UEL Student Branch

Relatório Final

Projeto: Alimentador Animal
Gerente: Ernandes Oliveira
Integrantes: Ernandes Oliveira, Renato Maroja Neto

Londrina, PR
31 de outubro de 2018

1 Resumo Final

O objetivo do produto final era um dispositivo que despejasse comida para algum PET quando fosse apertado um botão no celular, tal controle foi realizado utilizando um servo motor, foram feitos dois botões, um para encher e outro para completar, a diferença era o tempo que o servo permitia a passagem de comida.

O dispositivo deveria dizer o nível de comida no recipiente, isso foi feito utilizando um sensor ultrassônico HC-SR04.

Inicialmente era desejado que o dispositivo fosse capaz de dizer quanta comida ainda havia no pote de ração, isso seria realizado com uma célula de carga(Strain Gage, Load Cell) half bridge chamada "sen 1245"capaz de sentir 50kg pois era mais barata, e um amplificador comprado pronto de uso específico para esta célula chamado de Hx711. No entanto após diversas tentativas de funcionamento de tal equipamento a equipe votou por desistir de utilizar a mesma.

A interface foi optada por ser realizada por um webserver ao invés de um aplicativo por dois motivos, o primeiro era para se diferenciar de outros projetos da mesma área, o segundo era para aprender HTML.

O microcontrolador utilizado foi o ESP8266 NodeMCU, hoje em dia muito utilizado em projetos de IoT dentro e fora do Ramo.

2 Teoria

Ultrassonico: O sensor ultrassônico funciona com um principio extremamente simples, é enviada uma onda sonora, esta onda será refletida e voltará ao sensor, que sentirá ela, como a velocidade do som é conhecida e temos o tempo de demora até a onda retornar torna-se simples saber qual a distância de um objeto até o sensor. Para saber mais sobre o sensor ultrassônico recomenda-se dois vídeos simples, o primeiro explica a teoria do sensor e mostra como utilizar o mesmo com um arduino, o segundo mostra como fazer seu próprio sensor ultrassônico, ambos podem ser encontrados na bibliográfia em |4| e |5|

Servo Motor: O servo motor é um motor com um circuito de controle, o mesmo recebe um sinal PWM e responde ao duty cicle com um ângulo do motor, ou seja para um DT X o motor ficará em um ângulo θ.

Weberser: Um servidor Web pode se referir tanto à um software quanto à um hardware. quando se trata de hardware um servidor web é basicamente um computador que armazena documentos(HTML, imagens,Java) e fornece para o usuário, quando se trata de software um servidor web é basicamente um programa que serve e controla como o usuário acessa o conteúdo(documentos) através de um protocolo(geralmente HTTP).

Afim de melhor ilustrar as diferenças abaixo esta um exemplo retirado de $https://developer.mozilla.org/pt-BR/docs/Learn/Common_questions/o_que_eu_m_web_server$

"Em um nível mais básico, o navegador fará uma requisição utilizando o protocolo HTTP sempre que necessitar de um um arquivo hospedado em um servidor web. Quando a requisição alcançar o servidor web correto (hardware), o servidor HTTP (software) enviará o documento requerido, também via HTTP."

Existe uma imensa quantidade de teoria por trás do que é um webserver, no entanto elas não foram relevantes para o projeto, uma vez que o as bibliotecas do arduino tornam extremamente fácil a criação, configuração e utilização de um webserver.

2.1 Componentes e materiais utilizados

1. ESP8266 NodeMCU
2. Servo Motor
3. Sensor Ultrassônico
4. Madeiras(Estrutura)
5. Galão de água: recipiente para ração

Observação: Foi utilizado o ESP8266 por ser um microcontrolador com WiFi embutido, sendo muito mais barato do que comprar um arduino uno e um shield de ethernet.

3 Erros e Dificuldades

O maior erro do projeto foi a primeira estrutura, a mesma tentou seguir um modelo feito por outra pessoa mas foi extremamente mal planejada.

Uma dificuldade do projeto foi que o webserver cai com uma certa frequência, e não se pode permanecer à grandes distâncias do ESP para controlar o alimentador, não foi descoberto o que estava causando isto, nem como resolver o problema.

A terceira dificuldade foi a falta de documentação em vídeos sobre as explicações das funções utilizadas na criação de um webserver e na criação de um Access Point, no entanto foi possível encontrar conteúdo de qualidade que será apresentado posteriormente neste relatório.

4 Resultados Finais

4.1 Estrutura

A primeira estrutura feita pode ser observada na figura 1.

Figura 1: Estrutura Original

Fonte: Autor.

