# AquaBreeze

## Introducere

**Nume: Comisaru Gabriel-Cornel**
**Grupa: 331CB**

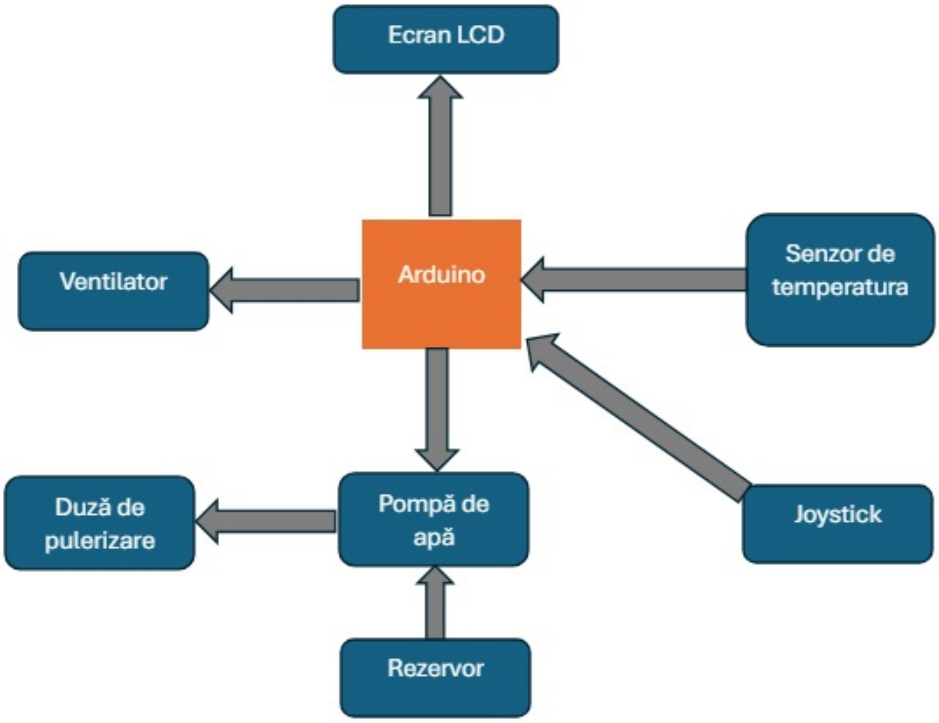
Înfruntând căldurile sufocante ale căminului și amenințările iminente de incendiu, AquaBreeze este răspunsul ingenios la climatul instabil din interior. Cu un singur scop: să aducă răcoare și siguranță în spațiile de viață, acest proiect revoluționar transformă căldura opresivă într-o adiere revigorantă de prospețime. Cu tehnologie de vârf și acțiuni rapide, AquaBreeze este soluția la incendiile și disconfortul termic, oferind un mediu confortabil și sigur pentru fiecare locuitor al căminului.

## Descriere generală

Modul de functionare al automatului va fi urmatorul:

- Senzorul de temperatura masoara valoarea temperaturii din incapere la fiecare moment de timp
- Informatia este procesata si se afiseaza pe un ecran LCD temperatura curenta
- Cand temperatura trece de un anumit threshold setat anterior utilizand joystick-ul, ventilatorul si pompa de apa vor porni
- Pompa ia apa din rezervor si o duce prin furtun pana la duza si stropeste cu apa prin intermediul acesteia

