



Pontifícia Universidade Católica de Minas Gerais

Instituto de Ciências Exatas e Informática

Matéria: Laboratório de Introdução a Programação

Atividade: Trabalho final – Arduino

Link do projeto: <https://www.tinkercad.com/things/iKMEhKwT0R3-v4arduino-projeto-final>

Nomes: Davi Cândido de Almeida

Mateus Martins Parreiras

Vitor Leite Setragni

Nome do projeto: Sistema de automação para lojas.

Introdução:

O projeto “Sistema de automação para lojas” foi desenvolvido como parte do trabalho final em arduino para demonstrar nossas habilidades em programação e montagem de circuitos. Esse relatório descreve o processo de criação do projeto, os elementos utilizados e como funciona o projeto.

1 - Descrição do projeto:

O projeto consiste em um sistema de automação para lojas cujo objetivo é facilitar a vida tanto dos atendentes, quanto dos clientes, fazendo com que fique mais dinâmica a interação do cliente com a própria loja, necessitando apenas de um atendente no caixa para finalizar as “compras”.

2 – Elementos utilizados:

Os elementos utilizados no nosso projeto foram: Um sensor de distância, um buzzer, um led, um servo motor, um monitor LCD, uma placa arduino uno, uma protoboard e um resistor.

3 - Motivação

Nossa principal motivação para criação do projeto foi devido a uma má experiência que algumas pessoas já passaram com lojas que não facilitam a vida do cliente, fazendo com que o cliente não se sinta acolhido e consequentemente não volte na loja.

4 – Funcionamento do projeto:

Para o sistema de automação da nossa loja, utilizamos um sensor de distância que facilita a experiência do cliente desde o momento em que ele se aproxima da porta. Assim que o cliente chega perto da entrada, o sensor é acionado e a porta se abre automaticamente. Em seguida, as luzes internas se acendem, proporcionando um ambiente acolhedor sem que o cliente precise fazer qualquer esforço.

Ao entrar, o cliente é recebido por uma tela de boas-vindas. Utilizamos a tela LCD que exibe uma mensagem de Boas-Vindas e realiza uma contagem regressiva de 5 segundos antes de fechar a porta e apagar a mensagem. Logo após, o buzzer é ativado, tocando uma música agradável para criar uma atmosfera mais confortável.

Durante a permanência do cliente na loja, o sistema permanece ativo, garantindo uma experiência agradável. Quando o cliente se dirige à saída e aciona novamente o sensor de distância, a porta se abre, exibindo uma mensagem de despedida com "Volte sempre", ao mesmo tempo em que as luzes se apagam e a música é interrompida, finalizando o ciclo de atendimento.

Este sistema de automação não só melhora a eficiência energética da loja, como também enriquece a experiência do cliente, proporcionando um ambiente inteligente e acolhedor.

5 - Código do projeto

```
// Inclui bibliotecas para música e funcionalidades
```

```
//Star Wars theme
```

```
// Define as frequências das notas musicais em Hz
```

```
#define NOTE_B0 31
```

```
#define NOTE_C1 33
```

```
#define NOTE_CS1 35
```

```
#define NOTE_D1 37
```

```
#define NOTE_DS1 39
```

```
#define NOTE_E1 41
```

#define NOTE_F1 44	#define NOTE_GS3 208
#define NOTE_FS1 46	#define NOTE_A3 220
#define NOTE_G1 49	#define NOTE_AS3 233
#define NOTE_GS1 52	#define NOTE_B3 247
#define NOTE_A1 55	#define NOTE_C4 262
#define NOTE_AS1 58	#define NOTE_CS4 277
#define NOTE_B1 62	#define NOTE_D4 294
#define NOTE_C2 65	#define NOTE_DS4 311
#define NOTE_CS2 69	#define NOTE_E4 330
#define NOTE_D2 73	#define NOTE_F4 349
#define NOTE_DS2 78	#define NOTE_FS4 370
#define NOTE_E2 82	#define NOTE_G4 392
#define NOTE_F2 87	#define NOTE_GS4 415
#define NOTE_FS2 93	#define NOTE_A4 440
#define NOTE_G2 98	#define NOTE_AS4 466
#define NOTE_GS2 104	#define NOTE_B4 494
#define NOTE_A2 110	#define NOTE_C5 523
#define NOTE_AS2 117	#define NOTE_CS5 554
#define NOTE_B2 123	#define NOTE_D5 587
#define NOTE_C3 131	#define NOTE_DS5 622
#define NOTE_CS3 139	#define NOTE_E5 659
#define NOTE_D3 147	#define NOTE_F5 698
#define NOTE_DS3 156	#define NOTE_FS5 740
#define NOTE_E3 165	#define NOTE_G5 784
#define NOTE_F3 175	#define NOTE_GS5 831
#define NOTE_FS3 185	#define NOTE_A5 880
#define NOTE_G3 196	#define NOTE_AS5 932

```

#define NOTE_B5 988

#define NOTE_C6 1047

#define NOTE_CS6 1109

#define NOTE_D6 1175

#define NOTE_DS6 1245

#define NOTE_E6 1319

#define NOTE_F6 1397

#define NOTE_FS6 1480

#define NOTE_G6 1568

#define NOTE_GS6 1661

#define NOTE_A6 1760

#define NOTE_AS6 1865

#define NOTE_B6 1976

#define NOTE_C7 2093

#define NOTE_CS7 2217

#define NOTE_D7 2349

#define NOTE_DS7 2489

#define NOTE_E7 2637

#define NOTE_F7 2794

#define NOTE_FS7 2960

#define NOTE_G7 3136

#define NOTE_GS7 3322

#define NOTE_A7 3520

#define NOTE_AS7 3729

#define NOTE_B7 3951

#define NOTE_C8 4186

#define NOTE_CS8 4435

