

LICEUL TEORETIC "BENJAMIN FRANKLIN"

București, Sector 3

VENDING MACHINE

Elev:

Motronu Ioana-Cozia

Profesor îndrumător:

Niculescu Mihaela-Carmen

Mai, 2022

CUPRINS

Motivarea alegerii temei	3
Descrierea limbajelor folosite	4
Noțiuni de bază în limbajul C++	5
Noțiuni introductive	5
Noțiuni de bază în Arduino C++	8
Instalarea IDE-ului Arduino?.....	9
Calibrarea IDE-ului pentru dispozitivele ESP8266	13
Noțiuni de bază despre NodeMCU ESP8266	16
Noțiuni de bază în Vue.js.....	17
Date introductive despre Vue.js.....	17
Construirea unui program simplu care folosește Vue.js 3	18
Descrierea componentelor folosite.....	19
Numărătorul de monede	19
Display-ul	21
Motoarele	23
Modulul RFID	25
Stocarea datelor	28
Aplicația web.....	30
Meniul administratorului.....	32
Modul de funcționare al vedomatului	36
Bibliografie.....	37

Motivarea alegerii temei

Încă de când eram foarte mică am avut o pasiune pentru partea tehnică a lucrurilor și mi-a plăcut să creez diferite lucruri. Visul meu era să devin inventator, când am crescut mi-am dat seama că trebuie să învăț fizică pentru a face acest lucru posibil.

Povestea acestui vendomat se învârtă în jurul numărătorului de monede, ideea de vendomat venind de aici. Mai întâi am făcut posibilă afișarea pe un ecran a numărului de monede introduse și valoarea acestora, apoi am introdus motoarele acționate de butoane atunci când creditul este mai mare sau egal cu o valoare prestabilită de mine. În luna octombrie a anului trecut(2021) aproape terminasem proiectul, însă m-am gândit că ar fi mult mai interesant dacă aș adăuga o aplicație web din care utilizatorii să își poată vedea creditul rămas și produsele achiziționate. Deși pare destul de simplu la prima vedere, acest lucru mi-a dat mari bătaii de cap deoarece a trebuit să implementez o bază de date locală pe memoria dispozitivului(EEPROM) și să lucrez cu tehnologii noi pentru mine precum Vuejs, Fomantic-cadru de dezvoltare a diverselor machete receptive, SQLite sau ASP.NET Core în care am simulat serverul de pe dispozitiv.

Descrierea limbajelor folosite

Complexitatea acestui proiect nu este dată doar de multitudinea de componente fizice pe care le utilizează, ci și de limbajele de programare folosite. Programul principal este construit în Arduino, un subset al programului C++. Acestuia i se adaugă o bază de date în SQLite și o aplicație web ce folosește HTML, CSS, JavaScript și Vue.js.

Am separat acest proiect în trei părți pentru a fi mult mai ușor de construit. Prima dată am realizat vendomatul fizic. Acest lucru presupune punerea laolaltă a componentelor, cablarea acestora și construirea unui program care permite bună funcționare simbiotică a acestora. Deși nu a fost deloc ușor, am reușit să fac acest lucru posibil. În folderul proiectului se află câteva videoclipuri cu motoarele, display-ul și cum funcționează vendomatul.

În cea de-a doua parte am creat o bază de date în SQLite. Cu ajutorul acesteia am reușit să stochez informații legate de utilizatori și tranzacțiile acestora, însă mai multe pe această temă la capitolul „Descrierea bazei de date”.

În ultima dar nu cea din urmă parte am construit o aplicație web ce folosește HTML, CSS, JavaScript și Vue.js. Descoperirea unui framework precum [Fomantic](#) mi-a fost de mare ajutor deoarece nu am construit de la zero interfața web. Acesta conține o multitudine de elemente gata construite ce așteaptă să fie folosite, însă acestea au nevoie de implementare JavaScript și Vue.js pentru a funcționa în parametrii.

Pentru o scurtă perioadă de timp am folosit ASP.NET CORE pentru a simula serverul de pe dispozitiv. Am ales acest web framework deoarece oferă flexibilitate și un timp de răspuns mai bun comparativ cu ESP8266, acest server fiind creat direct pe laptopul meu în comparație cu serverul plăcuței pe care o folosesc în proiect. Însă mai multe despre acest framework puteți găsi în capitolul „Noțiuni elementare în ASP.NET CORE”.

Pentru fiecare dintre aceste părți am pregătit câte un capitol separat însoțit de explicații de la 0 pentru a vă putea ajuta să înțelegeți acest proiect fără experiențe anterioare în acest domeniu.

Noțiuni de bază în limbajul C++

Noțiuni introductive

C++ (numit C plus plus) este un limbaj de programare orientat pe obiecte creat de informaticianul Bjarne Stroustrup ca parte a evoluției familiei de limbaje C. A fost dezvoltat ca o îmbunătățire multiplatformă a C pentru a oferi dezvoltatorilor un grad mai mare de control asupra memoriei și a resurselor sistemului.

Unii numesc C++ „C cu clase” deoarece introduce principii de programare orientată pe obiecte, inclusiv utilizarea unor clase definite, în cadrul limbajului de programare C. De-a lungul timpului, C++ a rămas un limbaj foarte util nu numai în programarea computerelor în sine, ci și în predarea noilor programatori despre cum funcționează programarea orientată pe obiecte. Cu toate acestea, nu suportă doar orientat pe obiecte, ci și procedural și funcțional. Datorită flexibilității și scalabilității sale ridicate, C++ poate fi utilizat pentru a dezvolta o gamă largă de software, aplicații, browsere, interfețe grafice cu utilizatorul (GUI), sisteme de operare și jocuri.

Astăzi, C++ este încă foarte apreciat pentru portabilitatea sa notabilă, care permite dezvoltatorilor să creeze programe care pot rula pe diferite sisteme de operare sau platforme foarte ușor. În ciuda faptului că este un limbaj de nivel înalt, deoarece C++ este încă aproape de C, poate fi folosit pentru manipulare la nivel scăzut datorită relației sale strânse cu limbajul mașină.

Care a fost scopul creării acestui limbaj?

Bjarne Stroustrup a dezvoltat C++ la Bell Labs la începutul anilor 1980 pentru a îmbina cele mai bune avantaje ale mai multor alte limbaje de programare. El a vrut să combine rapiditatea BCPL, nivelul înalt al Simula și universalitatea lui Dennis Ritchie C. S-a inspirat și din alte limbi, cum ar fi Ada, ML și ALGOL 68, pentru a crea un program bine structurat, , limbaj de uz general care ar putea compila aproape toate programele C fără a-și schimba codul sursă. C++ este atât de flexibil încât este adesea supranumit „Cuțitul de buzunar elvețian al limbajelor de programare” (deși această poreclă este împărtășită și de Python).

Structura unui program C++

Deoarece este un limbaj proiectat în limba engleză, C++ folosește doar litere mari și mici ale alfabetului englez, cifre(0,1,2,...,9), set de caractere „+ - * / () { } @ \$ ”. La scrierea liniilor într-un program C++ se folosesc anumite caractere speciale denumite separatori precum spațiu „ ”, „, „;” folosit ca separator de linie în program, „, „ ” utilizat cu scopul de a separa elemente ale unui șir. Comentariu este un element ajutător pentru a scrie anumite explicații într-o linie de program folosind semnul backslash dublu „//”, sau pe mai multe linii de cod folosind backslash și steluță la începutul și sfârșitul comentariului „/* acesta este un comentariu */”.

Orice program este alcătuit din una sau mai multe module numite funcții. Este obligatoriu ca orice program să conțină funcția principală „main()”. Aceste paranteze pot conține parametri formali (constante sau variabile) sau nimic. Întotdeauna perechea de acolade se va scrie sub cuvântul „main” și va conține declarații ale variabilelor locale și instrucțiuni specifice programului.



```
//funcția principală
int main(int argc, char *argv[])
{
    //cod

    return 0;
}
```

În figura de mai sus am scris un program C++ care returnează 0. Cuvântul scris în fața lui main reprezintă tipul valorii calculate și returnate în funcția main. Programele C++ au o structură standard: directive preprocesor, variabile globale, funcții.


Directivele preprocesor se scriu la începutul fiecărui program astfel „#include <iostream>”. Această directivă preprocesor utilizează diferite biblioteci pentru a citi și a afișa informația. În programare, nu putem avea variabile, funcții etc. cu același nume. De aceea, pentru a evita aceste conflicte, folosim un namespace. Deci, pentru un namespace putem avea un nume unic și același nume poate fi folosit și într-un alt namespace.

Valorile pe care le folosim de-a lungul unui program pot fi constante sau variabile. Acestea se pot declara inițial „#define pi 3.14”, înaintea lui main și se numesc declarații globale deoarece se pot folosi în toate funcțiile programului, sau local într-una dintre funcții. Acestea sunt de mai multe tipuri precum int (valoare întreagă), float (valoare reală), bool (poate avea valoarea doar true sau false), char (reține un caracter) etc. Aceste valori suportă diferite operații matematice precum atribuirea, adunarea sau scăderea.

Operatorii limbajului C++ sunt:

1. Matematici: +, -, *, /,
2. Relaționali: <, >, <=, >=, ==, !=
3. Logici: !(negație), &&(și), || (sau)

Folosind informațiile enunțate mai sus se poate construi un program C++ simplu care afișează pe ecran un mesaj.

A screenshot of a code editor window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. The code is written in C++ and includes comments in Romanian. It uses the iostream header and the std namespace to print a message.

```
//directive preprocesor
#include <iostream>
using namespace std;

//funcția principală
int main(){
    //afișarea unui mesaj
    cout<<"Atestat 2022";
}
```

Noțiuni de bază în Arduino C++

Proiectul Arduino a fost demarat la Institutul de Interaction Design Ivrea (IDII) din Ivrea, Italia. La acea vreme, studenții foloseau un microcontroler BASIC Stamp la un cost de 200 de lei. În 2003, Hernando Barragán a creat platforma de dezvoltare Wiring ca proiect de teză de masterat la IDII, sub supravegherea lui Massimo Banzi și Casey Reas. Casey Reas este cunoscut pentru că a creat, împreună cu Ben Fry, platforma de dezvoltare Processing. Scopul proiectului a fost acela de a crea instrumente simple și cu costuri reduse pentru crearea de proiecte digitale de către cei care nu sunt ingineri.

Platforma Wiring era compusă dintr-o placă de circuite imprimate (PCB) cu un microcontroler ATmega128, un IDE bazat pe Processing și funcții de bibliotecă pentru a programa cu ușurință microcontrolerul. În 2005, Massimo Banzi, împreună cu David Mellis, un alt student IDII, și David Cuartielles, au extins Wiring prin adăugarea suportului pentru microcontrolerul ATmega8, mai ieftin. Noul proiect, bifurcat din Wiring, s-a numit Arduino.

Hardware-ul original Arduino a fost fabricat de compania italiană Smart Projects. Unele plăci Arduino au fost proiectate de companiile americane SparkFun Electronics și Adafruit Industries. Începând cu 2016, au fost produse în comerț 17 versiuni ale hardware-ului Arduino.

Arduino este un subset al limbajului C++ ce se bazează pe cablarea fizică a componentelor. Este o istorie lungă de proiecte bazate pe alte proiecte, într-un mod foarte deschis. IDE-ul Arduino este compus din două medii, IDE-ul de procesare și IDE-ul de cablare.

În lucrul cu Arduino se folosește în mod obișnuit Arduino IDE (Integrated Development Environment), un software disponibil pentru toate platformele desktop majore (macOS, Linux, Windows), care ne oferă două lucruri: un editor de programare cu suport pentru biblioteci integrate și o modalitate de a compila și de a încărca ușor programele Arduino pe o placă conectată la computer.

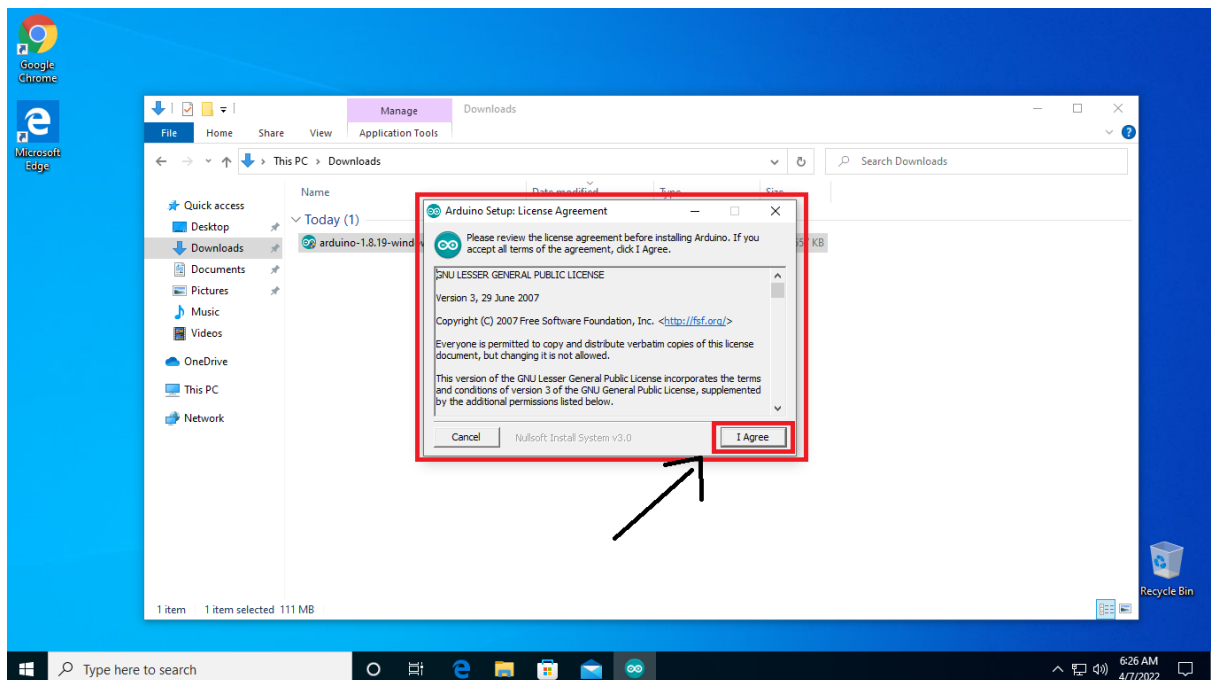
Arduino este mai exact o platformă de electronice open-source ce se bazează pe hardware și software. Plăcile dezvoltate de Arduino sunt capabile de citirea unor input-uri precum apăsarea unui buton și transformarea acestora în output-uri precum aprinderea unui led. Toate aceste lucruri sunt posibile prin transmiterea unui set de instrucțiuni microcontrolerului de la bordul plăcuței.

