Universidade Presbiteriana Mackenzie

Universidade Presbiteriana Mackenzie

Projeto Objetos Inteligentes Conectados

Automatização de produção de cerveja artesanal

Alunos:

Gustavo Trindade de Avila - 31515649
Joao Ailton Junior da Silva
Diego Muniz Sobrinho

Turma: 06J

O projeto consiste em automatizar o processo produção de cerveja artesanal através de um aplicativo chamado Blynk, as funcionalidades do aplicativo são:

- Mexer Mosto– aciona um dos motores DC para mexer a pá no balde.
- Liberar Malte – aciona outro motor DC que gira a broca para liberar o malte.
- Verificar Temperatura – apresenta a temperatura identificada pelo sensor de temperatura DS18B20.

O aplicativo irá fazer comunicação com o Arduino UNO para poder executar os comandos listados acima.

Objetivos:

Automatizar a produção de cerveja artesanal, permitindo o produtor fazer a cerveja de uma maneira mais rápida, menos cansativa, podendo produzir ainda mais cerveja de uma maneira inovadora.

Problema:

- Descrição do problema:

Atualmente o processo de produção de cerveja artesanal é um processo que leva muito tempo para ser concluído além de possuir algumas partes do processo que demandam uma segunda pessoa para auxiliar na produção, e também em alguns momentos é processo cansativo por exigir do produtor ficar mexendo constantemente o malte.

- Quem é afetado pelo problema:

Produtores de cerveja artesanal.

- Benefícios de uma boa solução:

Agilizar o processo de produção de cerveja, torna-lo mais simples e sem a necessidade de uma segunda pessoa além do produtor, além de fazer o processo menos cansativo, já que o produtor não terá necessidade de ficar mexendo o malte, atividade que será exercida por um dos motores DC.