```

```

#define NOTE_D8 4699

#define NOTE_DS8 4978

#define REST 0

// Define o tempo da música em BPM

int tempo = 108;

// Define o pino do buzzer

int buzzer = 3;

// Notas da melodia seguidas pela duração

int melody[] = {

    // Tema de Darth Vader (Marcha Imperial) -
    // Star Wars

    NOTE_AS4,8, NOTE_AS4,8, NOTE_AS4,8,
    //1

    NOTE_F5,2, NOTE_C6,2,

    NOTE_AS5,8, NOTE_A5,8, NOTE_G5,8,
    NOTE_F6,2, NOTE_C6,4,

    NOTE_AS5,8, NOTE_A5,8, NOTE_G5,8,
    NOTE_F6,2, NOTE_C6,4,

    NOTE_AS5,8, NOTE_A5,8, NOTE_AS5,8,
    NOTE_G5,2, NOTE_C5,8, NOTE_C5,8,
    NOTE_C5,8,

    NOTE_F5,2, NOTE_C6,2,

    NOTE_AS5,8, NOTE_A5,8, NOTE_G5,8,
    NOTE_F6,2, NOTE_C6,4,

    NOTE_AS5,8, NOTE_A5,8, NOTE_G5,8,
    NOTE_F6,2, NOTE_C6,4, //8

    NOTE_AS5,8, NOTE_A5,8, NOTE_AS5,8,
    NOTE_G5,2, NOTE_C5,-8, NOTE_C5,16,

```

```

    NOTE_D5,-4, NOTE_D5,8, NOTE_AS5,8,
    NOTE_A5,8, NOTE_G5,8, NOTE_F5,8,

    NOTE_F5,8, NOTE_G5,8, NOTE_A5,8,
    NOTE_G5,4, NOTE_D5,8, NOTE_E5,4,
    NOTE_C5,-8, NOTE_C5,16,

    NOTE_D5,-4, NOTE_D5,8, NOTE_AS5,8,
    NOTE_A5,8, NOTE_G5,8, NOTE_F5,8,

    NOTE_C6,-8, NOTE_G5,16, NOTE_G5,2,
    REST,8, NOTE_C5,8, //13

    NOTE_D5,-4, NOTE_D5,8, NOTE_AS5,8,
    NOTE_A5,8, NOTE_G5,8, NOTE_F5,8,

    NOTE_F5,8, NOTE_G5,8, NOTE_A5,8,
    NOTE_G5,4, NOTE_D5,8, NOTE_E5,4,
    NOTE_C6,-8, NOTE_C6,16,

    NOTE_F6,4, NOTE_DS6,8, NOTE_CS6,4,
    NOTE_C6,8, NOTE_AS5,4, NOTE_GS5,8,
    NOTE_G5,4, NOTE_F5,8,

    NOTE_C6,1

};

// Calcula o número de notas na melodia

int notes = sizeof(melody) / sizeof(melody[0]) /
2;

// Calcula a duração de uma nota inteira em
ms

int wholenote = (60000 * 4) / tempo;

int divider = 0, noteDuration = 0;

// Fim das definições musicais

// Inclusão de bibliotecas nativas

#include <Adafruit_LiquidCrystal.h>

#include <Servo.h>

// Definição de variáveis globais

Adafruit_LiquidCrystal lcd_1(0); // Inicializa o
display LCD