Instalarea IDE-ului Arduino?

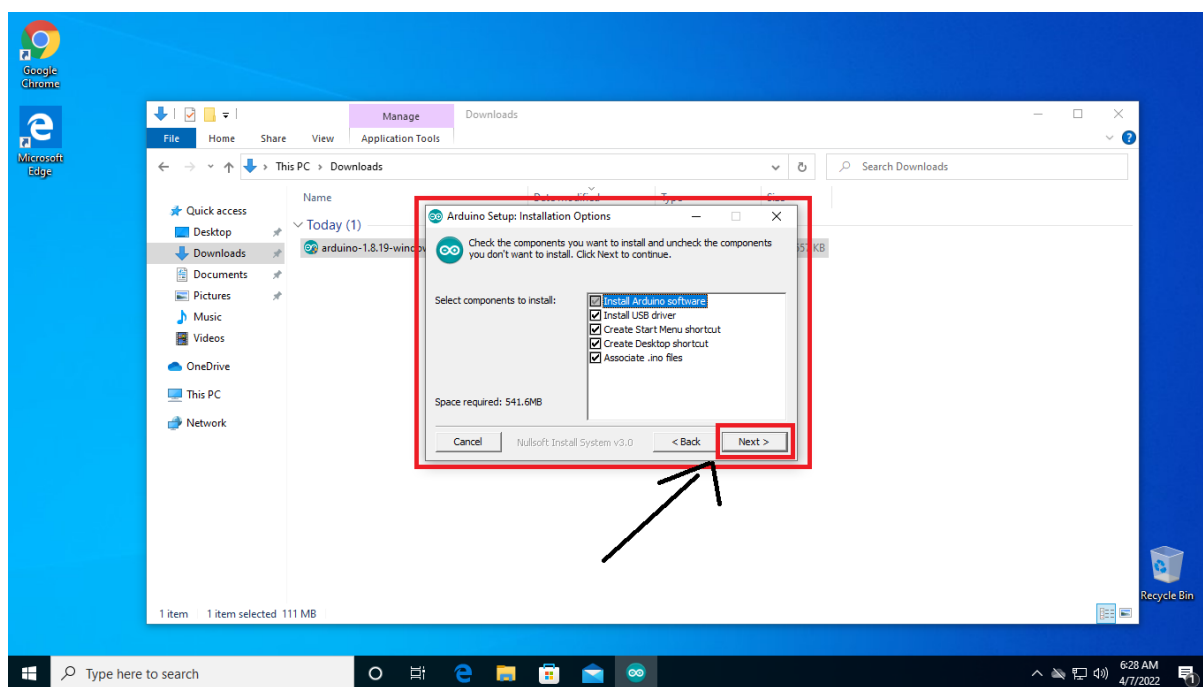
Pentru a instala Arduino IDE este necesară accesarea acestui link <https://www.arduino.cc/en/software> și descărcarea unei versiuni potrivite sistemului de operare.

The screenshot shows the Arduino IDE 1.8.19 download page. The page has a teal header with navigation links: PROFESSIONAL, EDUCATION, STORE, a search bar, and SIGN IN. Below the header is a secondary navigation bar with links: HARDWARE, SOFTWARE (highlighted), CLOUD, DOCUMENTATION, COMMUNITY, BLOG, and ABOUT. The main content area is titled "Downloads" and features a large card for "Arduino IDE 1.8.19". The card includes the Arduino logo, the version number, a description of the IDE, and a "SOURCE CODE" section. To the right of the card is a "DOWNLOAD OPTIONS" box with a red border, listing download links for Windows (Win 7 and newer, ZIP file), Windows app (Win 8.1 or 10), Linux (32 bits, 64 bits, ARM 32 bits, ARM 64 bits), and Mac OS X (10.10 or newer). Below the card are two sections: "Hourly Builds" and "Previous Releases". A "Help" button is visible in the bottom right corner.

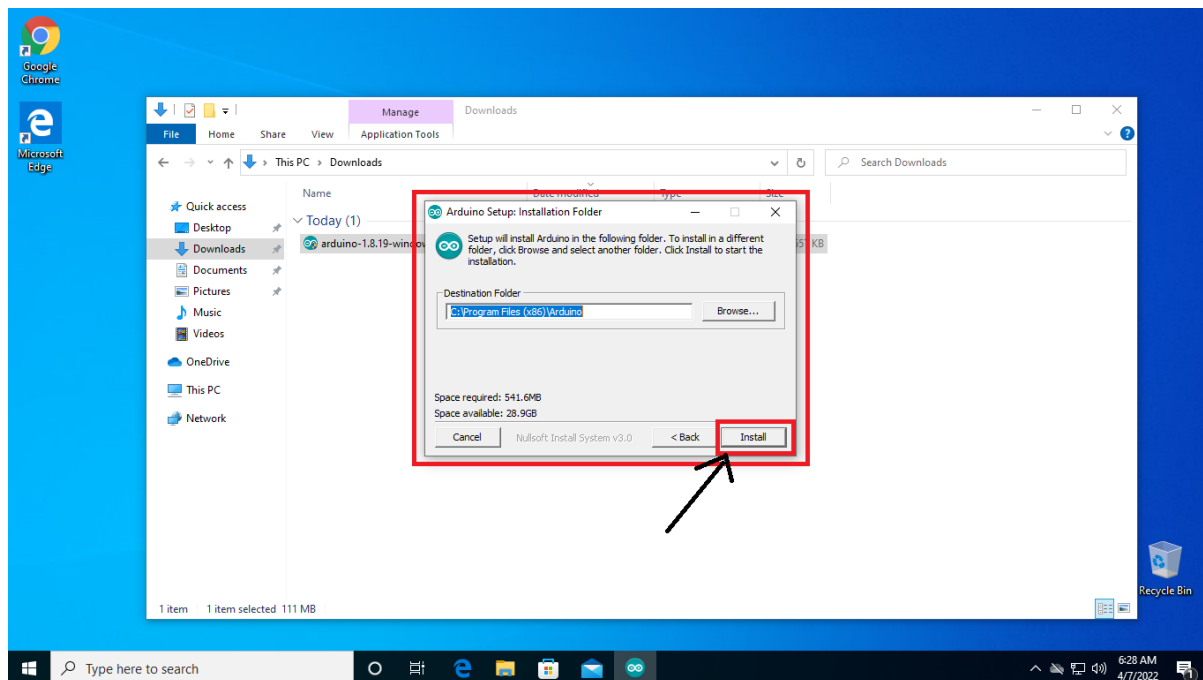
După descărcare se urmează procesul de instalare de tip „Next->Next” asemănător cu cel din imaginile următoare:



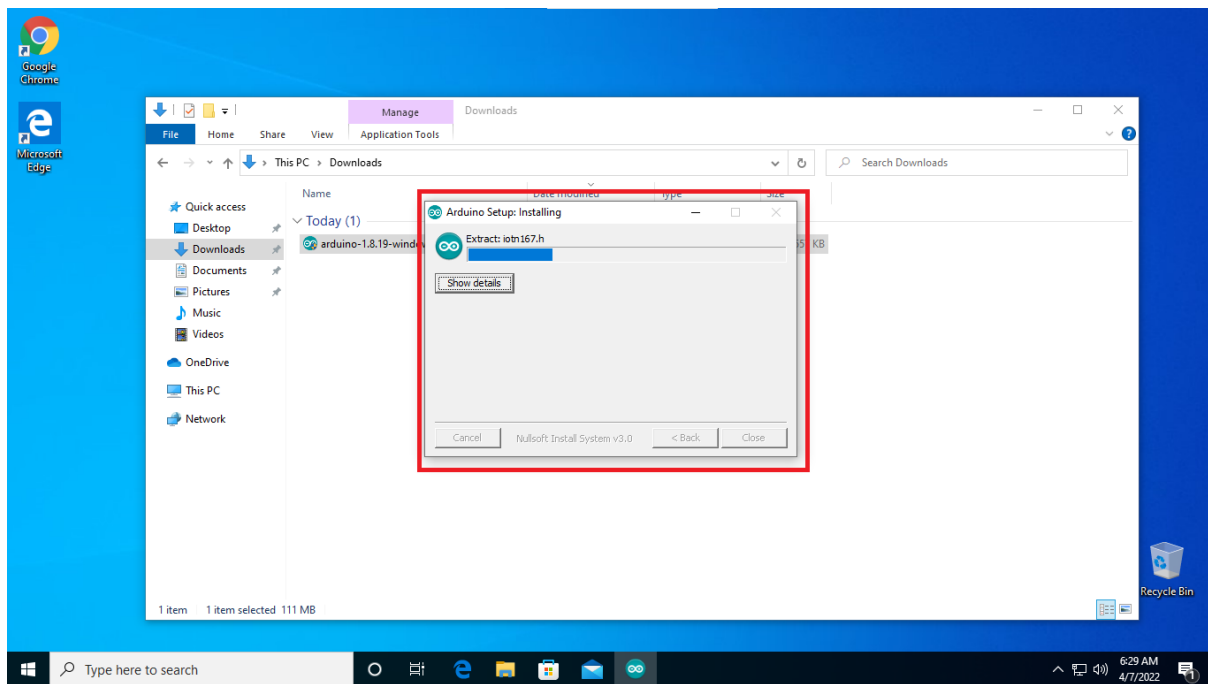
1. Apăsați „I Agree” pentru a continua procesul de instalare.



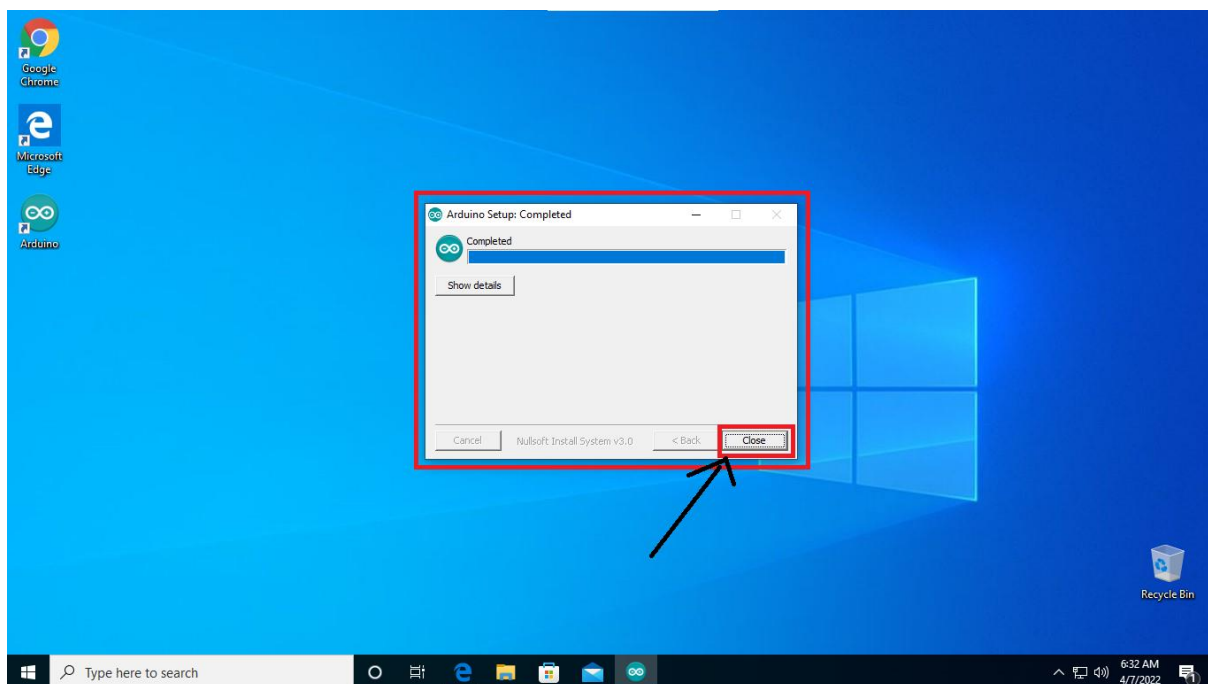
2. Apăsați „Next” pentru a continua procesul de instalare.



