza trimiterea datelor la aplicatia Java si fara conexiunea prin cablu. Am incercat sa implementez acest lucru prin utilizarea unui modul Bluetooth prin care se pot transmite date, dar am avut dificultati tehnice la conectarea acestuia cu laptopul (posibil sa nu imi recunosca laptopul modulul sau modulul sa nu fie cel mai efficient).

O alta imbunatatire ar fi ca administratorul sa poata adauga produse/ sa schimbe pretul/ sa modifice cantitati in baza de date din interfata grafica si apoi sa actualizeze cosul de cumparaturi cu noile date.

Tehnologia RFID este foarte utila in multe alte domenii si poate fi folosita multe viitoare proiecte: Securitatea unei case, legitimarea persoanelor la locul de munca/scoala, tinerea in evidenta a cartilor dintr-o librerie, etc.

Acest proiect a avut ca scop realizarea unei modalitati de a evita cozile lungi la casele de marcat si realizarea unui mod mai usor si util pentru cumparatori de a vizualiza produsele cumparate, dar si pentru a usura munca unui administrator/manager de magazin.



TECHNICAL
UNIVERSITY
OF CLUJ-NAPOCA
ROMANIA

8. Bibliografie

1. - <https://technobyte.org/2016/07/power-up-the-arduino-uno/>
2. - https://users.utcluj.ro/~madalin/teaching/II/labs/Lab_infoind_4.pdf
3. <https://www.arduino.cc>
4. Java and Serial port - https://xiaozhon.github.io/course_tutorials/Arduino_and_Java_Serial.pdf
5. <https://www.instructables.com/Interfacing-RFID-RC522-With-Arduino-MEGA-a-Simple-/>
6. <https://arduinogetstarted.com/tutorials/arduino-lcd-i2c>

