

**Politécnico
de Viseu**
Tecnologia
e Gestão Lamego

Trabalho prático de IOT e Ciência de dados

Realizado por:

Alexandre Oliveira

Daniel Santos

Júlio Guerreiro

Índice

Introdução	2
Componentes necessário	3
Montagem do circuito	5
Desenvolvimento do código	10
Valores lidos	12
Conclusão	13

Introdução

No âmbito deste projeto inovador, propomos a criação de um sistema inteligente de controle de portão, integrando tecnologia avançada para automação e monitoramento. O objetivo principal é empregar um sensor ultrassónico em conjunto com um motor, proporcionando a abertura e o fecho automático do portão. Além disso, implementaremos uma funcionalidade de contagem das operações de abertura e fecho do portão.

Para aprimorar a experiência e oferecer indicadores visuais claros, serão utilizados dois LEDs que alertarão sobre o status atual do portão, indicando se está em processo de abertura ou de fecho. A comunicação eficiente entre os diversos componentes será assegurada pela integração do arduino e a transmissão de dados entre os sensores e a aplicação central serão ilustrados no Putty.

Componentes necessários

2 Leds: É um díodo emissor de luz comumente utilizado como fonte luminosa ou sinalizadora em projetos eletrônicos e em determinados locais ou instrumentos onde se torna conveniente a utilização do LED ao invés de uma lâmpada comum.

Resistor 220Ω: São elementos que apresentam resistência à passagem de eletricidade. Podem ter uma resistência fixa ou variável. A resistência elétrica é medida em ohms. Chama-se de Resistência a oposição à passagem de corrente elétrica. Quanto maior a resistência, menor é a corrente elétrica que passa num condutor.

Buzzer: É um pequeno alto-falante destinado a emitir sinais sonoros a partir do oferecimento de energia DC ao módulo, não variando a frequência de emissão.

Arduíno Uno: O Arduíno Uno é uma plataforma open-source de computação física, baseada em uma linguagem de programação que possibilita desenvolver projetos maker diy e de automação residencial.

Jumper: É uma solução para o desenvolvimento de projetos diy robóticos que envolvam conexão de sensores, motores, drives ou mesmo uso para testes em protoboard.

Breadboard: É uma placa com furos e conexões condutoras para montagem de circuitos elétricos experimentais. A grande vantagem da placa de ensaio na montagem de circuitos eletrônicos é a facilidade de inserção de componentes, uma vez que não necessita soldagem nos contatos.

Motor SG90: É um módulo que apresenta movimentos proporcionais aos comandos indicados, controlando o girar e a posição, diferente da maioria dos motores. Possui um ângulo de rotação de 180 graus e acompanha um cabo de 3 pinos referente à alimentação/controlo e diversos acessórios

Sensor ultrassônico: O Sensor Ultrassônico HC-SR04 foi desenvolvido para aperfeiçoar projetos de robótica e microeletrônica, é ideal para calcular a distância com precisão de objetos, com operação entre ~2cm à ~400cm.