```

```

Servo servo; // Inicializa o objeto servo

const int pin_LED = 11; // Define o pino do
LED

int distancia = 0; // Variável para armazenar a
distância medida

int seconds = 5; // Variável para contagem
regressiva

int pos = 0; // Variável para posição do servo

bool ledLigado = false; // Estado do LED

bool musicalLigada = false; // Estado da
música

bool foiDetectado = false; // Marca se a
distância foi detectada

bool PessoaEntando = false; // Marca se há
uma pessoa entrando

bool entrando = true; // Marca se é a primeira
vez que a pessoa está entrando

// Configurações iniciais

void setup() {

    servo.attach(9, 500, 2500); // Anexa o servo
ao pino 9 com limites de pulso

    pinMode(pin_LED, OUTPUT); // Configura o
pino do LED como saída

    Serial.begin(9600); // Inicia a comunicação
serial

}

// Função para testar a distância

void testaDistancia() {

    distancia = 0.01723 *
readUltrasonicDistance(6, 5);

    // Verifica se a distância é menor que 50 cm

    if (distancia < 50) {

```

```

    if (!foiDetectado) {

        ledLigado = !ledLigado; // Inverte o estado
do LED

        musicaLigada = !musicaLigada; // Inverte o
estado da música

        foiDetectado = true; // Marca que a
distância foi detectada

        PessoaEntando = !PessoaEntando; //
Alterna o estado de PessoaEntando

        delay(2000); // Pequeno atraso para evitar
múltiplas detecções consecutivas

    }

    } else {

        foiDetectado = false; // Reseta a marcação
quando a distância não é detectada

    }

    Serial.println(distancia); // Imprime a
distância no Serial Monitor

    // Define o estado do LED com base na
variável ledLigado

    if (ledLigado) {

        digitalWrite(pin_LED, HIGH); // Liga o LED

    } else {

        digitalWrite(pin_LED, LOW); // Desliga o
LED

    }

    // Se há uma pessoa entrando, abre a porta,
exibe a mensagem e depois fecha a porta

    if (PessoaEntando) {

        AbrePorta();

        Mensagem();

```

```

        FechaPorta();

    }

}

// Função para tocar a música

void tocaMusica() {

    // Itera sobre as notas da melodia

    for (int thisNote = 0; thisNote < notes * 2 &&
musicaLigada; thisNote = thisNote + 2) {

        testaDistancia(); // Testa a distância durante a
reprodução da música

        // Calcula a duração de cada nota

        divider = melody[thisNote + 1];

        if (divider > 0) {

            noteDuration = (wholenote) / divider; //
Nota regular

        } else if (divider < 0) {

            noteDuration = (wholenote) / abs(divider);
// Nota pontuada

            noteDuration *= 1.5; // Aumenta a duração
em metade

        }

        // Toca a nota por 90% da duração,
deixando 10% como pausa

        tone(buzzer, melody[thisNote], noteDuration
* 0.9);

        delay(noteDuration); // Espera a duração
especificada antes de tocar a próxima nota

        noTone(buzzer); // Para a geração da onda
sonora antes da próxima nota

    }

}

```

// Função que simula biblioteca para ler a distância usando o sensor ultrassônico

```
long readUltrasonicDistance(int triggerPin, int echoPin) {
```

```
    pinMode(triggerPin, OUTPUT); // Configura o pino de disparo como saída
```

```
    digitalWrite(triggerPin, LOW); // Limpa o pino de disparo
```

```
    digitalWrite(triggerPin, HIGH); // Envia um pulso
```

```
    digitalWrite(triggerPin, LOW); // Limpa o pino de disparo
```

```
    pinMode(echoPin, INPUT); // Configura o pino de eco como entrada
```

```
    return pulseIn(echoPin, HIGH); // Lê o pino de eco e retorna o tempo de viagem da onda sonora em microssegundos
```

```
}
```