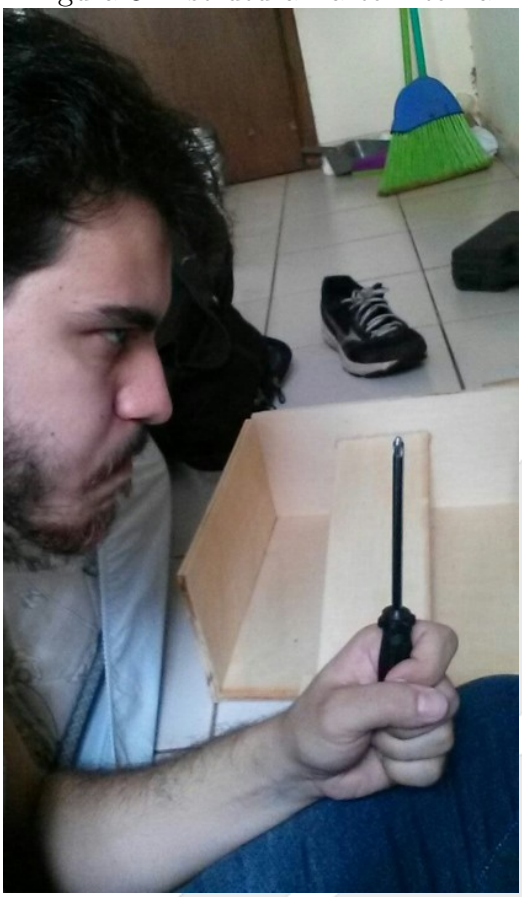
A ideia deste design era era colocar uma garrafa de refrigerante na parte de cima para conter a ração, a boca da garrafa seria tampada por uma madeira ligada à um servo motor, quando o servo abrisse a ração cairia em um pote. Infelizmente esta estrutura foi mal projetada, e cedeu com seu próprio peso, foi então recriado o design do alimentador, este novo design pode ser visto na figura 2.

Figura 2: Estrutura Recriada

Fonte: Autor.

Para compreender melhor este design pode ser observado a parte interna da estrutura deve-se olhar a figura 3.

Figura 3: Estrutura Parte Interna



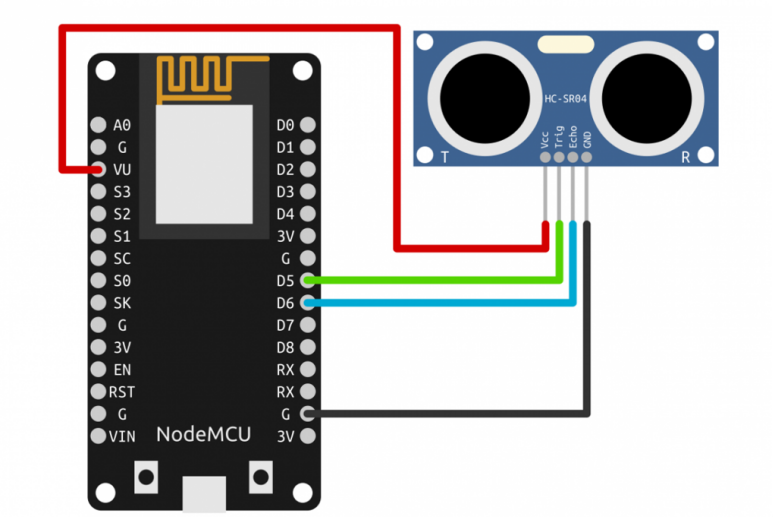
Fonte: Autor.

Como podem ser vistas nas figuras 2 e 3, a nova estrutura foi melhor projetada, consistia em um galão de água para servir como um recipiente para a ração, um servo(que será melhor explicado posteriormente) era utilizado para permitir a passagem da ração por uma rampa que levaria a ração até o pote de ração. neste modelo o grupo já havia desistido de utilizar o strain gage, por isso o pote de ração pode ser conectado diretamente na estrutura, na parte superior do recipiente foi utilizado um sensor ultrassônico para verificar o quanto de ração ainda existia no recipiente, esta informação era enviada para o webserver.

4.2 Sensor Utrassônico

Esta parte do projeto foi simples, simplesmente foi, foi seguido o esquemático diposto na figura 4