Montagem do circuito

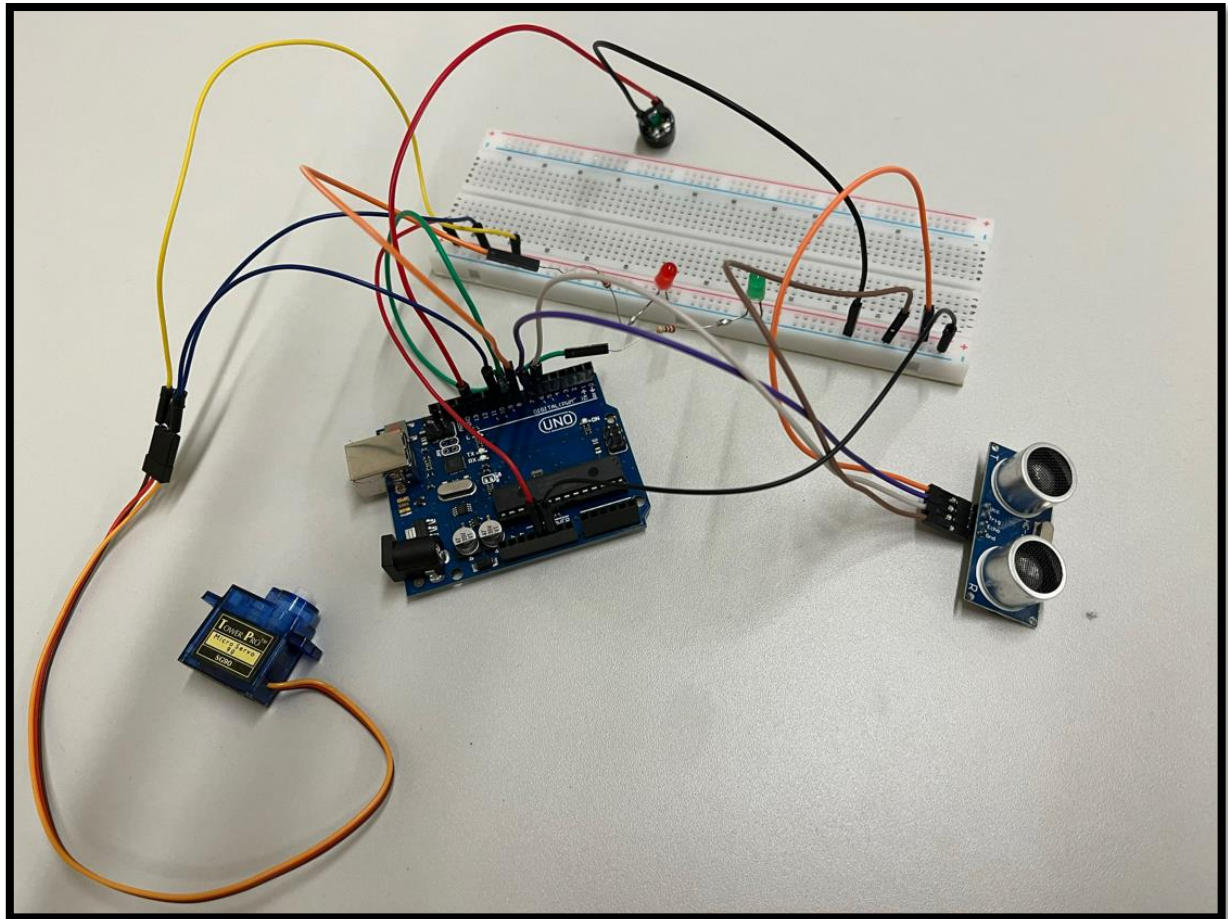


Figura 1- montagem do circuito

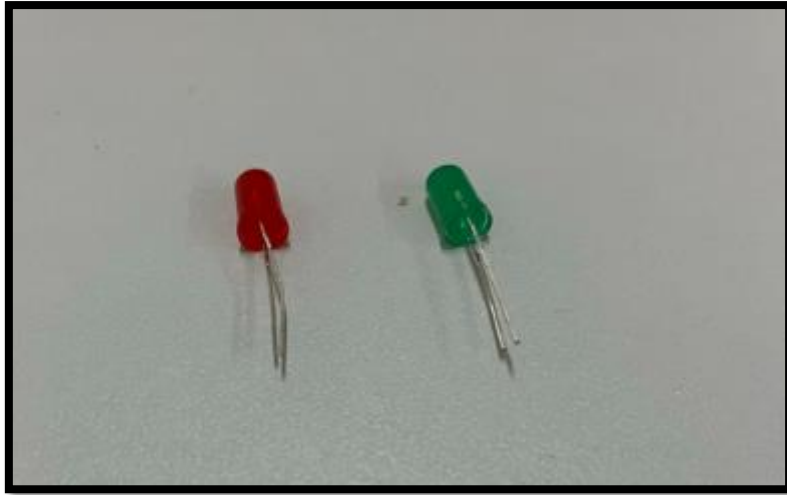


Figura 2- Leds