// Função para exibir mensagens no LCD

```
void Mensagem() {
```

```
    if (entrando) {
```

```
        lcd_1.begin(16, 2); // Inicializa o display LCD com 16 colunas e 2 linhas
```

```
        lcd_1.print("Bem Vindo"); // Exibe mensagem de boas-vindas
```

```
        while (seconds >= 0) {
```

```
            lcd_1.setCursor(0, 1); // Move o cursor para a segunda linha
```

```
            lcd_1.print("Fechando em: ");
```

```
            lcd_1.print(seconds);
```

```
            lcd_1.setBacklight(1); // Liga a luz de fundo do LCD
```

```
            delay(500); // Espera 500 milissegundos
```

```
        delay(500); // Espera mais 500 milissegundos
```

```
        seconds -= 1; // Decrementa a contagem
```

```
    }
```

```
    entrando = !entrando; // Alterna o estado de entrada
```

```
    } else {
```

```
        lcd_1.begin(16, 2); // Inicializa o display LCD com 16 colunas e 2 linhas
```

```
        lcd_1.print("Volte sempre"); // Exibe mensagem de despedida
```

```
        while (seconds >= 0) {
```

```
            lcd_1.setCursor(0, 1); // Move o cursor para a segunda linha
```

```
            lcd_1.print("Fechando em: ");
```

```
            lcd_1.print(seconds);
```

```
            lcd_1.setBacklight(1); // Liga a luz de fundo do LCD
```

```
            delay(500); // Espera 500 milissegundos
```

```
            delay(500); // Espera mais 500 milissegundos
```

```
            seconds -= 1; // Decrementa a contagem
```

```
        }
```

```
        entrando = !entrando; // Alterna o estado de entrada
```

```
    }
```

```
    lcd_1.setBacklight(0); // Desliga a luz de fundo do LCD
```

```
    PessoaEntando = false; // Marca que a pessoa não está mais entrando
```

```
    seconds = 5; // Reseta a contagem de segundos
```

```

}

// Função para abrir a porta (mover o servo
para 180 graus)

void AbrePorta() {

    for (pos = 90; pos <= 180; pos += 1) {

        servo.write(pos); // Move o servo para a
posição 'pos'

        delay(10); // Espera 10 milissegundos

    }

}

// Função para fechar a porta (mover o servo
para 0 graus)

void FechaPorta() {

    for (pos = 180; pos >= 90; pos -= 1) {

        servo.write(pos); // Move o servo para a
posição 'pos'

        delay(10); // Espera 10 milissegundos

    }

}

// Loop principal

void loop() {

    testaDistancia(); // Testa a distância
continuamente

    if (musicaLigada) {

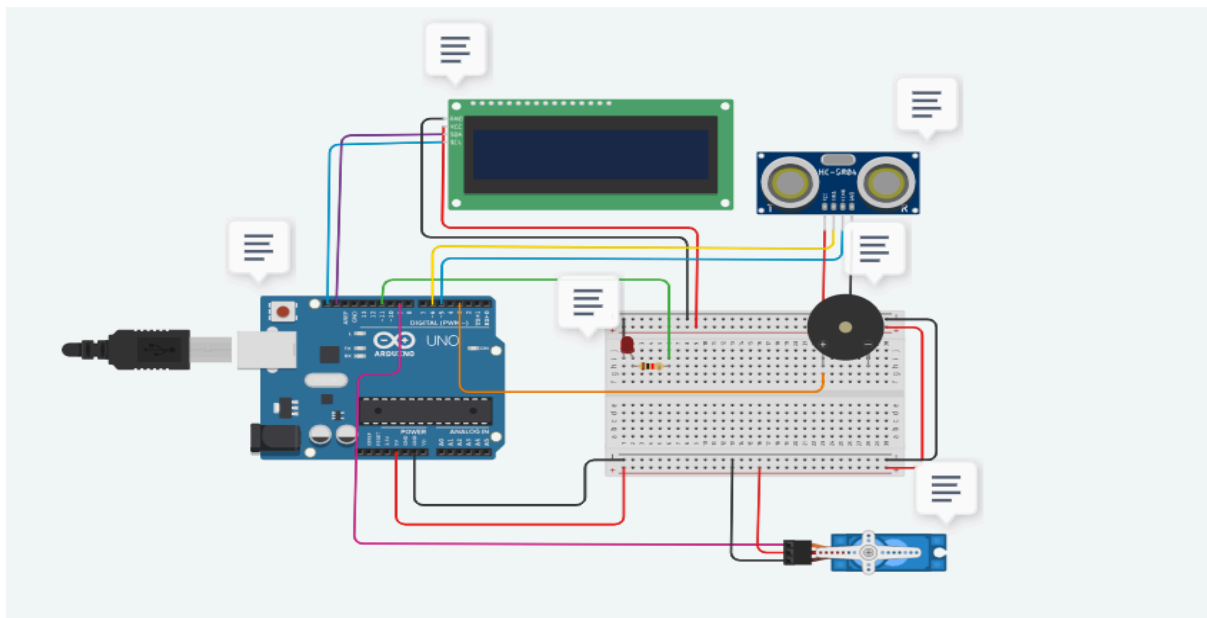
        tocaMusica(); // Toca a música se estiver
ligada

    }

}

```

6 – Imagem da montagem do projeto



7 - Conclusão

Pode-se concluir que o presente trabalho contribuiu de forma positiva para nosso aprendizado em lógica de circuitos e de programação. O uso da linguagem em texto, e a montagem de circuitos, nos trouxe uma clareza sobre como trabalhar em grupo.