3. Apăsați „Install” pentru a instala Arduino IDE.

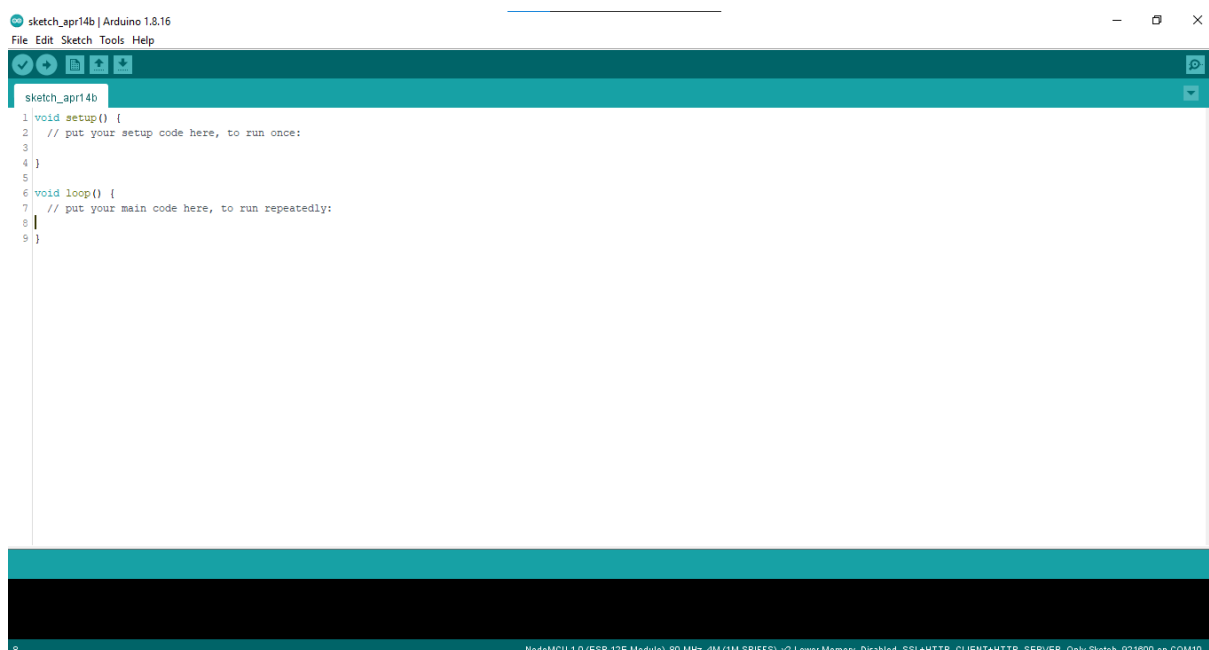
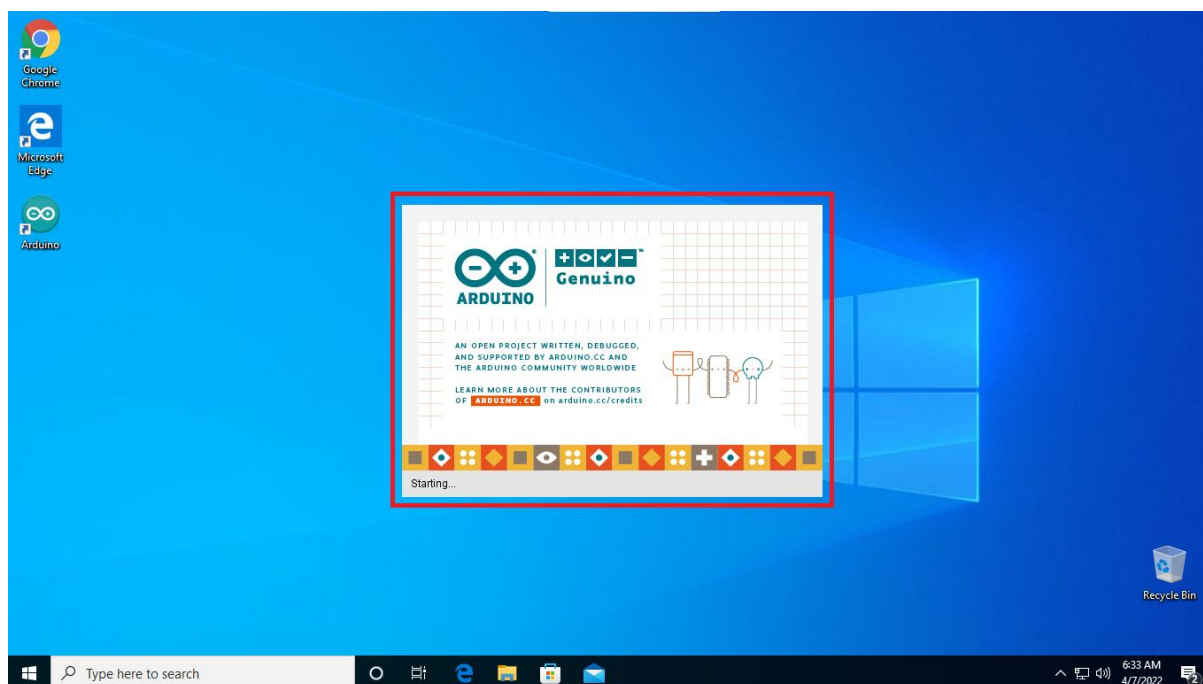


4. Așteptați instalarea aplicației.



5. După ce aplicația s-a instalat apăsați butonul „Close”.

Deschiderea mediului de programare va arăta ca în imaginea de mai jos:

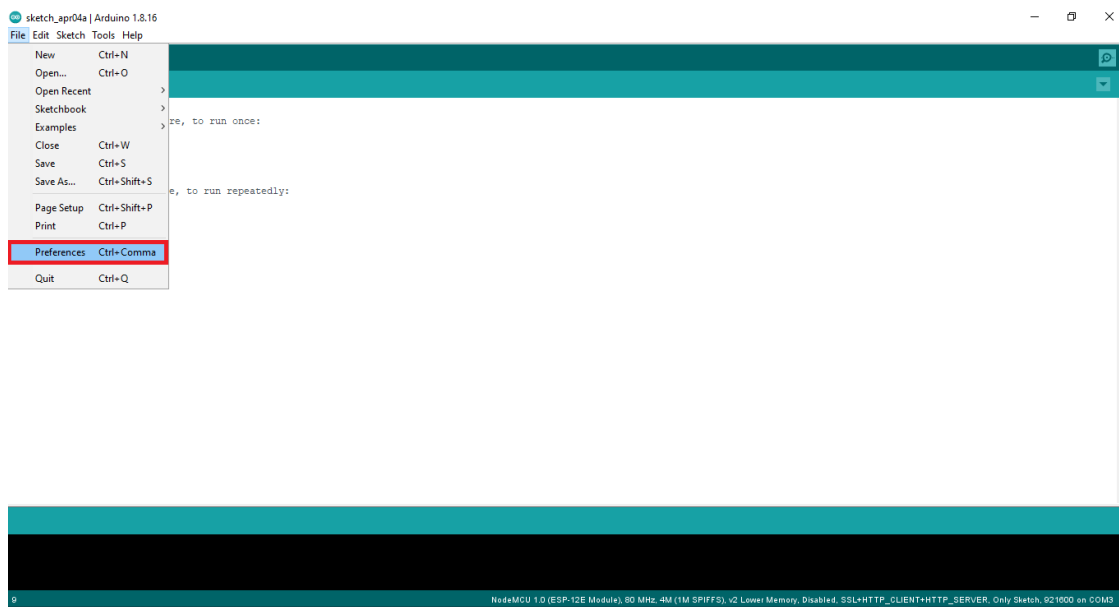


În cazul în care nu ați mai folosit Arduino se va deschide o filă goală asemănătoare cu cea din imaginea de mai sus.

Calibrarea IDE-ului pentru dispozitivele ESP8266

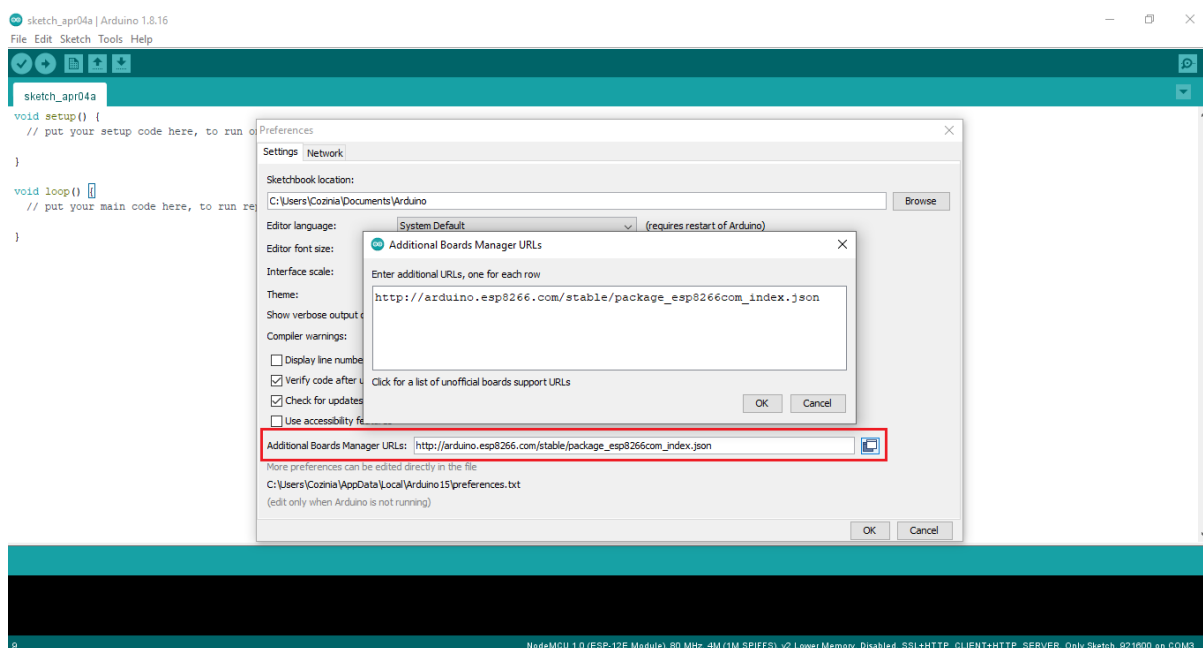
Arduino IDE vine presetat pentru a putea fi folosit de dispozitivele Arduino Uno. Deoarece în acest proiect voi folosi un ESP8266 trebuie să facem câteva schimbări:

1. Adăugarea unui link care permite descărcarea librăriilor speciale pentru ESP8266
 - a) După deschiderea programului Arduino intrați în File/Preferences/Settings/Additional Boards Manager URLs



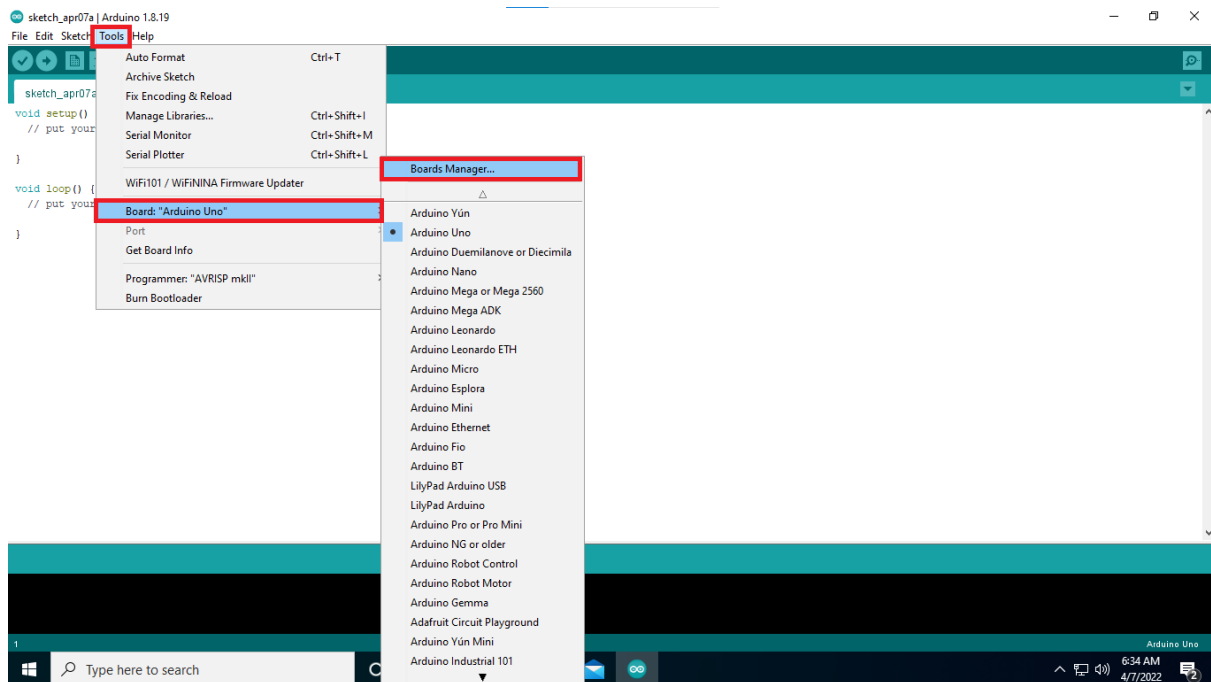
- b) Se inserează link-ul de mai jos în căsuța cu numele „Additional Boards Manager URLs”