Figura 3- sensor ultrassónico

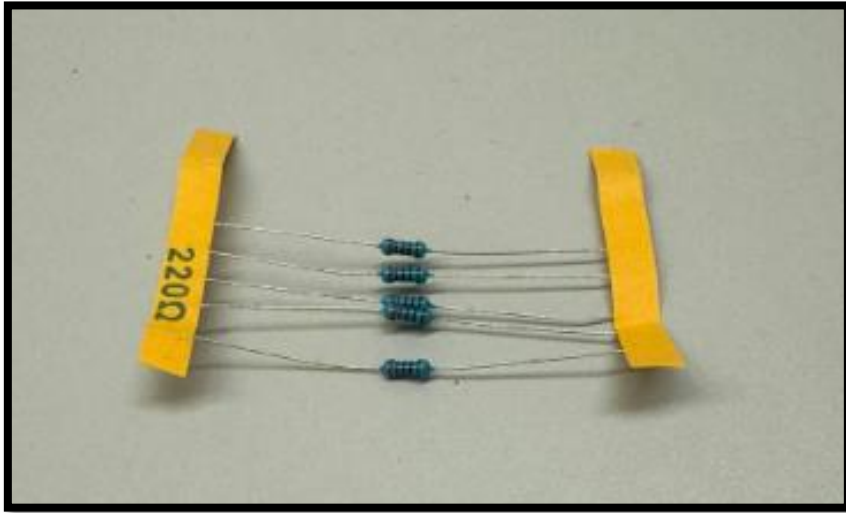


Figura 4- Resistor 220Ω



Figura 5- Arduino Uno



Figura 6- Motor SG90