Figura 4: Diagrama do Sensor Ultrassônico



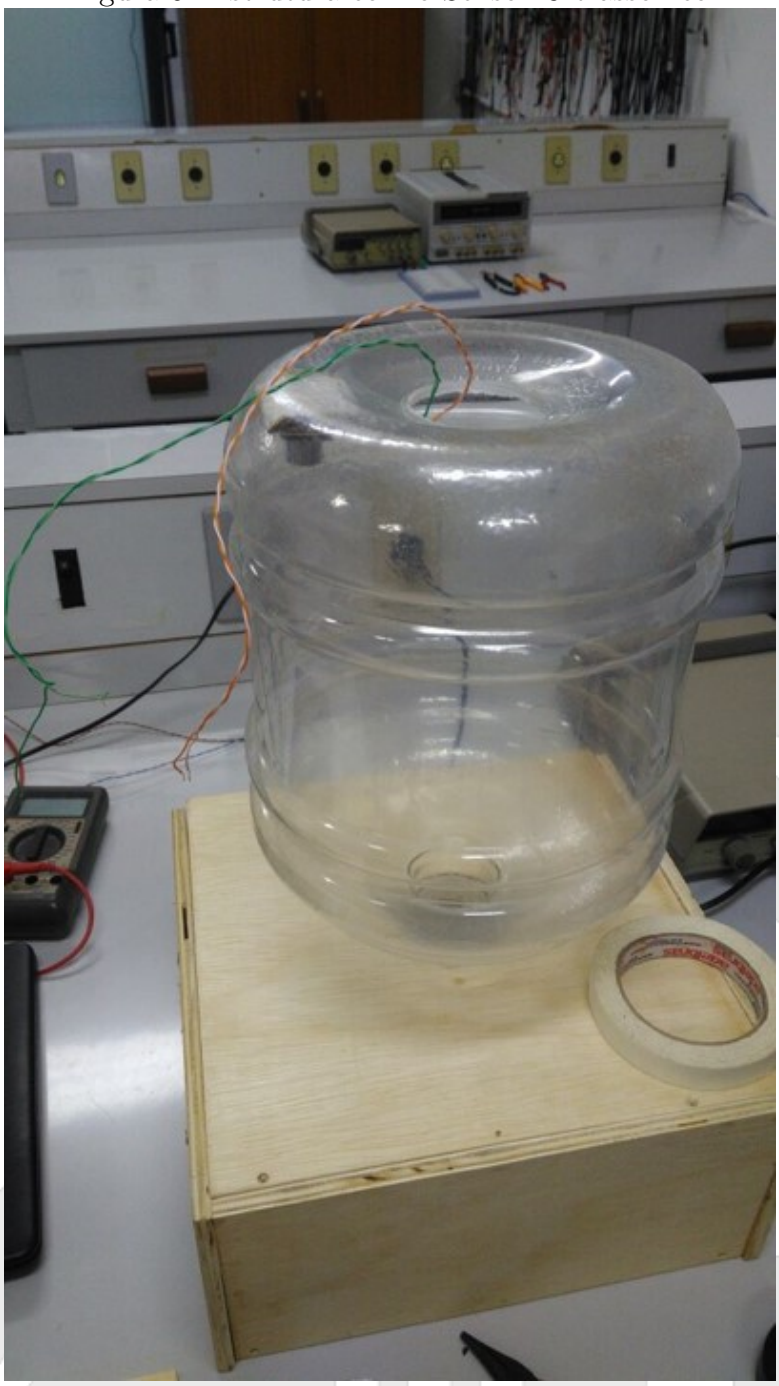
Fonte: |1|

Dois pinos para a alimentação do sensor ultrassônico, um para o Trigger e um para o ECHO, o código necessário para tal feito pode ser visto abaixo.

```
#include <Ultrasonic.h>

#define TRIGGER_PIN  D0
#define ECHO_PIN     D1
float recipiente;
float total = 30;

Ultrasonic ultrasonic(TRIGGER_PIN, ECHO_PIN);

void setup() {
}

void loop() {
  recipiente = ultrasonic.Ranging(CM);
  recipiente = total-recipiente;
  delay(200);
}
```

O código é simples, apenas é definido um pino para o TRIGGER e para o ECHO, são feitos dois floats, um para o recipiente, e um para o total, isso será utilizado durante o webserver, mas basicamente sabendo que o comprimento total do galão são 30cm e sabendo a distância do sensor ultrassônico até a ração é dado por ultrassonic.Ranging(CM), podemos fazer uma matematica simples para determinar quantos cm de ração ainda existem no recipiente. Uma foto do Sensor ultrassônico disposto no alimentador pode ser vista na figura 5.

Figura 5: Estrutura com o Sensor Ultrassônico

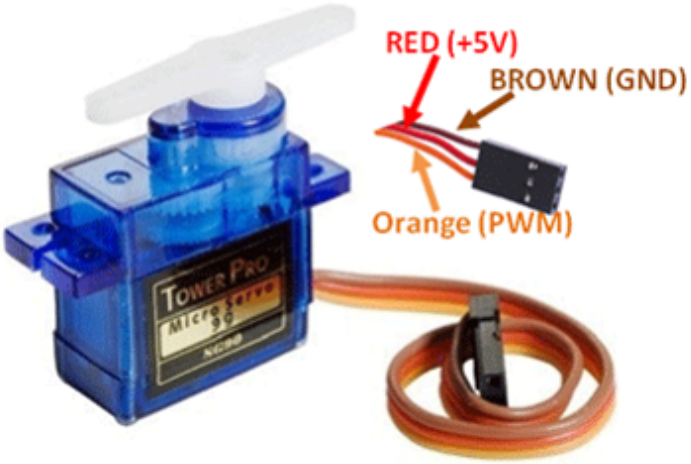


Fonte: Autor.