http://arduino.esp8266.com/stable/package_esp8266com_index.json

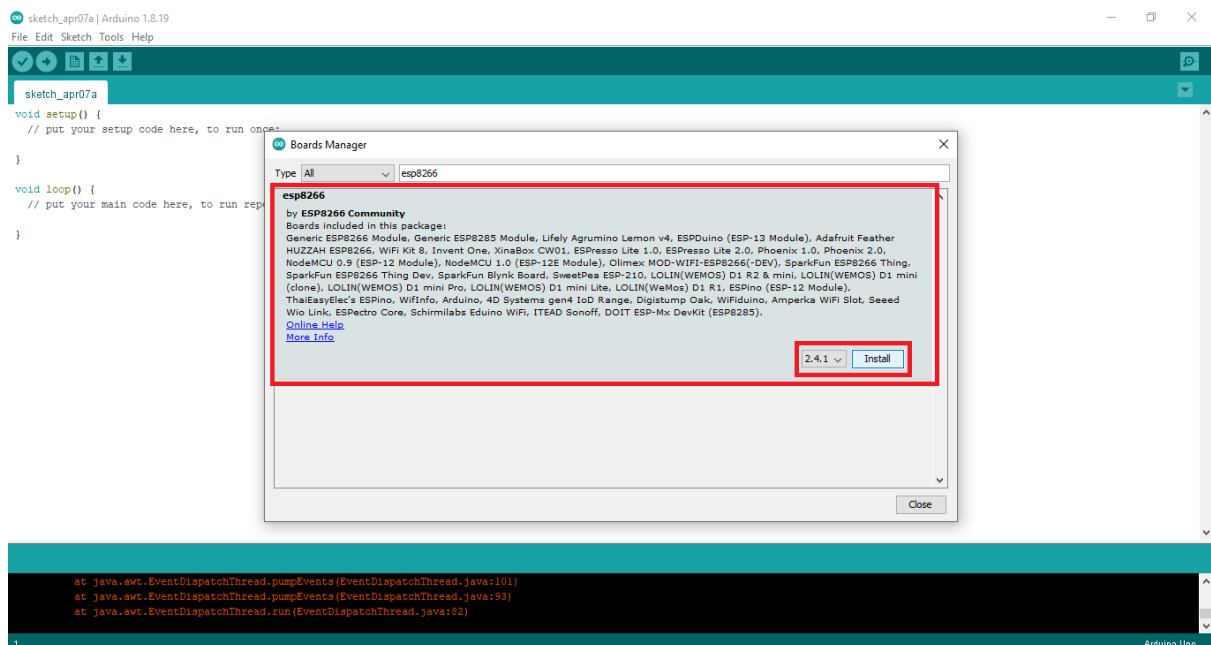


2.Instalarea librăriei ESP8266

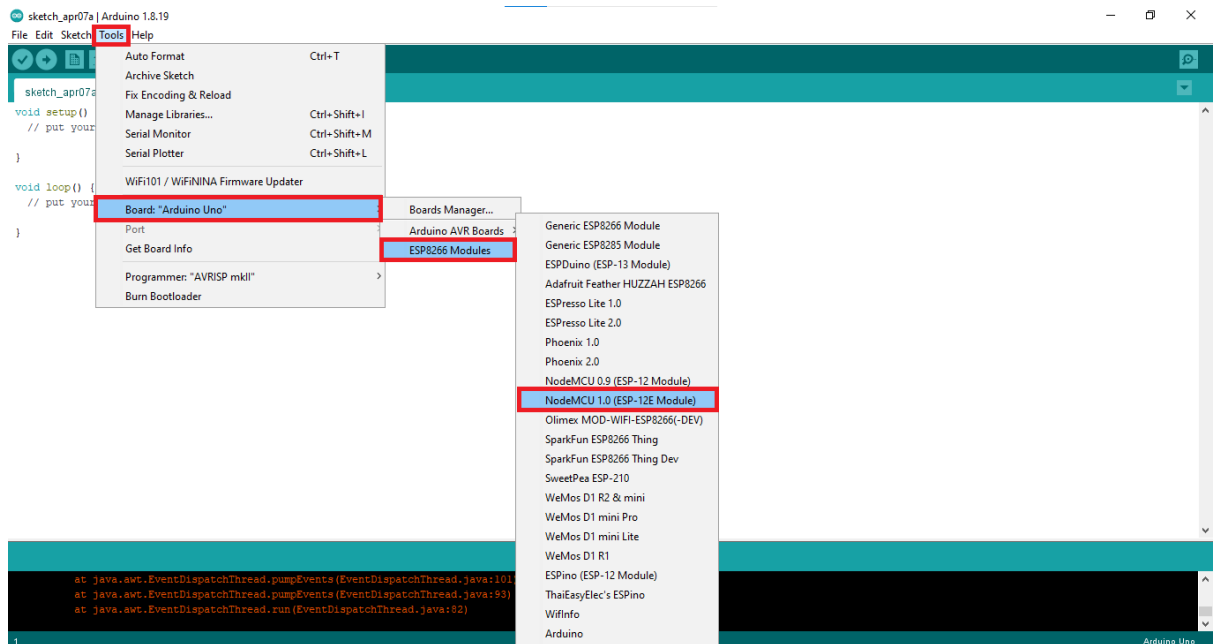
a) Se deschide meniul Tools/Boards/Board Manager



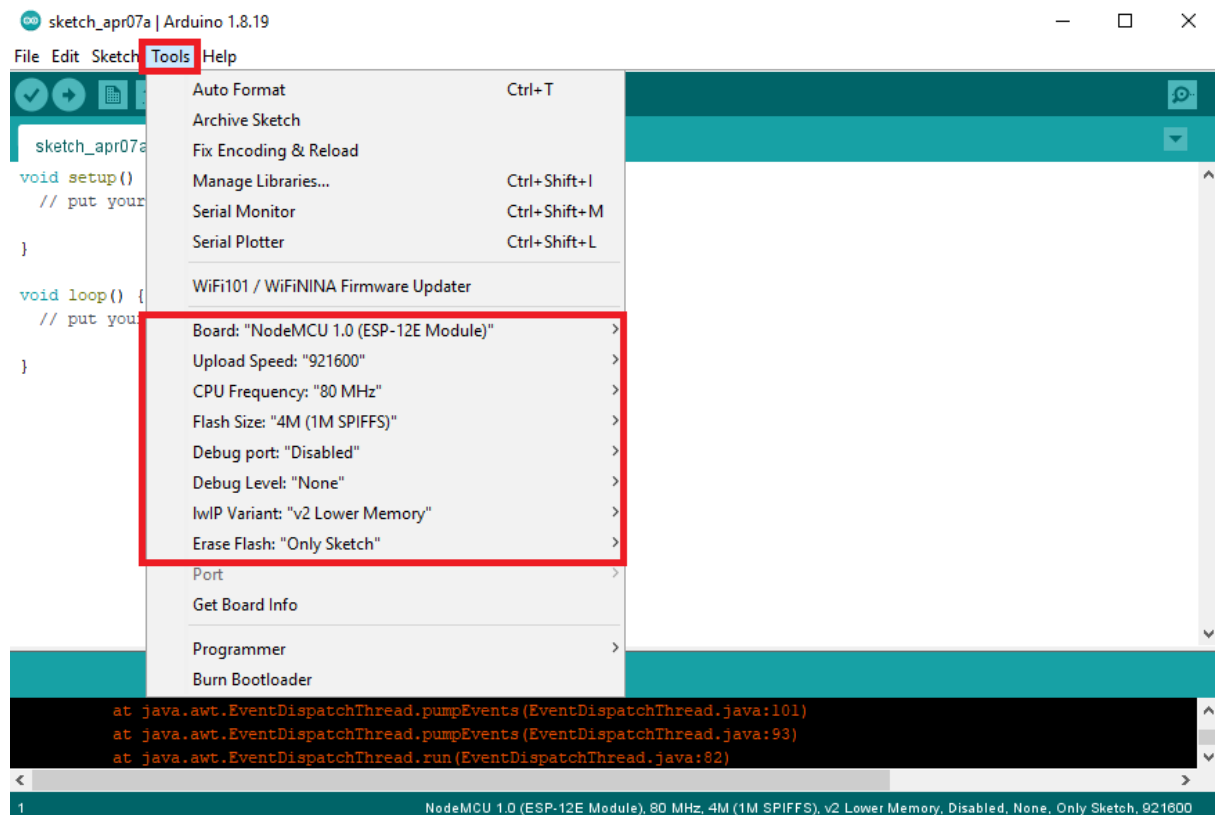
b) În fereastra deschisă se caută „esp8266 by ESP8266 Community” și se instalează versiunea 2.4.1(aceasta este versiunea folosită în acest proiect)



c) Se deschide fereastra Tools/Board/ și se selectează opțiunea „ESP8266 Modules/NodeMCU 1.0(ESP-12E Module)”



3. După instalarea librăriei pentru ESP8266 setările din aplicație ar trebui să arate ca în imaginea de mai jos:

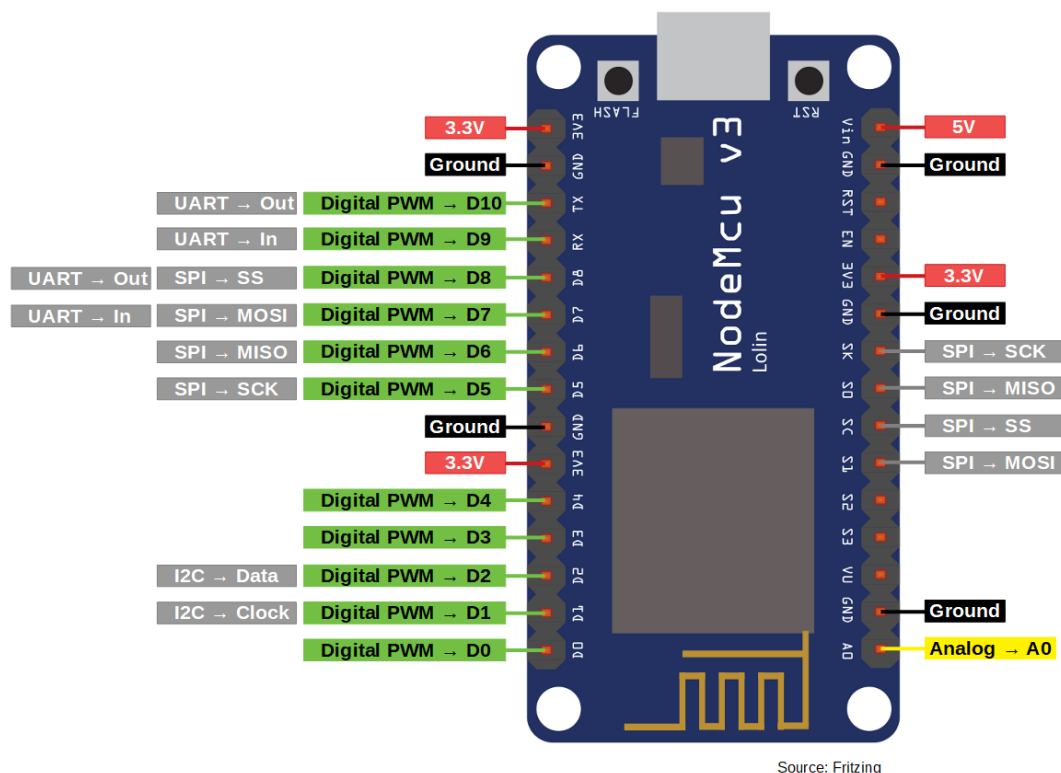


Acum Arduino IDE este pregătit pentru a fi folosit!

Noțiuni de bază despre NodeMCU ESP8266

Aceste plăcuțe sunt de diferite feluri însă în acest proiect se folosește un ESP8266(versiunea 12E), așadar mai jos sunt descrise câte lucruri esențiale despre acest tip de plăcuțe.

Mai jos este inserată o poză ce reprezintă configurația pinilor unui ESP8266.



Placa de dezvoltare NodeMCU ESP8266 vine cu modulul ESP-12E care conține cipul ESP8266 având microprocesor Tensilica Xtensa pe 32 de biți LX106 RISC. Acest microprocesor acceptă RTOS și funcționează la o frecvență reglabilă de la 80 MHz la 160 MHz. NodeMCU are 128 KB RAM și 4 MB de memorie Flash pentru a stoca date și programe. Puterea sa mare de procesare cu Wi-Fi sau Bluetooth încorporat și funcțiile de operare în stand-by îl fac ideal pentru proiecte IoT.

Aceste dispozitive au o memorie totală de 32MB(doar 4 MB sunt disponibili pentru a scrie programe), 11 pini digitali(D0-D10), un pin Analog(A0), 3 pini ce pot oferi curent de 3.3V, un pin ce oferă 5V, 4 pini de ground. Aceste microprocesoare suportă I2C(magistrală de comunicații seriale) pe pinii D1(clock) și D2(Data) și suport de extindere a memoriei printr-un modul de SD card ce folosește pinii MISO, MOSI, SCK, SS.

Pinii I2C sunt folosiți pentru a conecta senzori și periferice I2C. Sunt acceptate atât I2C Master, cât și I2C Slave. Funcționalitatea interfeței I2C poate fi realizată programatic, iar frecvența este de maximum 100 kHz. Trebuie remarcat faptul că frecvența I2C ar trebui să fie mai mare decât frecvența cea mai lentă a dispozitivului slave.