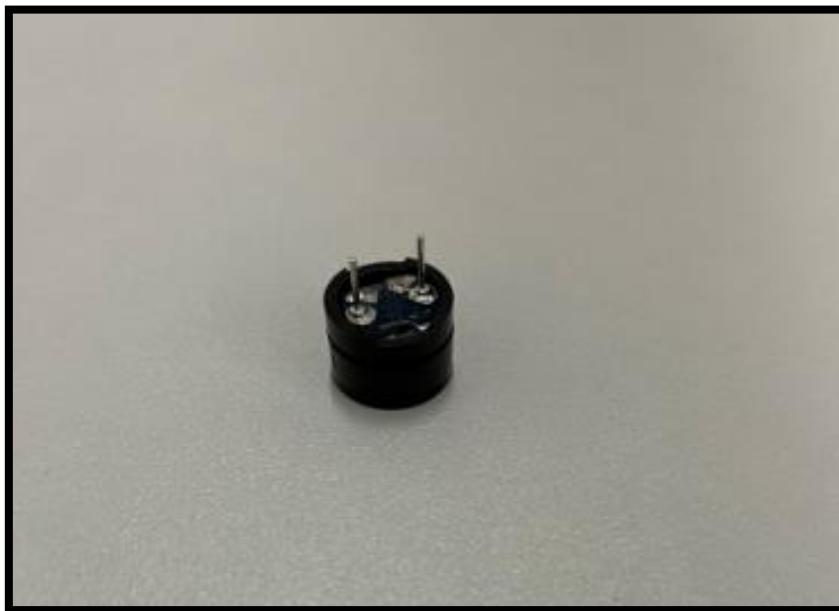


Figura 7- Buzzer ativo



Figura 8- Jumpers

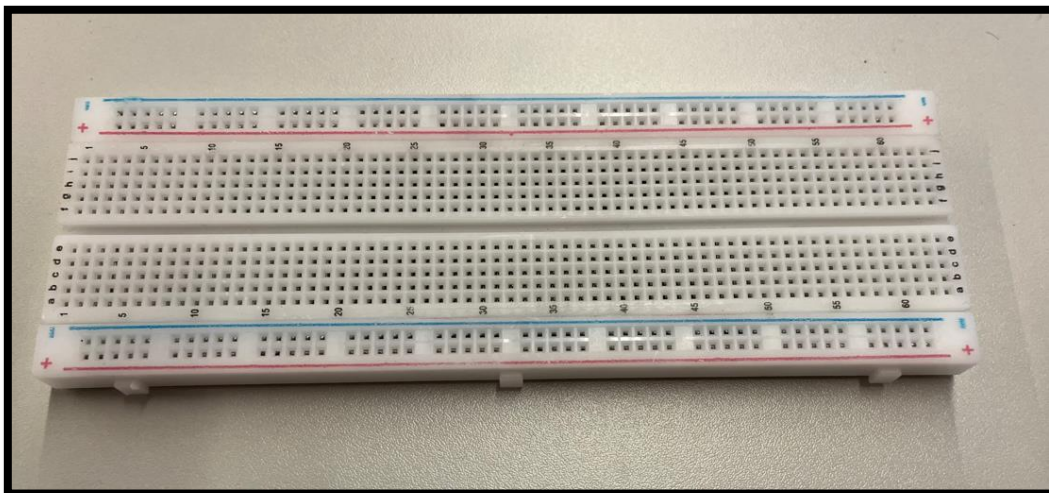
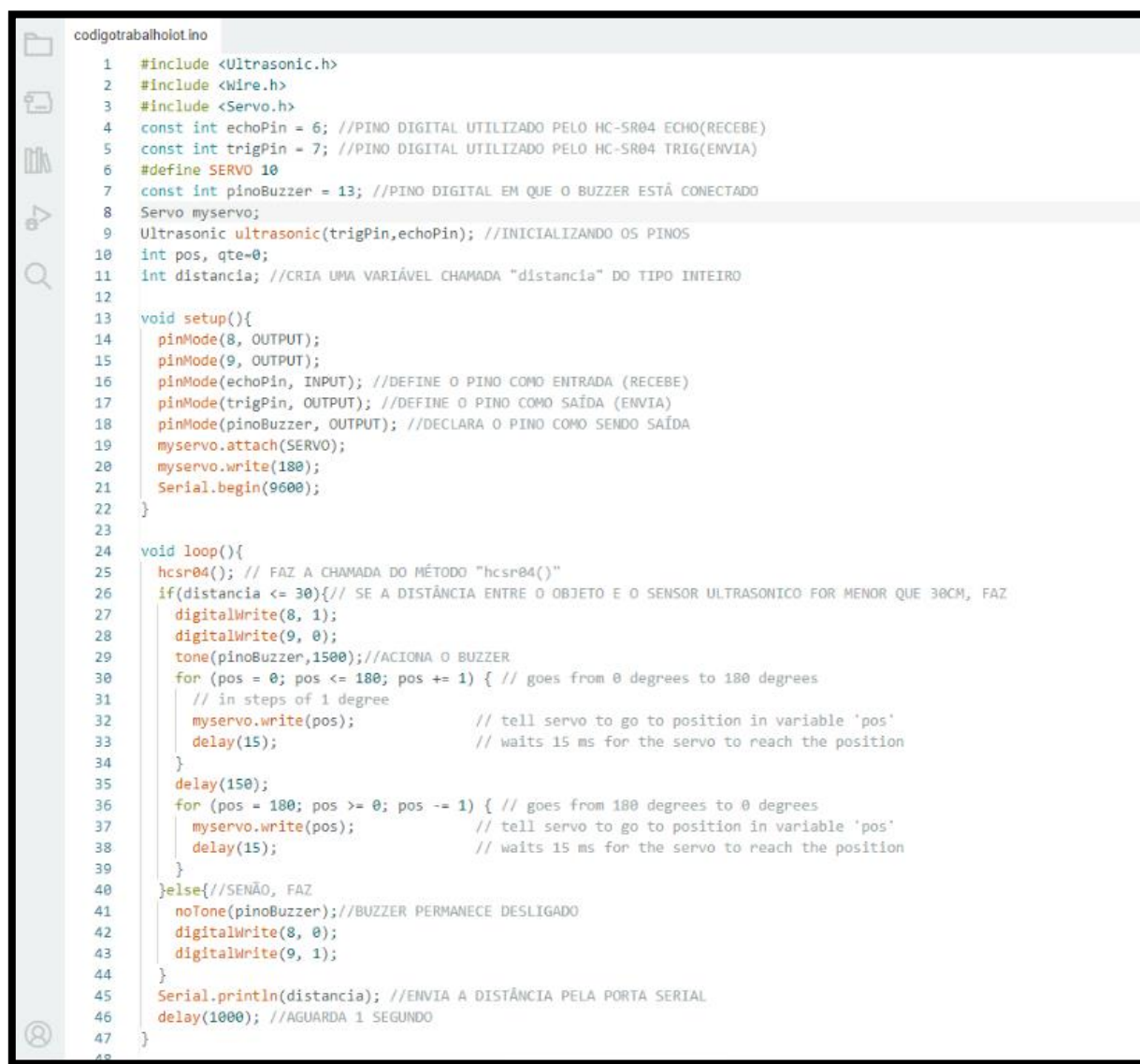