4.3 Servo

Uma foto do diagrama utilizado na montagem do servo pode ser observada na figura 6

Figura 6: Diagrama do Servo Motor



Fonte: Não Encontrada.

A figura 6 mostra o simples diagrama de montagem do servo como já comentado anteriormente, como pode ser visto basta dois pinos de alimentação e um pino de sinal, o qual consiste de um sinal PWM, no entanto enviar um PWM para se obter o ângulo desejado é extremamente fácil pois a biblioteca do arduino faz este trabalho sozinha. Uma imagem do Servo motor dentro da estrutura por ser vista na figura 7.

Figura 7: Servo Motor na Estrutura

Fonte: Autor.

A figura 7 mostra novamente a parte interna do alimentador, no entanto desta vez já é possível ver as paredes de papelão da rampa para garantir que a ração não caísse para fora, assim como o servo motor, que foi fixado com cola quente, basicamente quando aberto é permitido a passagem de alimentos. O código do servo motor é extremamente fácil e pode ser visto abaixo.

```
#include <Servo.h>
```

```
Servo servo1;

void Encher(){
  servo1.write(110);
  delay(3000);
  servo1.write(15);
  server.send(204,"");
}

void completar(){
  servo1.write(110);
  delay(1500);
  servo1.write(15);
  server.send(204,"");
  }

void setup() {
  servo1.attach(D3);
  servo1.write(15);
}
void loop() {
}
```

O código faz mais sentido quando em conjunto com o webserver, pois as funções completar e Encher são chamadas pelo webserver, que será explicado posteriormente, mas basicamente é criado um objeto chamado servo1, no setup o pino de sinal é colocado como D3, e é escrito no servo que ângulo desejado é de 15°, não 0° pois quando o servo fechada sempre ficava algum alimento abaixo dele, isso fazia o servo ser forçado e nunca atingir o ângulo, no entanto com 15° apesar de alimento ficar abaixo do servo isso força menos o servo e ainda fecha a porta.

4.4 Webserver

Para esta parte do projeto foi utilizado um código exemplo para utilizar o ESP como Access Point, este exemplo se chama WiFiAccessPoint, o código utilizado no alimentador pode ser observado abaixo.

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>

#define DEBUG true

const char *ssid = "ESPap";
const char *password = "thereisnospoon";
```

```
ESP8266WebServer server(80);

void handleRoot() {

String webpage ="<html> ";
   webpage +="<style> body { background-color: #cccccc; font-family: Arial,
   Helvetica, Sans-Serif; Color: #000088; } </style>";
   webpage+="O recipiente esta: ";
   webpage+= recipiente;
   webpage+= " cm cheio";
   webpage+= "<br/>";
   webpage+= "<br/>";
   webpage +="clique neste botao para encher o recipiente";
   webpage +="<button><a href=\"Encher\" link=\"\"> Encher </a></button>";
   server.on("/Encher",Encher);
   webpage+= "<br/>";
   webpage +="Clique neste botao para completar o recipiente";
   webpage +="<button><a href=\"completar\" link=\"\"> completar </a></button>";
   server.on("/completar",completar);
   webpage+="</html>";

  server.send(200, "text/html", webpage);
  Serial.println();
}

void setup() {

  Serial.begin(115200);
  Serial.println();
  Serial.print("Configuring access point...");
  WiFi.softAP(ssid, password);

  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(myIP);
  server.on("/", handleRoot);
  server.begin();
  Serial.println("HTTP server started");
}