Noțiuni de bază în Vue.js

Date introductive despre Vue.js

Povestea Vue.js începe în 2013, când Evan You lucra la Google, creând o mulțime de prototipuri chiar în browser. În acest scop, Evan a folosit practici din alte framework-uri cu care a lucrat și a lansat Vue.js în mod oficial în 2014. De atunci, acesta evoluează continuu. La începutul lui 2018, Vue.js a început să bată Angular și să devină mai faimos pe piață. Mai târziu, în septembrie 2018, Evan You a decis să anunțe lansarea Vue 3.0. Vue.js evoluează continuu odată cu creșterea rapidă a utilizării și a comunității acestui framework. Comunitatea va continua să crească, deoarece a fost construită pe cea mai bună combinație de caracteristici Angular și React.

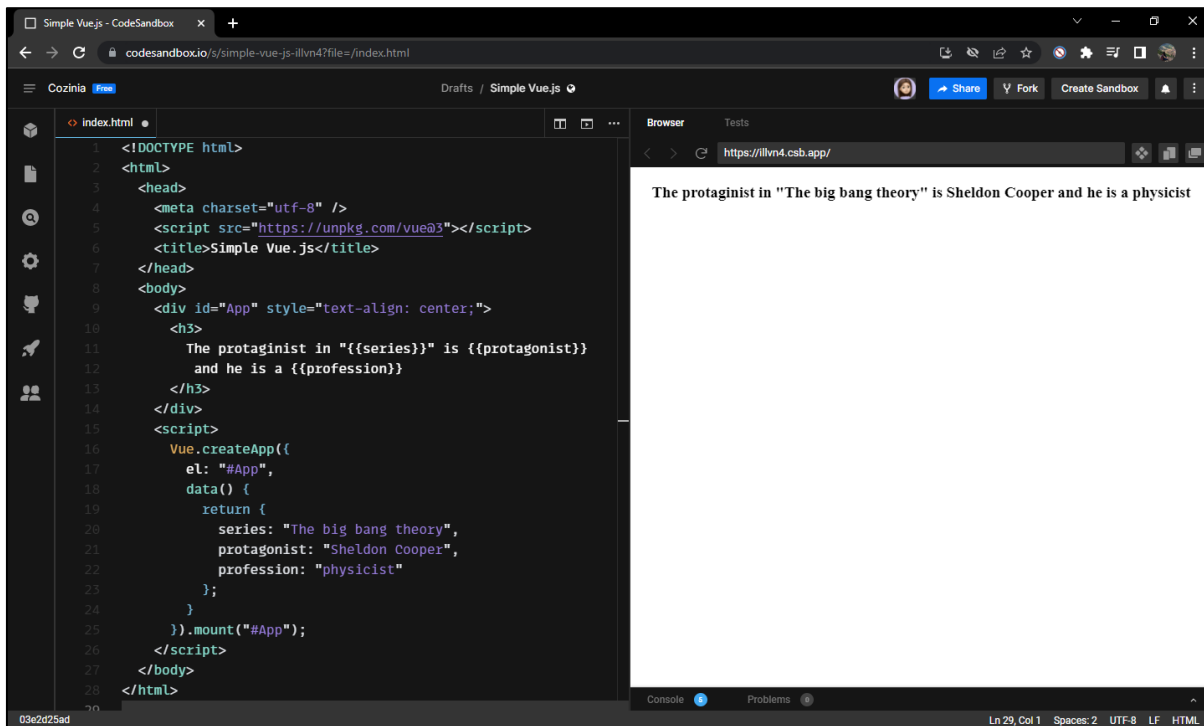
Într-un interviu, Evan dezvăluie că inițial Vue.js a fost o încercare de a lua ce e mai bun din Angular și de a construi un instrument personalizat, dar mai ușor: "Pentru mine, Angular oferea ceva grozav, și anume data binding și un mod de a trata un DOM bazat pe date, astfel încât să nu fie nevoie să atingi tu însuți DOM-ul."

Vue.js este un framework progresiv pentru JavaScript folosit pentru a construi interfețe web și aplicații de o pagină. Vue.js nu este folosit doar pentru interfețe web, ci și pentru dezvoltarea de aplicații desktop și mobile cu ajutorul framework-ului Electron. Extensia HTML și baza JS au făcut rapid din Vue un instrument front-end favorit, dovadă fiind adoptarea de către giganți precum Adobe, Behance, Alibaba, Gitlab și Xiaomi.

Vue.js este un framework JavaScript progresiv, care este folosit pentru a construi interfețe UI (interfețe ușor de folosit de către utilizatori) și SPA (aplicații cu o singură pagină). Acest framework este renumit pentru curba sa de învățare rapidă. Este o bibliotecă atât de ușor de învățat și accesibilă, încât, putem începe să construim aplicații web în Vue.js folosind cunoștințele de HTML, CSS și JavaScript. Curba de învățare rapidă este un fel de semnătură a acestui framework.

Construirea unui program simplu care folosește Vue.js 3

Deoarece în cadrul acestui proiect este folosită versiunea a treia a framework-ului Vue.js, mai jos este prezentată o mică aplicație care folosește această versiune de Vue.



Așa cum se poate vedea în imaginea de mai sus s-a construit un program foarte simplu în Vue.js. În următoarele rânduri este enunțată procedura de creare:

- 1) Construirea un fișier HTML simplu.
- 2) Pentru a folosi Vue.js se inserează în header următoarea linie de cod:
`<script src="https://unpkg.com/vue@3"></script>`

P.S: Aceasta se numește CDN și ne ajută să folosim vue.js în aplicația noastră HTML fără a fi nevoiți să instalăm framework-ul.

- 3) Se adaugă un `<div>` cu id-ul „App”.
- 4) În body se deschide un script.
 - a) În script se adaugă aplicația „Vue.createApp” în cadrul căreia se construiesc componentele Vue.js.
 - b) În „data()” se introduc valorile „series”, „protagonist” și „profession” cărora li se atribuie câte un string cu diferite informații.
- 5) Pentru ca aplicația Vue.js să funcționeze este importantă adăugarea sintaxei „.mount(„#App”)”. Această instrucțiune „împachetează totul”.

P.S.: Aplicația web de mai sus poate fi vizualizată la adresa:

<https://codesandbox.io/s/simple-vue-js-illvn4?file=/index.html:64-112>

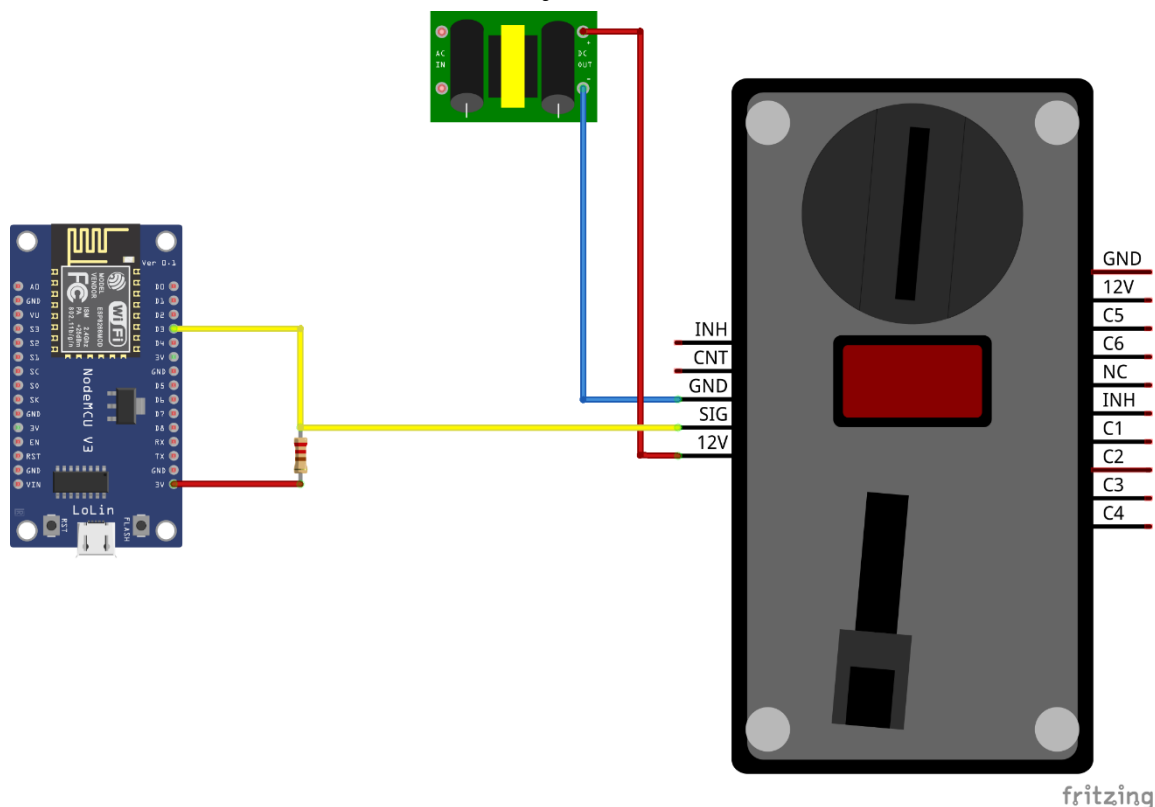
Descrierea componentelor folosite

În cadrul acestui proiect voi fi folosite mai multe componente așa că mai jos se află o mică prezentare a acestora și cum sunt legate la microcontroler.

Numărătorul de monede

Primul pe ordinea de idei este numărătorul de monede, CH-926 pe numele de cod, acesta acceptă până la 3 tipuri diferite de monede.

Acest modul trebuie conectat la o sursă de 12V , iar pinul ce se ocupă cu transmiterea mesajului de identificare a monedelor trebuie conectat la unul dintre pinii digitali ai microcontroler-ului ca în schema de mai jos.



În desen se observă microcontroler-ul în partea stângă, sursa de 12V în partea de sus și numărătorul de monede în partea dreaptă. Acesta este conectat la sursa externă de 12V prin cele două fire (roșu pentru VCC și albastru pentru GND), iar prin firul de culoare galbenă la ESP8266, mai exact la 3V3 ce unește printr-un tranzistor pinul SIG (Coin) al acceptorului de monede cu pinul digital D3 al plăcuței.

Modulul este programat pentru a distinge 3 monede diferite (de 5, 10 și 50 bani). Acesta se bazează pe pulsuri consecutive pentru fiecare tip de monedă în parte (un puls pentru 5 bani, două pulsuri pentru 10 bani, trei pulsuri pentru 50 de bani).

Iată câteva secvențe de cod interesante:

```
attachInterrupt(digitalPinToInterrupt(CountPin), CoinEvent, CHANGE);
```

Pentru a nu supraîncărca plăcuța cu citiri successive se folosește **attachInterrupt**, metodă cu ajutorul căreia funcția **CoinEvent** este invocată doar la detectarea introducerii unei monede.

```

145 void CoinEvent(){
146     crtValue = digitalRead(CountPin);
147
148     Serial.print("CoinEVENT_____");
149     Serial.print("\n");
150     Serial.print(crtValue);
151     Serial.print("\n");
152
153 }

```

digitalRead citește valoarea pinului Count și o stochează în variabila crtValue.

Funcția Coins se apelează în loop. Dacă timpul curent este diferit de cel anterior iar valoarea curentă de cea anterioară, timpul anterior ia valoarea celui prezent și valoarea anterioară ia valoarea celei prezente. Variabila de tip bool numită changed își schimbă valoarea în true când această condiție este îndeplinită. Dacă diferența dintre timpul curent și cel anterior este mai mare de 30 de milisecunde, iar changed este true, variabila number crește cu o unitate.

```

81 void Coins(){
82     //take current time
83     crtTime=millis();
84     //compare current time with previous time
85     if(crtTime > prevTime)
86     {
87         //compare current value with previous value
88         if(crtValue != prevValue)
89         {
90             prevTime = crtTime;
91
92             prevValue = crtValue;
93             changed = true;
94         }
95         if(crtTime - prevTime>30 && changed)
96         {
97             counter ++;
98             Serial.print(counter);
99             Serial.print("\n" );
100             changed = false;
101             number ++;
102             Time1 = millis();
103             changed =false;
104         }
105     }
106     Time2 = millis();
107     if((Time2-Time1>TIMER) && (number)){
108
109         Serial.print("Number: ");
110         Serial.print(number);
111         Serial.print("\n");
112         Serial.print("Time2-Time1: ");