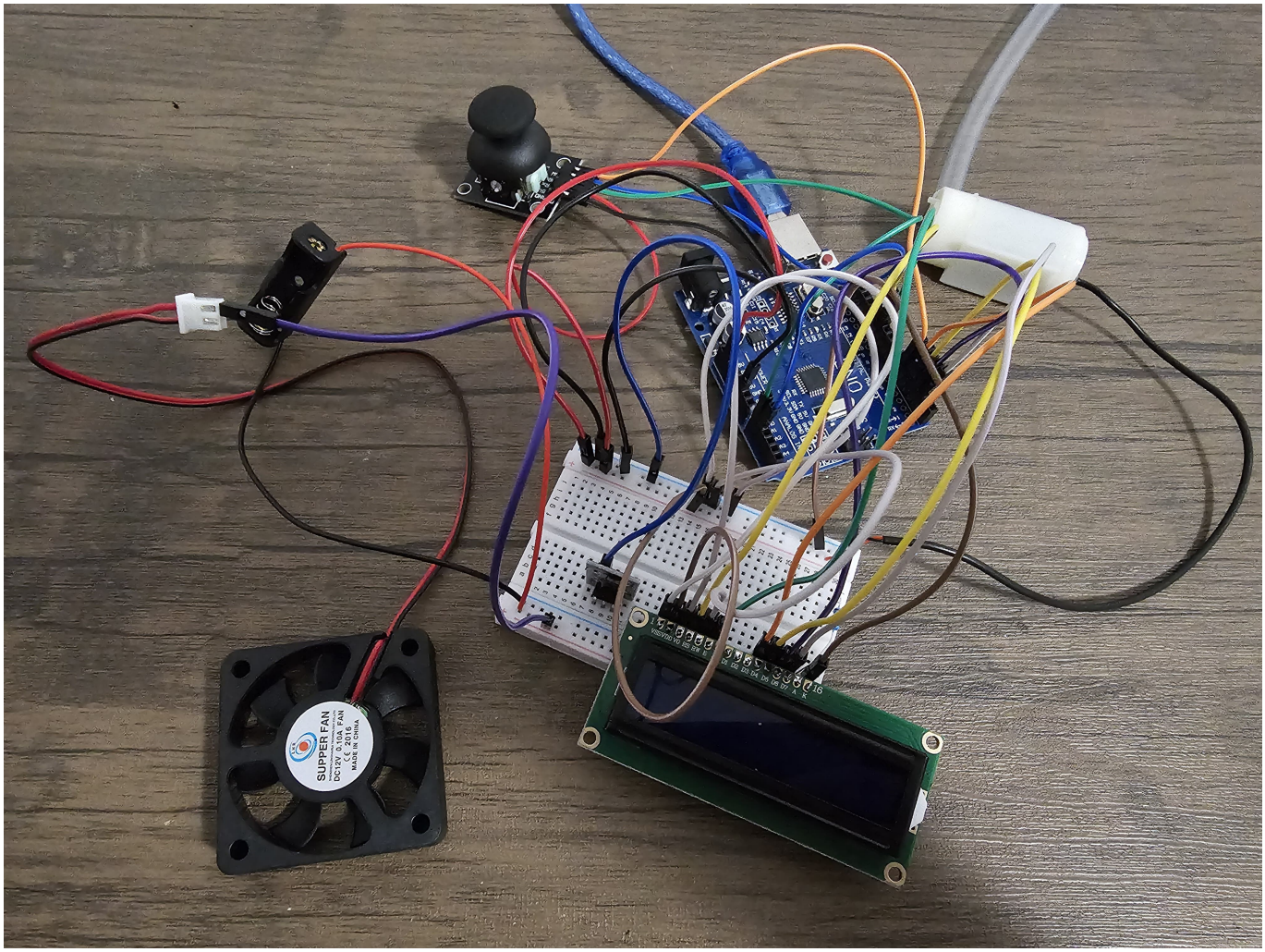
### Schema Bloc



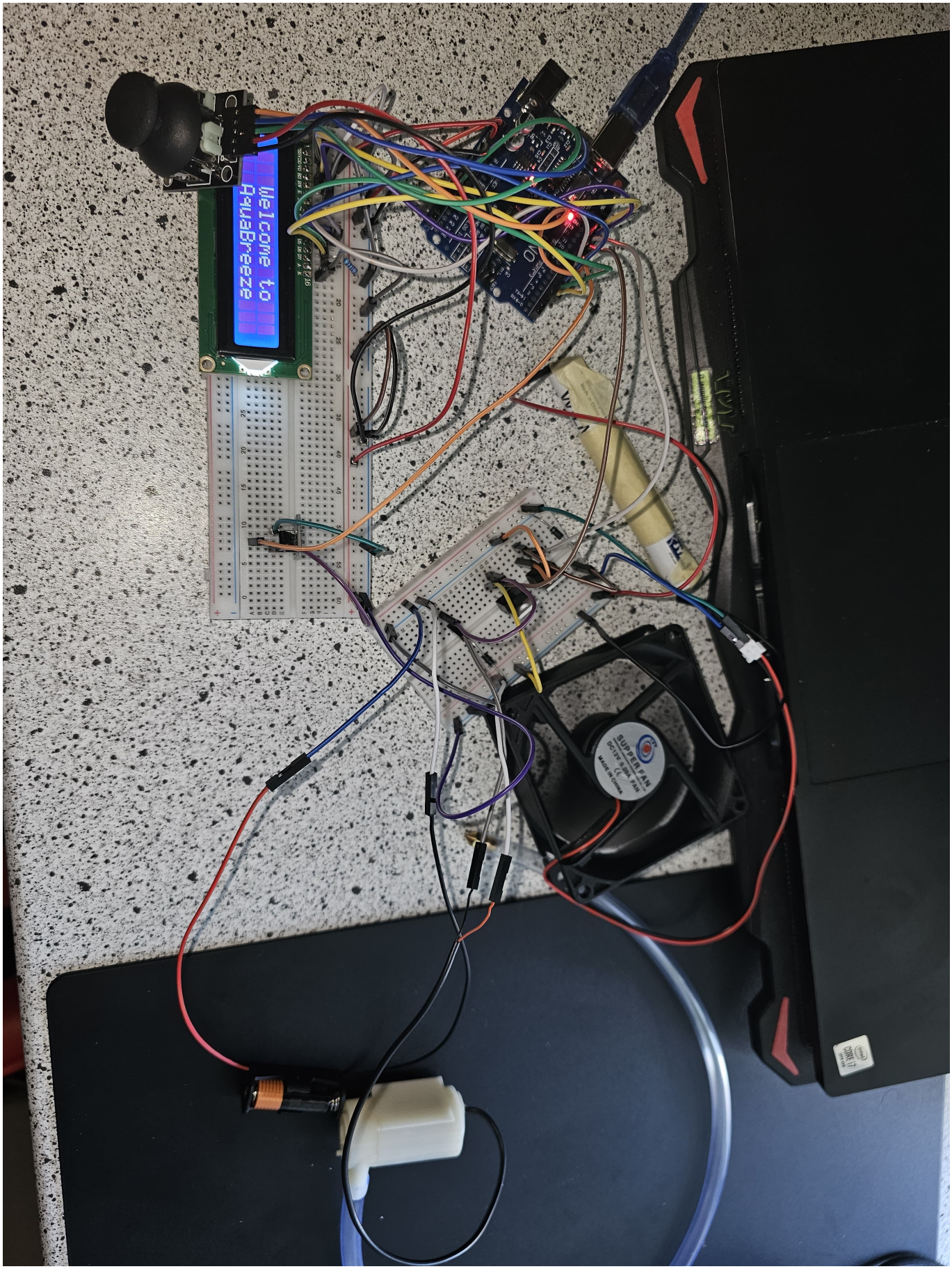

# Hardware Design

Lista de piese:

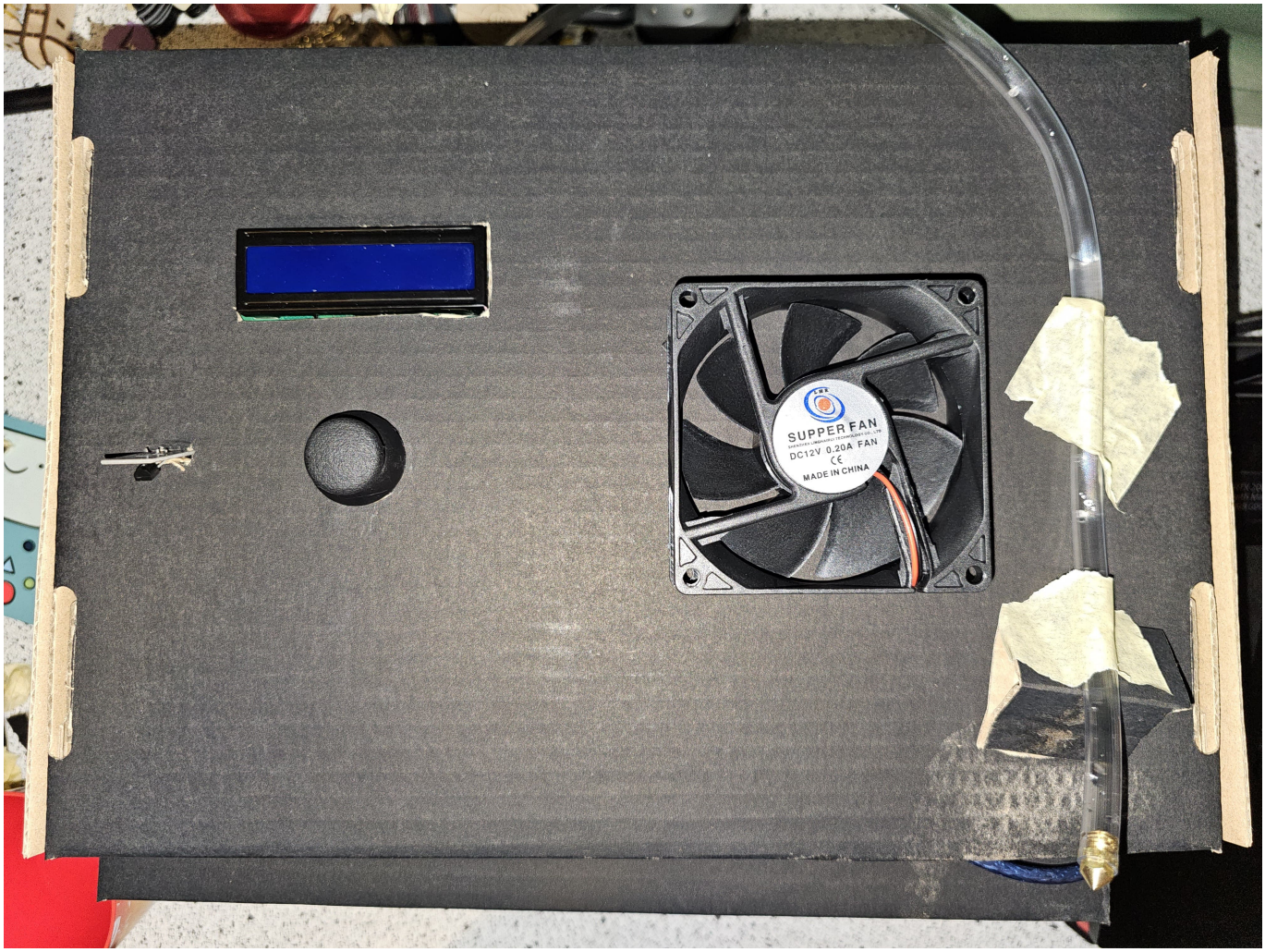
1. placa Arduino
2. senzor de temperatură
3. pompă de apă
4. rezervor de apă
5. duză de pulverizare
6. tranzistori & rezistoare
7. diode
8. conectori & fire
9. sursă de alimentare
10. display lcd cu meniu
11. joystick
12. ventilator