Figura 9- Breadboard

Desenvolvimento do código

Iniciámos por configurar os sensores, para estes ficarem funcionais. Após estes estarem configurados foi necessário desenvolver um código de forma a se comunicarem com a aplicação.

O código foi desenvolvido no arduino ide utilizando: 2Leds, uma Breadboard, um sensor ultrassónico, um arduino uno, um buzzer, duas resistências de 220 ohms e um motor para realizar o projeto.

The image shows a screenshot of the Arduino IDE interface. The file name is 'codigotrabalhoiotino'. The code is written in C++ and includes several libraries: <Ultrasonic.h>, <Wire.h>, and <Servo.h>. It defines several constants: 'echoPin' (6), 'trigPin' (7), 'SERVO' (10), and 'pinoBuzzer' (13). It also declares variables: 'myServo', 'ultrasonic', 'pos', 'qte=0', and 'distancia'. The 'setup()' function initializes pins 8, 9, echoPin, trigPin, and pinoBuzzer, attaches the servo, writes 180 to the servo, and starts the serial port at 9600. The 'loop()' function calls 'hcsr04()' and checks if the distance is less than or equal to 30. If so, it turns on LEDs at pins 8 and 9, turns on the buzzer at pin 13, and moves the servo from 0 to 180 degrees in steps of 1 degree, then back to 0 degrees. If the distance is greater than 30, it turns off the LEDs and buzzer. Finally, it prints the distance to the serial port and delays for 1 second.

```
1  #include <Ultrasonic.h>
2  #include <Wire.h>
3  #include <Servo.h>
4  const int echoPin = 6; //PINO DIGITAL UTILIZADO PELO HC-SR04 ECHO(RECEBE)
5  const int trigPin = 7; //PINO DIGITAL UTILIZADO PELO HC-SR04 TRIG(ENVIA)
6  #define SERVO 10
7  const int pinoBuzzer = 13; //PINO DIGITAL EM QUE O BUZZER ESTÁ CONECTADO
8  Servo myServo;
9  Ultrasonic ultrasonic(trigPin,echoPin); //INICIALIZANDO OS PINOS
10 int pos, qte=0;
11 int distancia; //CRIA UMA VARIÁVEL CHAMADA "distancia" DO TIPO INTEIRO
12
13 void setup(){
14   pinMode(8, OUTPUT);
15   pinMode(9, OUTPUT);
16   pinMode(echoPin, INPUT); //DEFINE O PINO COMO ENTRADA (RECEBE)
17   pinMode(trigPin, OUTPUT); //DEFINE O PINO COMO SAÍDA (ENVIA)
18   pinMode(pinoBuzzer, OUTPUT); //DECLARA O PINO COMO SENDO SAÍDA
19   myServo.attach(SERVO);
20   myServo.write(180);
21   Serial.begin(9600);
22 }
23
24 void loop(){
25   hcsr04(); // FAZ A CHAMADA DO MÉTODO "hcsr04()"
26   if(distancia <= 30){ // SE A DISTÂNCIA ENTRE O OBJETO E O SENSOR ULTRASONICO FOR MENOR QUE 30CM, FAZ
27     digitalWrite(8, 1);
28     digitalWrite(9, 0);
29     tone(pinoBuzzer,1500); //ACIONA O BUZZER
30     for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
31       // in steps of 1 degree
32       myServo.write(pos); // tell servo to go to position in variable 'pos'
33       delay(15); // waits 15 ms for the servo to reach the position
34     }
35     delay(150);
36     for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
37       myServo.write(pos); // tell servo to go to position in variable 'pos'
38       delay(15); // waits 15 ms for the servo to reach the position
39     }
40   }else{//SENÃO, FAZ
41     noTone(pinoBuzzer); //BUZZER PERMANECE DESLIGADO
42     digitalWrite(8, 0);
43     digitalWrite(9, 1);
44   }
45   Serial.println(distancia); //ENVIA A DISTÂNCIA PELA PORTA SERIAL
46   delay(1000); //AGUARDA 1 SEGUNDO
47 }
```