```

```

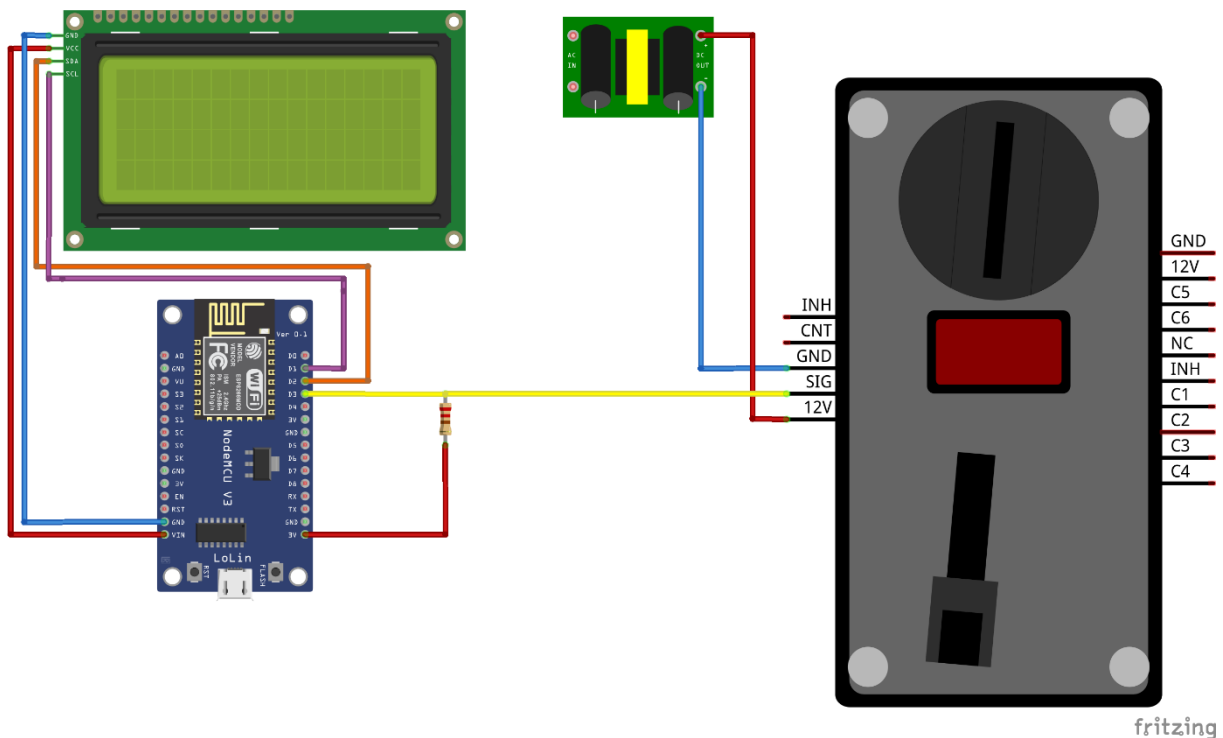
112     Serial.print("Time2-Time1: ");
113         Serial.print(Time2-Time1);
114         Serial.print("\n");
115
116     if(number==3){
117         credit=credit+0.50;
118         totalCoins3 ++;
119         number=0;
120     }
121     else if(number==2){
122         credit=credit+0.10;
123         totalCoins2 ++;
124         number=0;
125     }
126     else
127         if(number==1){
128             totalCoins1 ++;
129             credit=credit+0.05;
130             number=0;
131         }

```

În funcție de valoarea variabilei number, aceasta îndeplinește una dintre condițiile de mai sus. În acest fel creditul crește cu o anumită valoare(5, 10 sau 50 de bani), valoarea number este resetată la 0, iar variabila totalCoins crește cu o unitate pentru a înregistra numărul total de monede de un anumit tip.

Display-ul

După punerea în funcțiune a numărătorului de monede a fost adăugat și un display pe care să afișeze creditul inserat. Mai jos este inserată o schemă ce conține și display-ul.

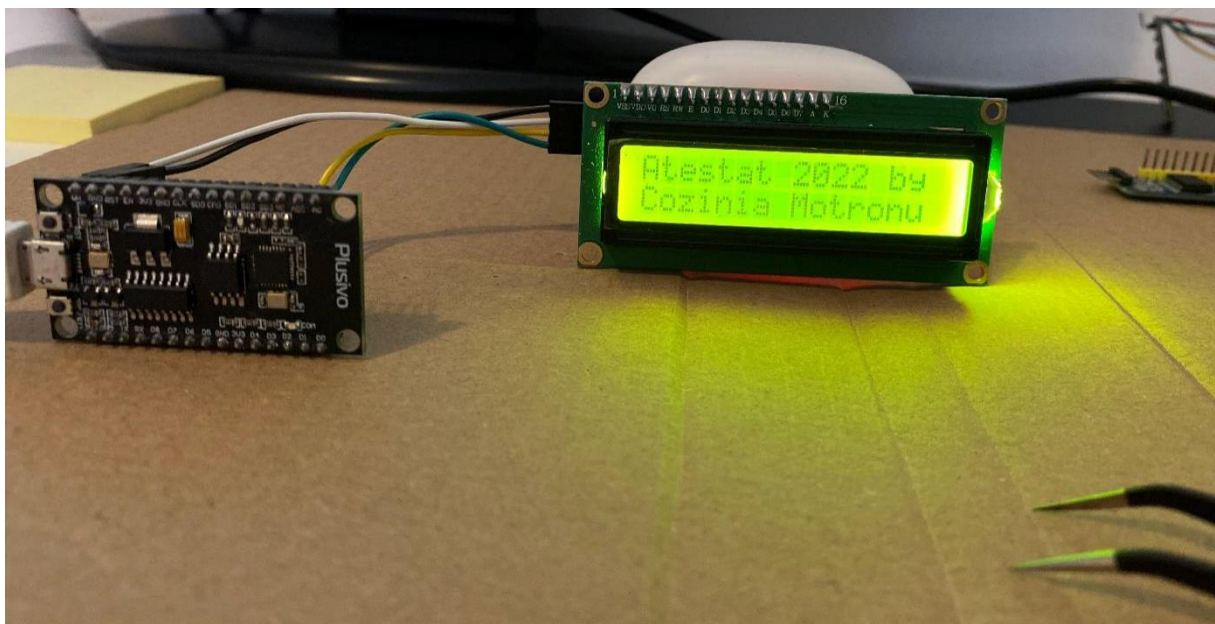


Display-ul comunică prin protocolul I2C cu microcontroler-ul. Pentru a face acest lucru posibil se conectează pinii D1 la SCL(reprezentat prin cablul de culoare mov), D2 la SDA(reprezentat prin cablul de culoare oranj), pinul VIN(cel care transportă curent de 5V) la VCC și GND la GND.

```
1 //YWR0BOT
2 //Compatible with the Arduino IDE 1.0
3 //Library version:1.1
4 #include <Wire.h>
5 #include <LiquidCrystal_I2C.h>
6
7 LiquidCrystal_I2C lcd(0x27, 16, 2); // set the LCD address to 0x27 for a 16 chars and 2 line display
8
9 void setup()
10 {
11     lcd.init();           // initialize the lcd
12     // Print a message to the LCD's screen
13     lcd.backlight();
14     lcd.setCursor(0, 0);
15     lcd.print("Atestat 2022 by");
16     lcd.setCursor(0, 1);
17     lcd.print("Cozina Motronu");
18 }
19
20
21
22 void loop()
23 {
24 }
```

Acest cod folosește librăria „LiquidCrystal_I2C”. După declararea directivelor preprocesor „Wire” și „LiquidCrystal_I2C” se inițializează obiectul lcd care se află la adresa 0x27. În funcția setup se inițializează lcd, backlight și se setează cursorul pe poziția linia 0 și coloana 0 pentru a scrie un mesaj, respectiv lini 1 și coloana 0 pentru continuarea mesajului.

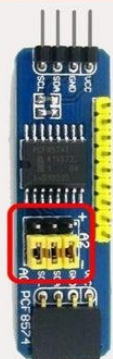
Mai jos este inserată o poză cu un mesaj afișat pe display.



Motoarele

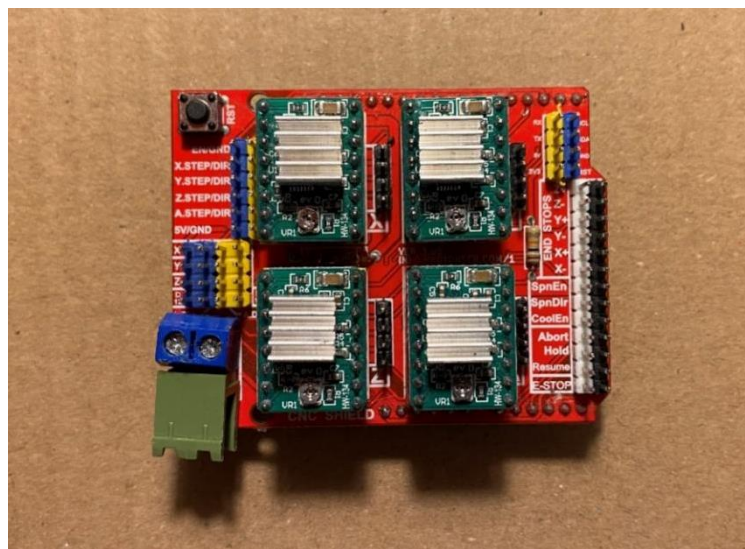
Acum schema începe să se încarce deoarece fiecare motor are 4 fire cu ajutorul cărora se conectează la shield-ul CNC și alte două care duc la modulul de expansiune PCF8574. Se folosesc aceste module deoarece atât motoarele, cât și celelalte componente ocupă foarte mulți pini și nu lasă loc disponibil pe microcontroler. PCF8574 folosește același protocol precum display-ul, adică I2C pentru a transmite diferite evenimente ce se petrec pe pini. Acest lucru este convenabil deoarece se pot lega mai multe module de expansiune pentru a extinde numărul de pini. Aceste module se diferențiază prin adresă, fiecare putând fi programat manual ca având o anumită adresă (de la 0x20 la 0x27) așa cum este ilustrat în imaginea de mai jos.

PCF8574 Addressing			
A0	A1	A2	Address Pins
0	0	0	= 0x20
0	0	1	= 0x21
0	1	0	= 0x22
0	1	1	= 0x23
1	0	0	= 0x24
1	0	1	= 0x25
1	1	0	= 0x26
1	1	1	= 0x27

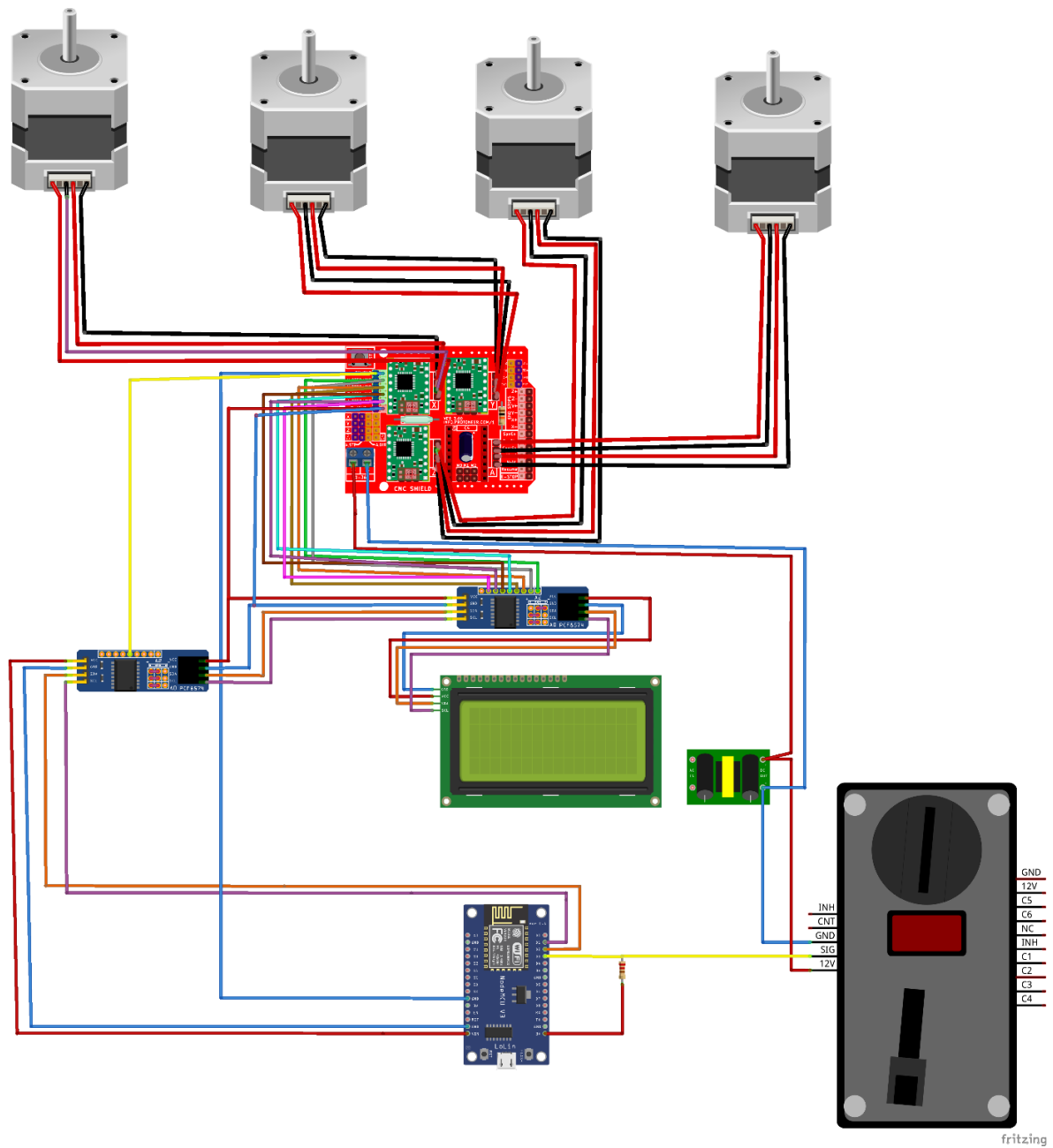


Shield-ul CNC împreună cu driver-ul de motoare A4988 asigură buna funcționare a motoarelor oferind o interfață ușor de utilizat.