void loop() {
  server.handleClient();
}
```

Após a devida inclusão das bibliotecas são configuradas respectivamente o nome da rede que estará sendo gerada, assim como sua senha. É então criado um objeto chamado server e conectado o mesmo na porta 80, todos os vídeos procurados apenas citam a porta 80, no entanto não explicam o que ela é.

É configurada a função handlleRoot, podem ser criadas mais de uma página, com várias páginas em diferentes funções, como mostra o excelente vídeo sobre access point disponibilizado nas referências em |2|. Dentro desta função é criada uma string chamada webpage, para alguns casos é utilizado um char ao invés de string, o importante é que dentro desta string é feito um código em html, o qual basicamente diz o quanto de ração ainda tem no recipiente e oferece dois botões para serem apertados, um para encher e outro para completar, sendo a diferença entre os dois apenas o tempo em que o servo permite a passagem de comida.

Para aqueles que desejem aprender sobre HTML é recomendado que acessem as vídeos aulas disponibilizadas na bibliografia em |3|, porém será ensinado o básico, primeiramente é aberta uma tag de html que é fechada no final do código, para fechar uma tag é colocado uma barra na frente do comando.

Exemplo:

Abrindo tag de HTML: <html>

Fechando tag de HTML: </html>

Note que o operador "é utilizado para abrir e fechar o que deve ser interpretado em ASC no código, portanto caso deseje-se utilizar o operador "em alguma parte do código hmtl basta utilizar o operador / antes, ou seja /".

Existem tags que abrem e fecham no mesmo lugar, como a tag
(line break), ela basicamente funciona como uma tecla enter.

Dentro da tag <style> é colocada as configurações do texto, como por exemplo a cor de fundo, a cor da letra, que fonte de letra será utilizada.

Note no código que pode-se acrescentar variáveis dentro da string como é feito com o float recipiente.

Vale a pena se concentrar em na parte de botões do código, utilizando a tag <button>, note que o mesmo chama uma função, no caso o primeiro botão criado chama a função Encher, enquanto o segundo chama a função completar.

Por final para esta função se atente ao comando server.send, seu primeiro argumento é 200, assim como 404 significa not found, 200 significa successful response. Seu segundo argumento é o formato em que a resposta será enviada, para o caso demonstrado será html, e por final é enviado o código.

Dentro da função setup existem vários códigos sobre comunicação serial que não serão explicados pois não é necessário para o trabalho, focando nas partes importantes deste código, o comado WiFi.softAP(ssid,password) é bastante intuitivo, e simplesmente cria um Access Point com o nome senha escolhidos previamente. Posteriormente a linha IPAddress... Esta linha de código é menos intuitiva, mas simplesmente é criado um objeto chamado myIP do tipo IPAddress, ou seja, este objeto apenas armazenará valores do tipo IPAdreess(no caso endereços de IP), para pegar o endereço de IP da rede é utilizado Wifi.softAPIP, novamente muito intuitivo, o motivo de se desejar o IP é para plota-lo posteriormente no código, pois é através dele que será acessada a webpage.

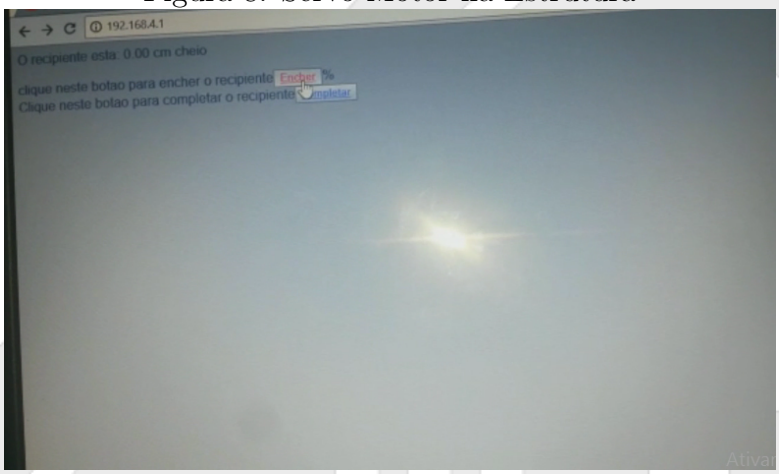
A próxima linha de código importante é dada por server.on, onde seu primeiro argumento

recebido é /, e o segundo é handdleRoot(função onde esta a webpage), o argumento / define que handdleRoot será a função que irá fornecer a página raiz(principal) do webserver. Como explicado no vídeo disponibilizado em |2|, caso fosse necessário criar mais de uma página poderia se utilizar por exemplo /1, ou /2, e atribuir estes argumentos à alguma função que contenha uma webpage. caso não esteja claro o porque imagine um link de algum site, como por exemplo a página pessoal de um professor da uel (http://www.uel.br/pessoal/fulano/) o link é separado com barras ou seja, /pessoal indicando que estamos indo para páginas pessoais, e /fulano indicando que estamos indo para página pessoal de fulano. No caso do webserver criado é muito simples, o endereço de IP é dado por 192.168.4.1 onde a página principal é dada por isso mesmo, ou se for escrito 192.168.4.1/ , já caso fosse criada uma página referenciada por /1 teríamos 192.168.4.1/1. Basicamente o server.on é utilizado para alocar o usuário em diferentes páginas. Voltando ao código do botão nota-se que foram utilizados serve.on, na prática isso levava o usuário para páginas inesitantes, fazendo com que o usuário precisasse voltar para a página inicial, isso não é um erro de código, mas é uma péssima implementação, uma que não faz sentido, a desculpa é que o código foi criado sem os membros possuírem muitos conhecimentos.

A última função de interesse é server.handleClient, novamente uma função extremamente intuitiva, ela simplesmente comanda para que o ESP gerencie todos os clientes que estiverem acessando, ou requisitando uma conexão.

Uma imagem que mostra como ficou a webpage pode ser vista na figura, 8, como pode ser visto ela é extremamente simples para não dizer feia, porém é um simples trabalho em HTML para que página fique mais bonita.

Figura 8: Servo Motor na Estrutura



Fonte: Autor.

4.5 Finalização

Por final foi feita colocado EVA por fora do alimentador para torna-lo azul seguindo as cores do RAMO, foi também feita uma PCB para o mesmo, mas não adiantou muito na contenção de fios uma vez que o circuito não possuía componentes, apenas conexões.

Todos os códigos foram agrupados formando assim o código do alimentador, que pode ser visto abaixo.