Figura 10- Código no Arduino ide

```
49 //MÉTODO RESPONSÁVEL POR CALCULAR A DISTÂNCIA
50 void hcsr04(){
51     digitalWrite(trigPin, LOW); //SETA O PINO 6 COM UM PULSO BAIXO "LOW"
52     delayMicroseconds(2); // DELAY DE 2 MICROSSEGUNDOS
53     digitalWrite(trigPin, HIGH);
54     delayMicroseconds(10); //
55     distancia = ultrasonic.read(CM); //
56 }
57
```

Figura 11- Continuação do código

Valores Lidos no Putty



Figura 12- Distância lida no Putty

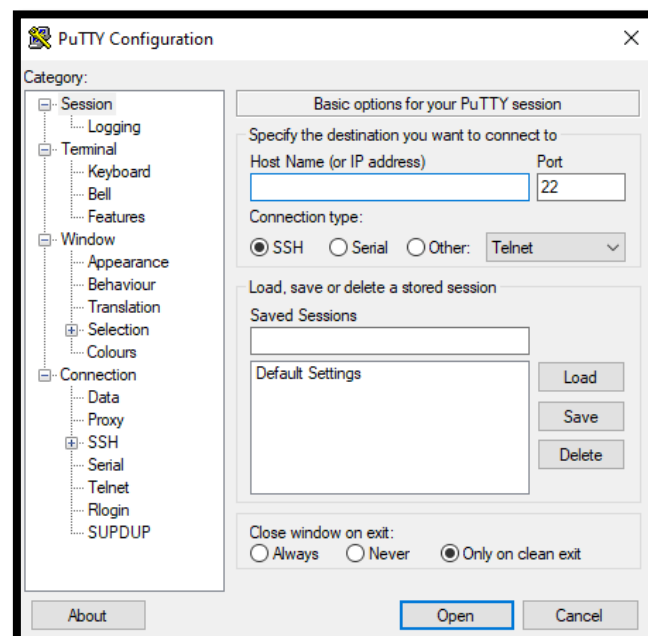


Figura 13- Configuração do Putty

Conclusão

Em suma, o desenvolvimento deste projeto proporcionou uma integração sinérgica de tecnologias avançadas, visando a automação e monitoramento inteligente de um portão.

A implementação de funcionalidades adicionais, como a contagem das operações do portão e indicadores visuais através dos LEDs, elevou a experiência do utilizador a um novo patamar, oferecendo informações claras sobre o estado atual do portão. Esses elementos visuais não apenas aprimoram a usabilidade do sistema, mas também promovem uma interface intuitiva para o utilizador, facilitando a compreensão e interação do funcionamento do portão automático.

Assim, concluímos que este projeto não apenas abraça as tecnologias emergentes para criar soluções inovadoras, mas também demonstra a capacidade de integrar hardware e software de forma coerente.