Shield-ul oferă suport pentru maxim 4 drivere de motoare, un pin Enable(EN) pentru a activa mișcarea motoarelor, mai mulți pini de ground(GND), pini care permit mișcarea motoarelor într-o anumită direcție și un anumit număr de pași, de exemplu: X.STEP(indică numărul de pași pe care motorul X să îl parcurgă), X.DIR(direcția în care să se învârtă motorul X). Shield-ul trebuie alimentat cu 12V de la o sursă externă și 5V(curent necesar cerut de driver-ele de motoare).



Mai jos se poate observa schema proiectului la care se adaugă motoarele, shield-ul cu driver-ele de motoare și modulele de expansiune a pinilor.



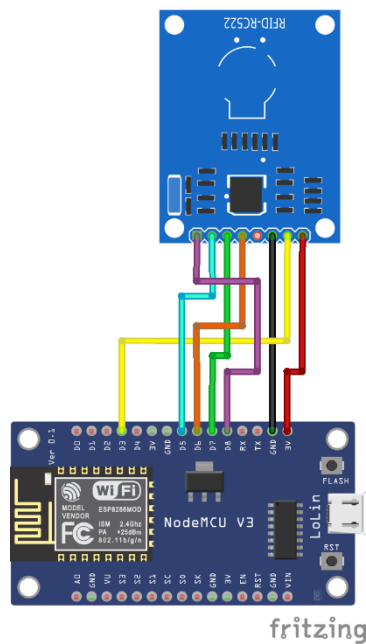
Modulul RFID

RFID este acronimul de la "radio-frequency identification" (identificare prin radiofrecvență) și se referă la o tehnologie prin care datele digitale codificate în etichete RFID sau etichete inteligente sunt captate de un cititor prin intermediul undelor radio. Tehnologia RFID este similară codului de bare, în sensul că datele de pe o etichetă sunt capturate de un dispozitiv care stochează datele într-o bază de date. Cu toate acestea, RFID are mai multe avantaje față de sistemele care utilizează un software de urmărire a activelor cu coduri de bare. Cel mai notabil este faptul că datele din etichetele RFID pot fi citite în afara liniei de vizibilitate, în timp ce codurile de bare trebuie aliniate cu un scanner optic.

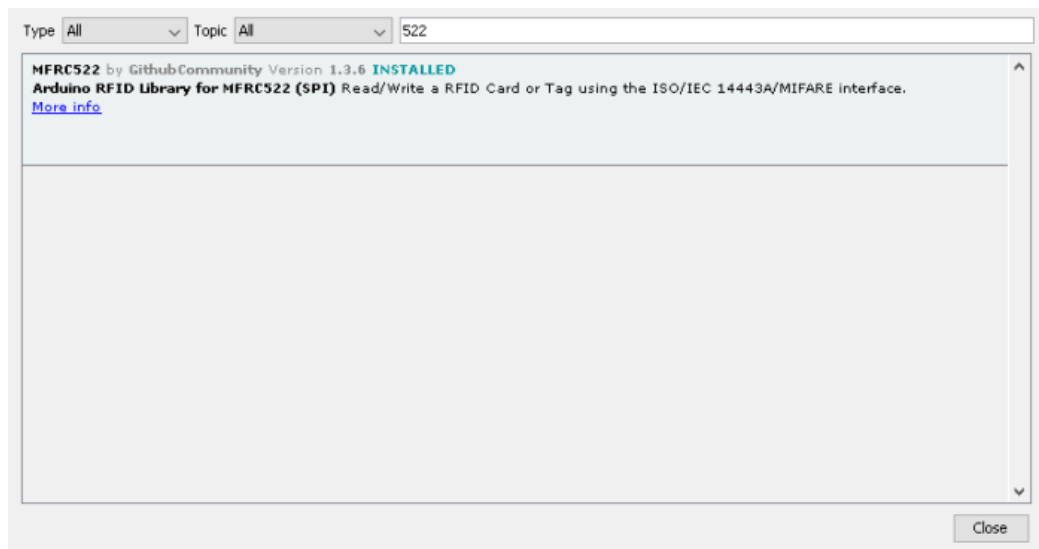
Modul de utilizare este destul de simplu: utilizatorul apropie cardul de scanner și datele sunt citite și prelucrate. În cazul acestui proiect utilizatorul folosește cardul pentru a se înregistra la vendomat, nemaifiind nevoie să introducă bani pentru a achiziționa produse.

Modulul RFID este conectat ca în schema de mai jos la pinii următori:

RC522	ESP8266 NodeMCU
Signal	Pin
RST/Reset	D3
SPI SS	D8
SPI MOSI	D7
SPI MISO	D6
SPI SCK	D5



Pentru a putea folosi modulul RFID trebuie instalată librăria MFRC522 ca în figura de mai jos:



Codul folosit este descris mai jos:

```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN      D3      // Configurable, see typical pin layout above
#define SS_PIN       D8      // Configurable, see typical pin layout above
MFRC522 mfrc522(SS_PIN, RST_PIN); // Create MFRC522 instance

void setup()
{
    Serial.begin(9600);    // Initialize serial communications with the PC
    SPI.begin();           // Init SPI bus
    mfrc522.PCD_Init();    // Init MFRC522
    mfrc522.PCD_DumpVersionToSerial(); // Show details of PCD - MFRC522 Card Reader details
    Serial.println(F("Scan PICC to see UID, SAK, type, and data blocks..."));
}

void loop()
{
    // Look for new cards
    if (!mfrc522.PICC_IsNewCardPresent())
    {
        return;
    }
    // Select one of the cards
    if (!mfrc522.PICC_ReadCardSerial())
    {
        return;
    }
    // Dump debug info about the card; PICC_HaltA() is automatically called
    mfrc522.PICC_DumpToSerial(&(mfrc522.uid));
}
```

După încărcarea acestui cod pe serială ar trebui să apară un mesaj asemănător cu cel de mai jos(mesajul poate să difere în funcție de versiunea moduluului RFID) unde puteți scana cardul.

```

)SMo{^ $){f
wOMsqD$!SUOM$!Md$Firmware Version: 0x91 = v1.0
Scan PICC to see UID, SAK, type, and data blocks...
Card UID: E5 04 C6 AD
Card SAK: 08
PICC type: MIFARE 1KB
Sector Block  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 AccessBits
  15    63  00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      62  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      61  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      60  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
  14    59  00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      58  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      57  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      56  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
  13    55  00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      54  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      53  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      52  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
  12    51  00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      50  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      49  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      48  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
  11    47  00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF [ 0 0 1 ]
      46  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]
      45  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [ 0 0 0 ]

```

☒ Autoscroll No line ending 9600 baud Clear output

Mai departe în implementarea codului care va identifica ID-ul cardului se va folosi „Card UID” .

Stocarea datelor

Din cauza limitărilor spațiului de stocare pe care le au microcontroler-ele ESP8266(8MB RAM), nu este posibilă realizarea unei baze de date care să stocheze utilizatorii, produsele și tranzacțiile și în același timp să execute și comenzile celorlalte componente.În acest caz se folosesc structuri care imită tabelele enumerate mai sus.Acestea sunt realizate într-o clasă numită FileDb.

```
typedef struct
{
    char CNP[12];
    char FirstName[20];
    char LastName[15];
    char Password[20];
    char TagNumber[15];
    float Credit;
    int isAdmin;
    int Enabled;
}User;
```

Această structură reține CNP-ul, nume, prenumele, parola, numărul cardului RFID, creditul, statutul utilizatorului(dacă este sau nu administrator) , iar parametrul Enabled are valoarea 1 dacă acesta este activ și 0 în caz contrar.Celelalte structuri arată similar.

După cum îi spune și numele, funcția „AddUser()” adaugă utilizatori.Aceasta primește ca parametru o entitate de tip User* și verifică dacă CNP-ul utilizatorului care se dorește să fie introdus se află deja în structură.Dacă da, funcția returnează 1, iar în caz contrar stochează datele noului utilizator și crește userCount.Procesul este similar pentru produse și tranzacții.

```
int FileDb::AddUser(User* entity)
{
    bool exists = false;
    for (int i = 0; i < this->userCount; i++)
    {
        if (strcmp(entity->CNP, this->users[i].CNP)==0)
        {
            exists = true;
            return 1;
        }
    }

    this->users[this->userCount] = *entity;
    this->userCount++;
    return 0;
}
```

În funcția „InitData” sunt creați câțiva utilizatori, produse și tranzacții urmând modelul din figurile inserate mai jos.

```
void FileDb::InitData()
{
    User user1;
    strcpy(user1.CNP, "19712073264");
    strcpy(user1.Password, "abc");
    strcpy(user1.FirstName, "Ion");
    strcpy(user1.LastName, "Vasilescu");
    strcpy(user1.TagNumber, "22918314235");
    user1.isAdmin = 1;
    user1.Enabled = 1;
    user1.Credit = 2.40;
    AddUser(&user1);
}
```

```
Product product1;
product1.ID = 1;
strcpy(product1.Name, "Snickers");
product1.Position = 3;
product1.Price = 2.5;
product1.Enabled = 1;
product1.Quantity = 5;
AddProduct(&product1);
```

Aplicația web

Aplicația web este construită folosind HTML, CSS și VueJS. Din cauza spațiului mic de stocare al microcontroler-ului această aplicație web este construită pe o singură pagină.

Primul element care va apărea în pagină este formularul de logare. În acest formular utilizatorul va introduce CNP-ul și parola. Prin apăsare butonului „Login” se va apela funcția „Login()”.

```
<div id="app" v-model="Startup()" v-cloak>
  <!--Login page-->
  <div id="Menu2" v-if="!UserIsLogged">
    <div style="margin: 150px 0px 0px 0px; text-align: center;">
      <h1 style="text-align:center;">Login form</h1>
      <form class="ui form">
        <div class="field">
          <label>Username</label>
          <input type="text" id="username" placeholder="Username">
        </div>
        <div class="field">
          <label>Password</label>
          <input type="text" id="password" placeholder="Password">
        </div>
        <button class="ui button" type="button" v-on:click="Login()"><p>Login</p></button>
      </form>
    </div>
  </div>
</div>
```

```
async Login() {
  var username = document.getElementById("username").value;
  var password = document.getElementById("password").value;
  var url = encodeURI("/LogIn" + "?" + "Username=" + username + "&" + "Password=" + password);
  var result = await fetch(url);
  if (result.status == 200) {
    this.UserInfo.Data = await result.json();
    if (!this.UserInfo.Data[0].isAdmin) {
      this.UserIsLogged = true;
      this.isAdmin = true;
      alert("Welcome back, " + this.UserInfo.Data[0].FirstName + "!");
    } else if (this.UserInfo.Data[0].isAdmin) {
      this.UserIsLogged = true;
      this.isAdmin = false;
      alert("Welcome back, " + this.UserInfo.Data[0].FirstName + "!");
    }
  } else {
    this.UserInfo.Error.Message = await result.json();
    alert(this.UserInfo.Error.Message);
    this.UserIsLogged = false;
    this.isAdmin = false;
  }
},
```

Această funcție este de tip async deoarece așteaptă rezultatul de la server. Datele introduse de utilizator sunt aduse aici prin instrucțiunea „document.getElementById”. Acestea sunt trimise la server și interpretate.

```

int Login()
{
    String cnp = server.arg("Username");
    String pwd = server.arg("Password");
    User* user = fileDb.GetUserByCNP((char*)cnp.c_str());
    if (user != NULL)
    {
        if (strcmp(user->Password, pwd.c_str()) == 0)
        {
            sprintf(MyBuffer, "[{"CNP": \"%s\", \"FirstName\": \"%s\", \"LastName\": \"%s\", \"isAdmin\": %d, \"Credit\": %f, \"TagNumber\": %d, \"isActive\": %d}]",
                user->CNP, user->FirstName, user->LastName, user->isAdmin, user->Credit, user->TagNumber, user->Enabled);
            server.send(200, "application/json", MyBuffer);
        }
        else
        {
            server.send(601, "text/plain", "Invalid CNP or password");
        }
    }
    server.send(602, "text/plain", "User doesn't exist!");
    return 1;
}

```

Această funcție se află pe server și interpretează datele introduse. Acestea sunt preluate din aplicația web prin instrucțiunea „server.arg(„ExNum”)”. Variabila user de tip User* stochează răspunsul funcției „GetElementByCNP” aflată în clasa FileDb.

```

User* FileDb::GetUserByCNP(char* cnp)
{
    for (int i = 0; i < this->userCount; i++) {
        if (strcmp(cnp, this->users[i].CNP) == 0) {
            return &this->users[i];
        }
    }
    return NULL;
}

```

Această funcție verifică dacă parametrul cnp corespunde utilizatorului users[i]. În caz afirmativ returnează datele utilizatorului, iar în caz contrar NULL.

În funcția „Login” se verifică dacă parola utilizatorului primit anterior corespunde cu cea trimisă din aplicația web. În caz afirmativ datele se formatează în format json folosind funcția sprintf și se trimit în pagina web ca răspuns alături de codul 200(ok). În caz contrar se trimite codul de eroare 601 (acest cod este creat de mine și înseamnă „CNP sau parolă invalidă”). În cazul în care nici parola și nici CNP-ul nu se potrivesc cu datele existente deja se trimite codul de eroare 602 (utilizatorul nu există).

Dacă CNP-ul și parola se potrivesc, pe client se primește codul 200(ok) și se verifică dacă utilizatorul este administrator sau nu. În acest afirmativ, variabila de tip boolean isAdmin ia valoarea true, în caz contrar ia valoarea false. În ambele dintre aceste cazuri este afișat mesajul „Welcome back” urmat de numele utilizatorului. În caz că datele introduse nu se potrivesc celor existente în structură se va afișa mesajul venit de la server stocat în UserInfo.ErrorMessage.