Plataformas de Desenvolvimento: **Fritzing e Arduino IDE**

Imagem Esquemático

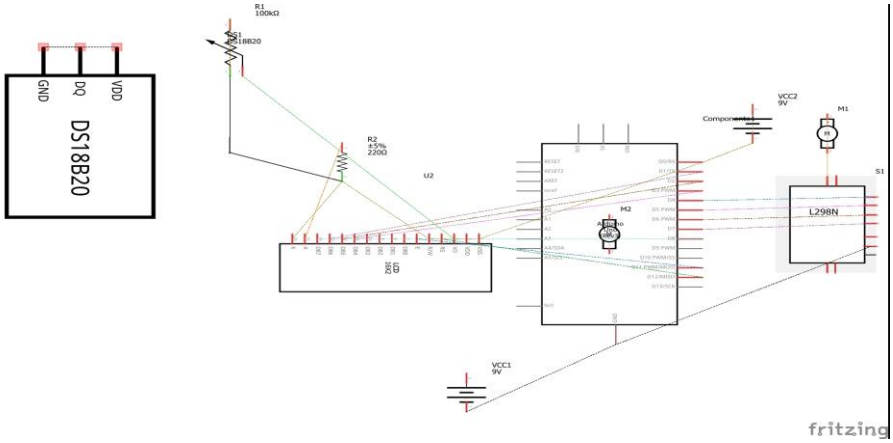
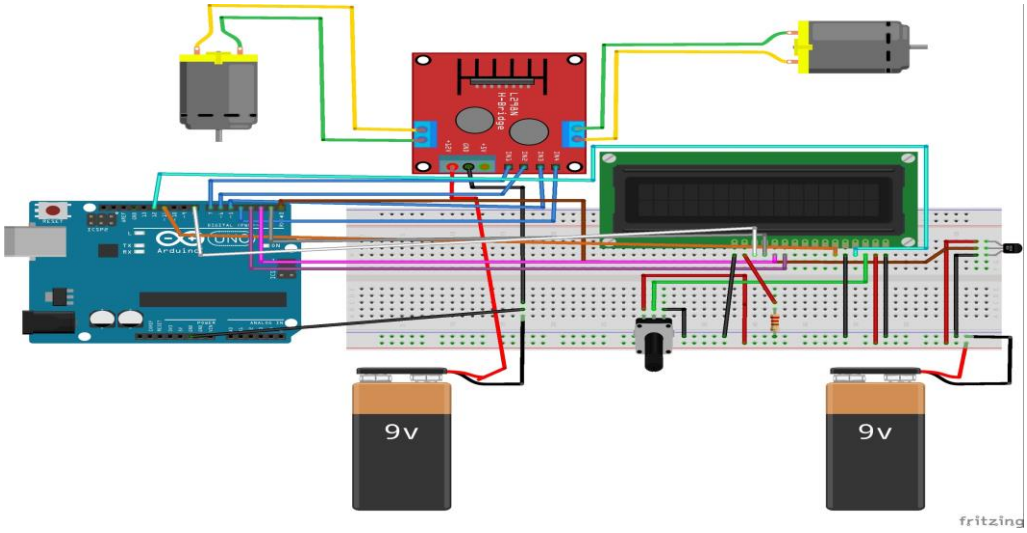
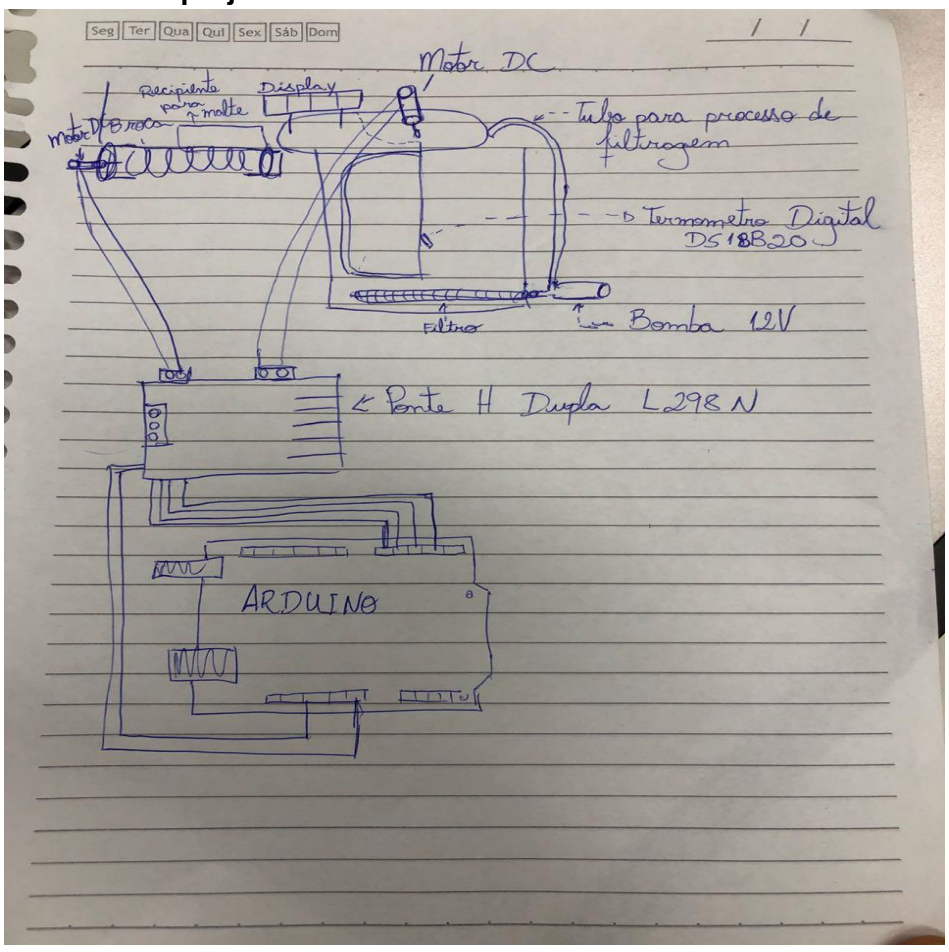


Imagem Protoboard.