```
    #include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
#include <Ticker.h>
#include <EEPROM.h>


#define buttonPin D2  // the number of the pushbutton pin
#define rele D3      // the number of the LED pin
#define Led_Aviso D4

Ticker botao;
Ticker tyme;

// Variables will change:
int releState = LOW;         // the current state of the output pin
int buttonState;             // the current reading from the input pin
int lastButtonState = LOW;   // the previous reading from the input pin
int led_S = LOW;


unsigned long lastDebounceTime = 0;  // the last time the output pin was toggled
unsigned long debounceDelay = 25;    // the debounce time; increase if the output flicke



enum Estado_Botao {
  desligado,
  iniciar_banho,
  pausar_banho,
  continuar_banho
};


enum Estado_Banho {
  habilitado,
  desabilitado
};



Estado_Botao Estado_Bot = desligado;
Estado_Banho Banho = habilitado;
```

```
/* Set these to your desired credentials. */
const char *ssid = "ESPap";
const char *password = "thereisnospoon";



//posiçoes para armazenar variaveis na EEPROM
int address_tempo = 0;
int address_espera = 1;
int address_pausa = 2;

byte armazenado;
byte minutos = EEPROM.read(address_tempo); //tempo de banho
byte minutos_espera = EEPROM.read(address_espera); //tempo de espera até o banho ser hab
byte minutos_pausa = EEPROM.read(address_pausa); // tempo que o banho pode ficar pausado
int tempo = (int)minutos*60;
int tempo_espera = (int)minutos_espera*60;
int tempo_de_pausa= (int)minutos_pausa*60;


ESP8266WebServer server(80);

/* Just a little test message.  Go to http://192.168.4.1 in a web browser
 * connected to this access point to see it.
 * http://www.esp8266.com/viewtopic.php?f=8&t=4345
 */
void handleRoot() {

  String webpage ="<html> ";

   webpage +="<style> body { background-color: #cccccc; font-family: Arial, Helvetica, S
   webpage +="<h1> Web Server - Chuveiro</h1> ";
   webpage+="<br/><h2>Aqui podem ser controlado o tempo de duração do banho, o tempo que
   webpage+="O banho esta configurado para: <br/>";
   webpage+=minutos;
   webpage+=" minutos de duraçao, ";
   webpage+=minutos_espera;
   webpage+=" minutos de espera e, ";
   webpage+=minutos_pausa;
   webpage+=" minutos de pausa";
```

```
webpage+="<br/><br/>Escolha quantos minutos o banho vai durar: <br/>";
webpage +="<button><a href=\"tresmin\" link=\"\">3 minutos</a></button>";
server.on("/tresmin",tresmin);
webpage +="<button><a href=\"cincomin\" link=\"\"> 5 minutos</a></button>";
server.on("/cincomin",cincomin);
webpage +="<button><a href=\"setemin\" link=\"\">7 minutos</a></button>";
server.on("/setemin",setemin);
webpage +="<button><a href=\"dezmin\" link=\"\">10 minutos</a></button>";
server.on("/dezmin",dezmin);
webpage +="<button><a href=\"quinzemin\" link=\"\">15 minutos</a></button>";
server.on("/quinzemin",quinzemin);
webpage +="<button><a href=\"vintemin\" link=\"\">20 minutos</a></button>";
server.on("/vintemin",vintemin);
webpage +="<button><a href=\"trintamin\" link=\"\">30 minutos</a></button>";
server.on("/trintamin",trintamin);


webpage += "<br/> Incrementar ou decrementar o numero de minutos atual: <br/>";
webpage +="<button><a href=\"maisum\" link=\"\"> +1minutos</a></button>";
server.on("/maisum",maisum);
webpage +="<button><a href=\"maistres\" link=\"\">+3 minutos</a></button>";
server.on("/maistres",maistres);
webpage +="<button><a href=\"maiscinco\" link=\"\">+5 minutos</a></button>";
server.on("/maiscinco",maiscinco);
webpage +="<button><a href=\"menosum\" link=\"\">-1 minutos</a></button>";
server.on("/menosum",menosum);
webpage +="<button><a href=\"menostres\" link=\"\">-3 minutos</a></button>";
server.on("/menostres",menostres);
webpage +="<button><a href=\"menoscinco\" link=\"\">-5 minutos</a></button>";
server.on("/menoscinco",menoscinco);
webpage +="</b></h2>";