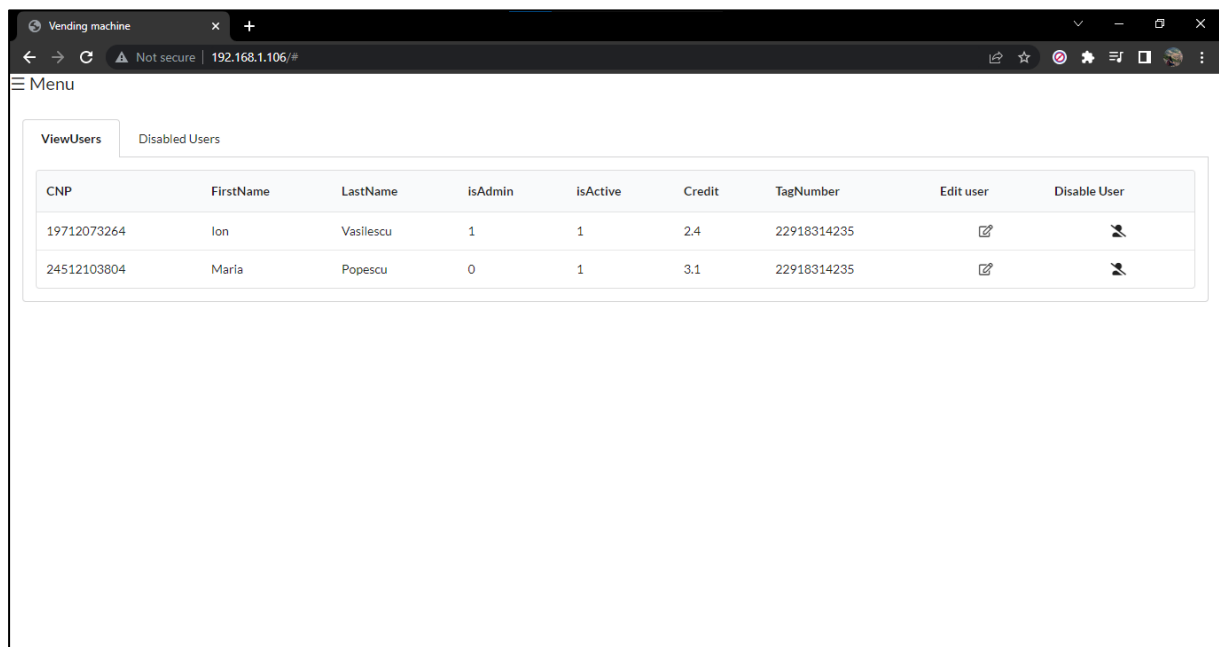
După ce utilizatorul a fost logat variabila de tip boolean UserIsLogged ia valoarea true, iar în funcție de valoarea variabilei isAdmin se afișează meniul administratorului sau al utilizatorului.

Meniul administratorului

Mai jos este inserată o captură de ecran reprezentând structura paginii administratorului. Acesta poate vedea lista utilizatorilor și a produselor, poate adăuga utilizatori și produse, poate deschide ușa vendomatului și nu în ultimul rând, poate testa funcționalitatea motoarelor.



Meniul de vizualizare a utilizatorilor afișează toți utilizatorii și informațiile acestora. În acest meniu administratorul poate edita datele acestora și îi poate dezactiva.



Pentru a face acest lucru posibil, la apăsarea butonului „ViewUsers” se apelează funcția „GetUsers”.

```
async GetUsers() {  
    const responseUsers = await fetch("/Users");  
    this.UserInfo.Users = await responseUsers.json();  
    this.users = this.UserInfo.Users;  
},
```

Această funcție se conectează la funcția Users din server și preia datele tuturor utilizatorilor existenți precum este prezentat în imaginea de mai jos.

```
void Users() {  
    MyBuffer[0] = NULL;  
    TempBuffer[0] = NULL;  
    strcpy(MyBuffer, "");  
    for (int i = 0; i < fileDb.userCount; i++)  
    {  
        Serial.print(MyBuffer);  
        sprintf(TempBuffer, "{\"CNP\": \"%s\", \"FirstName\": \"%s\", \"LastName\": \"%s\", \"isAdmin\": %d, \"Credit\": %f, \"TagNumber\": \"%s\", \"isActive\": %d}\",  
            fileDb.users[i].CNP, fileDb.users[i].FirstName, fileDb.users[i].LastName, fileDb.users[i].isAdmin, fileDb.users[i].Credit, fileDb.users[i].TagNumber,  
            fileDb.users[i].Enabled);  
        if (i < fileDb.userCount - 1)  
        {  
            strcat(MyBuffer, TempBuffer);  
            strcat(MyBuffer, ",");  
        }  
        else  
        {  
            strcat(MyBuffer, TempBuffer);  
        }  
    }  
    strcat(MyBuffer, "];");  
    server.send(200, "application/json", MyBuffer);  
}
```

Informațiile utilizatorilor sunt formate în format json în funcția sprintf în variabila de tip char TempBuffer. După care se pune o virgulă între datele utilizatorilor și se concatenează în variabila de tip char MyBuffer folosind funcția strcat. La final se trimite variabila MyBuffer în aplicația web și codul 200. Acestea sunt afișate pe ecran într-un tabel.

```
<!--If the "ViewUsers" button was pressed this div will be displayed on screen-->  
<div class="ui basic segment" v-if="Menu.ViewUsers">  
    <div id="isClicked">  
        <div class="ui top attached tabular menu">  
            <a class="item active" v-on:click="ShowSubmenu('ViewUsersMenu')" id="ViewUsers">  
                ViewUsers  
            </a>  
            <a class="item" v-on:click="ShowSubmenu('DisableMenu')" id="Disable">  
                Disabled Users  
            </a>  
        </div>  
        <!--View users-->  
        <div class="ui bottom attached segment" v-if="isClicked.ViewUsersMenu">  
            <div class="row">  
                <table class="ui unstackable table">  
                    <thead>  
                        <tr>  
                            <th>CNP</th>  
                            <th>FirstName</th>  
                            <th>LastName</th>  
                            <th>isAdmin</th>  
                            <th>isActive</th>  
                            <th>Credit</th>  
                            <th>TagNumber</th>  
                            <th>Edit user</th>  
                            <th>Disable User</th>  
                        </tr>  
                    </thead>
```

```

<tbody v-for="(user, index) in users">
  <tr>
    <td id="userCNP">{{user.CNP}}</td>
    <td id="FirstName">{{user.FirstName}}</td>
    <td id="LastName">{{user.LastName}}</td>
    <td id="isAdmin">{{user.IsAdmin}}</td>
    <td id="isActive">{{user.IsActive}}</td>
    <td id="Credit">{{user.Credit}}</td>
    <td id="TagNumber">{{user.TagNumber}}</td>
    <td style="text-align:center" id="editUserBtn" v-on:click="EditUserModal(user)">
      <i class="edit outline icon"></i></td>
    <td style="text-align:center" id="disableUser_btn" value="on" v-on:click="DisableUser11(user)">
      <i class="user alternate slash icon"></i></td>
  </tr>
</tbody>
</table>
</div>
</div>

```

Apăsând iconița de edit se va deschide modal-ul de mai jos în care se pot edita informațiile utilizatorului.

The screenshot shows a web application interface. In the background, there is a table with columns: CNP, First Name, Last Name, IsAdmin, IsActive, Credit, and TagNumber. The table contains two rows of data. Overlaid on this is a modal window titled "Edit user's information". The modal contains the following fields and controls:

- First Name: Text input field with the value "Maria".
- Last Name: Text input field with the value "Dumitrescu".
- Password: Text input field with the value "11235".
- Credit: Text input field with the value "2.30" and a dropdown menu showing "RON".
- Admin: A checkbox that is currently unchecked.
- TagNumber: Text input field with the value "123456".
- Buttons: "Edit user" and "Cancel".

Prin apăsarea butonului aceste informații ajung în funcția „EditUser” de unde sunt trimise pe server unde sunt prelucrate. Dacă prelucrarea s-a putut face atunci serverul va returna codul 200, în caz contrar va returna un cod de eroare.

```

async EditUser() {
  var toEditFirstName = document.getElementById("FirstName-modal").value;
  var toEditLastName = document.getElementById("LastName-modal").value;
  var toEditPassword = document.getElementById("Password-modal").value;
  var toEditCredit = document.getElementById("Credit-modal").value;
  var toEditAdmin = document.getElementById("CheckIfAdmin-modal").value;
  if (toEditAdmin.checked == true) {
    toEditAdmin = 1;
  } else {
    toEditAdmin = 0;
  }
  var toEditTagNumber = document.getElementById("TagNumber-modal").value;
  var CNP = this.TempUserEdit.CNP;
  if (toEditFirstName != "" && toEditLastName != "" && toEditCredit != "" && toEditTagNumber != "") {
    var editUrl = encodeURIComponent("/EditUsers" + "?" + "CNP=" + CNP + "&" + "toEditFirstName="
    + toEditFirstName + "&" + "toEditLastName=" + toEditLastName + "&" + "toEditPassword="
    + toEditPassword + "&" + "toEditCredit=" + toEditCredit + "&" + "toEditAdmin=" + toEditAdmin + "&"
    + "toEditTagNumber=" + toEditTagNumber);
    var editResult = await fetch(editUrl);
    var message = await editResult.json();
    if (editResult.status == 200) {
      this.ShowModal("User's info has been updated!");
    }
    else {
      this.ShowModal("Couldn't update user's info!");
    }
  }
  else {
    this.ShowModal("Fill all fields before submitting the form!");
  }
  this.GetUsers();
},

```

Mai jos este inserată o captură de ecran cu un exemplu de date ce pot fi inserate în câmpurile meniului de adăugarea a utilizatorilor.

The screenshot shows a web browser window with the address bar displaying "Not secure | 192.168.1.106/#". The page has a "Menu" button on the left. The main content area is titled "Add user" and contains the following form fields:

- CNP:** 19712073264
- First Name:** Vasile
- Last Name:** Tudor
- Password:** Password123
- TagNumber:** 12364
- Admin:** ☒

At the bottom of the form is a blue button labeled "Add user".

Adăugarea utilizatorilor se face trimițând datele inserate în formularul de mai jos pe server unde se execută funcția „AddUser” prezentată mai sus.

Modul de funcționare al vendomatului

Modul de funcționare de bază:

Clientul introduce monede în vendomat, selectează produsul dorit, iar vendomatul returnează produsul.

Cum funcționează procesul de selecție:

- Butonul a fost apăsător
- Se verifică stocul produsului
- Se verifică dacă creditul inserat de utilizator este suficient pentru produsul selectat
- În cazul în care toate condițiile au fost îndeplinite vendomatul returnează produsul clientului

Vendomatul suportă 3 tipuri de monede(5, 10 și 50 de bani), 4 tipuri de produse din care utilizatorul poate alege și două moduri: standard și company.

În modul standard clientul inserează monede în vendomat, vede creditul introdus pe ecran, selectează produsul dorit , iar vendomatul îl returnează.

Pentru a vedea aplicația web din modul company este necesară conectarea la aceeași rețea ca cea a vendomatului.

În modul company utilizatorul scanează cardul RFID pentru a se autentifica, selectează produsul dorit , iar vendomatul îl returnează.În acest mod de funcționare utilizatorul are un cont la care se poate loga și unde poate vedea tranzacțiile făcute și creditul rămas.

Modul de funcționare a sistemului de logare cu cardul RFID:

- Utilizatorul scanează cardul
- Un mesaj este afișat pe ecran
- După 3 secunde pe ecran este afișat creditul
- Începe procesul de selecție prezentat mai sus
- După terminarea procesului de selecție clientul apasă buton de logout

Modul company vine și cu un cont de administrator unde acesta poate vedea, edita sau adăuga utilizatori și produse și nu în ultimul rând, poate testa funcționalitatea motoarelor.

Bibliografie

Surse folosite în această lucrare:

- Informații despre C++:

Sursă 1: <https://www.toptal.com/c/the-ultimate-list-of-resources-to-learn-c-and-c-plus-plus>

Sursă 2: <https://www.techopedia.com/definition/26184/c-plus-plus-programming-language>

- Despre C++

Sursă: caietul de informatică de clasa a X-a

- Limbajul Arduino:

Sursă 1: <https://flaviocopes.com/arduino-programming-language/>

Sursă 2: <https://www.arduino.cc/en/Guide/Introduction>

Sursă 3: <https://en.wikipedia.org/wiki/Arduino>

- Noțiuni de bază despre NodeMCU ESP8266

Sursă 1: <https://components101.com/development-boards/nodemcu-esp8266-pinout-features-and-datasheet>

Sursă 2: <https://www.make-it.ca/nodemcu-details-specifications/>

- Poză ESP8266

Sursă: https://diyi0t.com/wp-content/uploads/2019/06/NodeMCU_pinout.png

- Informații generale Vue.js

Sursă 1: https://linuxhint.com/about_vue_js/

Sursă 2: <https://www.altexsoft.com/blog/engineering/pros-and-cons-of-vue-js/>

- Poză adrese PCF8574

Sursă: <https://www.instructables.com/PCF8574-GPIO-Extender-With-Arduino-and-NodeMCU/>

- Poză conectarea pinilor modulului RFID la ESP8266:

Sursă: <http://www.esp8266learning.com/esp8266-rfid-rc522-module-example.php>