Desenho do projeto:



Materiais utilizados no projeto:

Sensores:
1 Termômetro Digital DS18B20

Atuadores:
2x Motor DC

Displays:
1 Display LCD

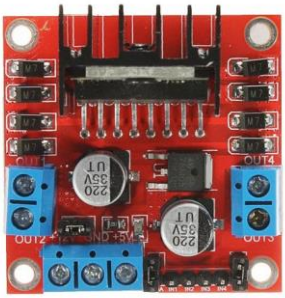
Arduino:
1 Arduíno UNO

Outros:
1 Bomba 12V
1 Driver Ponte H Dupla L298N
1 Broca
1 Tubo PVC (1M)
1 Mangueira (1,5 M)
1 Balde de Plástico (5L)

Imagens Equipamentos

Motor DC

Termômetro Digital DS18B20 e Display LCD

Driver Ponte H Dupla L298N 

Arduíno UNO 

Balde de Plástico (5L) 

Mangueira (1,5 M)

Garrafa Pet (500ml)

Broca

Código do projeto:

```
/*************************************************************
  Download latest Blynk library here:
    https://github.com/blynkkk/blynk-library/releases/latest

  Blynk is a platform with iOS and Android apps to control
  Arduino, Raspberry Pi and the likes over the Internet.
  You can easily build graphic interfaces for all your
  projects by simply dragging and dropping widgets.

    Downloads, docs, tutorials: http://www.blynk.cc
    Sketch generator:           http://examples.blynk.cc
    Blynk community:            http://community.blynk.cc
    Follow us:                  http://www.fb.com/blynkapp
                                http://twitter.com/blynk_app

  Blynk library is licensed under MIT license
```

```
  This example code is in public domain.

 *************************************************************
  =>
  =>         USB HOWTO: http://tiny.cc/BlynkUSB
  =>

  You can send/receive any data using WidgetTerminal object.

  App project setup:
    Terminal widget attached to Virtual Pin V1
 *************************************************************/


/* Comment this out to disable prints and save space */
#define BLYNK_PRINT SwSerial

#include <Wire.h>
#include <OneWire.h>
#include <DallasTemperature.h>

// Import required libraries
#define ONE_WIRE_BUS 3

// Setup a oneWire instance to communicate with any OneWire devices (not just
Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);
// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);  // Set the LCD I2C address


#include <LiquidCrystal_I2C.h>
int IN1 = 11;
int IN2 = 6;
int IN3 = 5;
int IN4 = 10;
float temp;

#include <SoftwareSerial.h>
SoftwareSerial SwSerial(10, 11); // RX, TX

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "e9eef5c379d04bf5801021b17e04dfc3";

// Attach virtual serial terminal to Virtual Pin V1
WidgetTerminal terminal(V3);
```

```
// You can send commands from Terminal to your hardware. Just use
// the same Virtual Pin as your Terminal Widget
BLYNK_WRITE(V3)
{

  // if you type "Marco" into Terminal Widget - it will respond: "Polo:"
  if (String("Marco") == param.asStr()) {
    terminal.println("You said: 'Marco'") ;
    terminal.println("I said: 'Polo'") ;
  } else {

    // Send it back
    terminal.print("You said:");
    terminal.write(param.getBuffer(), param.getLength());
    terminal.println();
  }

  // Ensure everything is sent
  terminal.flush();
}



void setup()
{

  //Define os pinos como saida
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);

  // start serial port
  //  Serial.begin(9600);
  //  Serial.println("Dallas Temperature IC Control Library Demo");
  // Start up the library
  sensors.begin();
  lcd.begin(16,2);
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(0,0);

 Serial.begin(9600);
 Blynk.begin(auth);

}
```

```
void loop() {

    // put your main code here, to run repeatedly:
    Blynk.run();


   //Gira o Motor A no sentido horario
   digitalWrite(IN1, HIGH);
   digitalWrite(IN2, LOW);
   delay(2000);
   //Para o motor A
   digitalWrite(IN1, HIGH);
   digitalWrite(IN2, HIGH);
   delay(500);

   //Gira o Motor B no sentido horario
   digitalWrite(IN3, HIGH);
   digitalWrite(IN4, LOW);
   delay(2000);
   //Para o motor B
   digitalWrite(IN3, HIGH);
   digitalWrite(IN4, HIGH);
   delay(500);

     sensors.requestTemperatures(); // Send the command to get temperatures
     // Serial.print("Temperature for the device 1 (index 0) is: ");
     float x = sensors.getTempCByIndex(0);
     //  Serial.println(x);
     lcd.setCursor(0,0);
     lcd.print("Temperature: ");
     lcd.setCursor(0,1);
     lcd.print("          ");
     lcd.setCursor(0,1);
     lcd.print(x);

    delay(3000);

}
```