webpage += "<br/><br/> Escolha o tempo de que o chuveiro fica desligado após ser util
webpage +="<button><a href=\"tresminu\" link=\"\">3 minutos</a></button>";
server.on("/tresminu",tresminu);
webpage +="<button><a href=\"cincominu\" link=\"\"> 5 minutos</a></button>";
server.on("/cincominu",cincominu);
webpage +="<button><a href=\"seteminu\" link=\"\">7 minutos</a></button>";
server.on("/seteminu",seteminu);
webpage +="<button><a href=\"dezminu\" link=\"\">10 minutos</a></button>";
server.on("/dezminu",dezminu);
webpage +="<button><a href=\"quinzeminu\" link=\"\">15 minutos</a></button>";
server.on("/quinzeminu",quinzeminu);
```

```
webpage +="<button><a href=\"vintemin\" link=\"\">20 minutos</a></button>";
server.on("/vinteminu",vinteminu);
webpage +="<button><a href=\"trintaminu\" link=\"\">30 minutos</a></button>";
server.on("/trintaminu",trintaminu);



webpage += "<br/> Incrementar ou decrementar o tempo de espera do banho: <br/>";
webpage +="<button><a href=\"masum\" link=\"\"> +1minutos</a></button>";
server.on("/masum",masum);
webpage +="<button><a href=\"mastres\" link=\"\">+3 minutos</a></button>";
server.on("/mastres",mastres);
webpage +="<button><a href=\"mascinco\" link=\"\">+5 minutos</a></button>";
server.on("/mascinco",mascinco);
webpage +="<button><a href=\"menoum\" link=\"\">-1 minutos</a></button>";
server.on("/menoum",menoum);
webpage +="<button><a href=\"menotres\" link=\"\">-3 minutos</a></button>";
server.on("/menotres",menotres);
webpage +="<button><a href=\"menocinco\" link=\"\">-5 minutos</a></button>";
server.on("/menocinco",menocinco);



webpage += "<br/><br/> Escolha o tempo de que o banho pode ficar pausado: <br/>";
webpage +="<button><a href=\"tresminut\" link=\"\">3 minutos</a></button>";
server.on("/tresminut",tresminut);
webpage +="<button><a href=\"cincominut\" link=\"\"> 5 minutos</a></button>";
server.on("/cincominut",cincominut);
webpage +="<button><a href=\"seteminut\" link=\"\">7 minutos</a></button>";
server.on("/seteminut",seteminut);
webpage +="<button><a href=\"dezminut\" link=\"\">10 minutos</a></button>";
server.on("/dezminut",dezminut);
webpage +="<button><a href=\"quinzeminut\" link=\"\">15 minutos</a></button>";
server.on("/quinzeminut",quinzeminut);
webpage +="<button><a href=\"vinteminut\" link=\"\">20 minutos</a></button>";
server.on("/vinteminut",vinteminut);
webpage +="<button><a href=\"trintaminut\" link=\"\">30 minutos</a></button>";
server.on("/trintaminut",trintaminut);



webpage += "<br/> Incrementar ou decrementar o tempo que o banho pode ficar pausado:
webpage +="<button><a href=\"mas_um\" link=\"\"> +1minutos</a></button>";
server.on("/mas_um",mas_um);
```

```
   webpage +="<button><a href=\"mas_tres\" link=\"\">+3 minutos</a></button>";
   server.on("/mas_tres",mas_tres);
   webpage +="<button><a href=\"mas_cinco\" link=\"\">+5 minutos</a></button>";
   server.on("/mas_cinco",mas_cinco);
   webpage +="<button><a href=\"meno_um\" link=\"\">-1 minutos</a></button>";
   server.on("/meno_um",meno_um);
   webpage +="<button><a href=\"meno_tres\" link=\"\">-3 minutos</a></button>";
   server.on("/meno_tres",meno_tres);
   webpage +="<button><a href=\"meno_cinco\" link=\"\">-5 minutos</a></button>";
   server.on("/meno_cinco",meno_cinco);



   webpage +="</b></h2>";


  server.send(200, "text/html", webpage);
  Serial.println();
}

//setar os minutos

void tresmin(){
  minutos =3;
  EEPROM.write(address_tempo,minutos);
  EEPROM.commit();
  server.send(204,"");
  }

void cincomin(){
  minutos =5;
  EEPROM.write(address_tempo,minutos);
  EEPROM.commit();
  server.send(204,"");
  }

  void setemin(){
  minutos =7;
  EEPROM.write(address_tempo,minutos);
  EEPROM.commit();
  server.send(204,"");
  }

  void dezmin(){
```

```
minutos =10;
EEPROM.write(address_tempo,minutos);
EEPROM.commit();
server.send(204,"");
}

void quinzemin(){
minutos =15;
EEPROM.write(address_tempo,minutos);
EEPROM.commit();
server.send(204,"");
}

void vintemin(){
minutos =20;
EEPROM.write(address_tempo,minutos);
EEPROM.commit();
server.send(204,"");
}

void trintamin(){
minutos =30;
EEPROM.write(address_tempo,minutos);
EEPROM.commit();
server.send(204,"");
}