Updated:




Final product:




# Software Design

**Mediu de dezvoltare:** Arduino IDE
**Versiunea:** 1.8.19

**Librarii si surse 3rd-Party:**

1. LiquidCrystal.h
   - Biblioteca este utilizata pentru a controla ecranul LCD oferind unfctii simple pentru a afisa text pe un LCD cu 16×2 caractere.
   - #include <LiquidCrystal.h>
2. DallasTemperature.h
   - Biblioteca folosita pentru citirea temperaturii de la senzorul de temperatura.
   - #include <DallasTemperature.h>
3. OneWire.h
   - Biblioteca folosita alaturi de cea Dallas pentru comunicarea cu senzorul de temperatura.
   - #include <OneWire.h>
4. TimerOne.h
   - Aceasta biblioteca permite utilizarea timerului 1 de pe microcontroller pentru a seta intreruperi temporizate. Am utilizat-o pentru a verifica temperatura la intervale regulate.
   - #include <TimerOne.h>

**Algoritmi si structuri pe care le-am implementat:**

1. Meniul interactiv
   - Meniul interactiv este controlat printr-un joystick. Structura meniului permite utilizatorului sa navigheze intre optiuni si sa selecteze diverse actiuni. Exista doua optiuni prinicpale:
     1. Afisarea temperaturii curente: Afiseaza temperatura citita de la senzor.
     2. Schimbarea temperaturii de prag: Permite utilizatorului sa seteze o noua valoare de preag pentru temperatura.
   - Functii:
     1. updateMenu() Actualizeaza meniul pe ecranul LCD in functie de selectiile utilizatorului.
     2. menuSelection() Gestioneaza actiunile efectuate atunci cand utilizatorul selecteaza o optiune din meniu.
2. Controlul pompei si al ventilatorului

Pompa si ventilatorul sunt controlate in functie de temperatura citita de la senzor.

- Daca temperatura depaseste pragul setat, ventilatorul este pornit si pompa este activata pentru o secunda, urmata de o perioada de pauza de 5 secunda, dupa care este pornita din nou s.a.m.d.
- Ventilatorul ramane pornit atata timp cat temperatura este peste pragul setat.

Functii:

- timerISR() Seteaza un flag la intervale de 5 secunde pentru a indica ca este timpul sa se verifice temperatura.
- loop() Codul din loop() gestioneaza activarea si dezactivarea pompei si ventilatorului.

**Surse si functii implementate:**

```
setup()
```

Initializarea LCD-ului, senzorului si configurarea piniilor pentru joystick, pompa si ventilator si a temporizatorului pentru verificarea temperaturii:

setup.c

```
// set up the LCD's number of columns and rows:
lcd.begin(16, 2);
sensors.begin();
Serial.begin(9600); // Starting Serial for Joystick
lcd.setCursor(3,0);

// Print the welcome message to the LCD.
lcd.print("Welcome to");
```

```
  lcd.setCursor(3,1);
  lcd.print("AquaBreeze");

  // Initial position of the JoyStick
  xInit = analogRead(joyX);
  yInit = analogRead(joyY);

  pinMode(joyButton, INPUT_PULLUP);
  pinMode(pumpPin, OUTPUT);
  pinMode(fanPin, OUTPUT);

  delay(5000);

  // Check the temp every 5s
  Timer1.initialize(5000000);
  Timer1.attachInterrupt(timerISR);

  updateMenu();
```

```
loop()
```

Citirea valorilor de la joystick pentru navigarea in meniu:

joystick.c

```
int xValue = analogRead(joyX);
int yValue = analogRead(joyY);
int joyButtonState = digitalRead(joyButton);
```

Actualizarea afisajului LCD.

lcd.c

```
f (yValue < yInit - DRIFT) {
      menuIndex = 0;
      updateMenu();
      delay(300);
    } else if (yValue > yInit + DRIFT) {
      menuIndex = 1;
      updateMenu();
      delay(300);
    }
}
```

Verificarea si actualizarea starii pompei si ventilatorului in functie de temperatura citita de la senzor.

motors.c

```
if (temp > tempThreshold) {
    unsigned long currentTime = millis();
    digitalWrite(fanPin, HIGH);
    if (!pumpActive && (currentTime - lastPumpTime >= pumpCooldown)) {
        digitalWrite(pumpPin, HIGH);
        delay(pumpActiveTime);
        digitalWrite(pumpPin, LOW);
        lastPumpTime = currentTime;
    }
} else {
        digitalWrite(fanPin, LOW);
}
```

```
menuSelection()
```

Afisarea temperaturii curente.

currTemp.c

```
lcd.setCursor(0,0);
lcd.print("Current Temp:");
lcd.setCursor(0,1);
lcd.print(temp);
lcd.print(" C");
```

Afisarea schimbarii temperaturii de prag de catre utilizator.

changeTemp.c

```
lcd.setCursor(0,0);
lcd.print("Set new temp:");
lcd.setCursor(0,1);
lcd.print("< ");
lcd.print(tempThreshold);
lcd.print(" C >");
```

# Download

aquabreeze.zip

# Rezultate Obţinute

Video cu demo: https://www.youtube.com/watch?v=FfeswgPBJQo&ab_channel=GabrielComisaru

# Concluzii

Per total, proiectul a fost un succes, am implementat tot ce mi-am propus. Au existat destule probleme întâmpinate, precum gestionarea consumului de energie, având în vedere numărul de componente conectate la Arduino. Inițial, când conectam senzorul, joystick-ul, LCD-ul și pompa, Arduino-ul era suprasolicitat și se reseta de fiecare dată când se pornea pompa . A fost nevoie să folosesc o alimentare externă, dar pompa avea nevoie de 3-6V și nu am găsit astfel de baterii, așa că am conectat în serie două baterii de 1.5V folosind bandă adezivă . Din fericire, această improvizație a funcționat și am reușit să duc la bun sfârșit proiectul.

Din acest lucru am învățat că pe viitor trebuie să fiu mai atent când îmi comand piesele și să mă gândesc mai bine dacă Arduino-ul este suficient ca sursă de alimentare pentru toate componentele pe care le voi avea.

# Jurnal

05.05.2024 - Creare pagina wiki
12.05.2024 - Comanda piese
16.05.2024 - Compunere schema bloc
17.05.2024 - Adaugare schema hardware + poze
24.05.2024 - Adaugare implementare software
25.05.2024 - Rezultatele obtinute printr-un video

# Bibliografie/Resurse

https://www.sensingthecity.com/tutorial-controlling-submersible-pump-with-arduino/
https://docs.arduino.cc/built-in-examples/usb/JoystickMouseControl/
https://www.youtube.com/watch?v=s_-nIgo71_w&ab_channel=ScienceBuddies

https://www.arduino.cc/reference/en/libraries/liquidcrystal/
https://www.arduino.cc/reference/en/libraries/dallastemperature/
https://github.com/sparkfun/SparkFun-Eagle-Libraries

Export to PDF