//incrementar e decrementar

void maisum(){
minutos =minutos + 1;
if(minutos>45){
  minutos=45;
  }
  EEPROM.write(address_tempo,minutos);
  EEPROM.commit();
server.send(204,"");
}

  void maistres(){
minutos = minutos+3;
if(minutos>45){
  minutos=45;
  }
```

```
  EEPROM.write(address_tempo,minutos);
  EEPROM.commit();
server.send(204,"");
}

  void maiscinco(){
minutos = minutos+5;
if(minutos>45){
  minutos=45;
  }
  EEPROM.write(address_tempo,minutos);
  EEPROM.commit();
server.send(204,"");
}

    void menosum(){
minutos = minutos -1;
if(minutos<1){
  minutos=1;
  }
 if(minutos>45){
  minutos=1;
  }
  EEPROM.write(address_tempo,minutos);
  EEPROM.commit();
server.send(204,"");
}


    void menostres(){
minutos = minutos-3;
if(minutos<1){
  minutos=1;
  }
if(minutos>45){
  minutos=1;
  }
  EEPROM.write(address_tempo,minutos);
  EEPROM.commit();
server.send(204,"");
}

    void menoscinco(){
minutos = minutos-5;
```

```
  if(minutos<1){
    minutos=1;
    }
  if(minutos>45){
    minutos=1;
    }
    EEPROM.write(address_tempo,minutos);
    EEPROM.commit();
  server.send(204,"");
  }



////////////////////////////////////////////tempo_espera/////////////////////////////



//setar os minutos de espera

void tresminu(){
  minutos_espera =3;
  EEPROM.write(address_espera,minutos_espera);
  EEPROM.commit();
  server.send(204,"");
  }

void cincominu(){
  minutos_espera =5;
  EEPROM.write(address_espera,minutos_espera);
  EEPROM.commit();
  server.send(204,"");
  }

  void seteminu(){
  minutos_espera =7;
  EEPROM.write(address_espera,minutos_espera);
  EEPROM.commit();
  server.send(204,"");
  }

  void dezminu(){
  minutos_espera =10;
```

```
EEPROM.write(address_espera,minutos_espera);
EEPROM.commit();
server.send(204,"");
}

void quinzeminu(){
minutos_espera =15;
EEPROM.write(address_espera,minutos_espera);
EEPROM.commit();
server.send(204,"");
}

void vinteminu(){
minutos_espera =20;
EEPROM.write(address_espera,minutos_espera);
EEPROM.commit();
server.send(204,"");
}

void trintaminu(){
minutos_espera =30;
EEPROM.write(address_espera,minutos_espera);
EEPROM.commit();
server.send(204,"");
}

//incrementar e decrementar os minutos de espera

void masum(){
minutos_espera =minutos_espera + 1;
if(minutos_espera>45){
  minutos_espera=45;
  }
  EEPROM.write(address_espera,minutos_espera);
  EEPROM.commit();
server.send(204,"");
}

  void mastres(){
minutos_espera = minutos_espera+3;
if(minutos_espera>45){
  minutos_espera=45;
  }
  EEPROM.write(address_espera,minutos_espera);
```

```
  EEPROM.commit();
server.send(204,"");
}

  void mascinco(){
minutos_espera = minutos_espera+5;
if(minutos_espera>45){
  minutos_espera=45;
  }
  EEPROM.write(address_espera,minutos_espera);
  EEPROM.commit();
server.send(204,"");
}

  void menoum(){
minutos_espera = minutos_espera -1;
if(minutos_espera<1){
  minutos_espera=1;
  }
if(minutos_espera>45){
  minutos_espera=1;
  }
  EEPROM.write(address_espera,minutos_espera);
  EEPROM.commit();
server.send(204,"");
}


 void menotres(){
minutos_espera = minutos_espera-3;
if(minutos_espera<1){
  minutos_espera=1;
  }
if(minutos_espera>45){
  minutos_espera=1;
  }
  EEPROM.write(address_espera,minutos_espera);
  EEPROM.commit();
server.send(204,"");
}

 void menocinco(){
minutos_espera = minutos_espera-5;
if(minutos_espera<1){
```

```
    minutos_espera=1;
    }
  if(minutos_espera>45){
    minutos_espera=1;
    }
    EEPROM.write(address_espera,minutos_espera);
    EEPROM.commit();
  server.send(204,"");
  }



////////////////////////////////////////////////tempo_pausa//////////////////////////////

//setar os minutos de pausa

void tresminut(){
  minutos_pausa =3;
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
  server.send(204,"");
  }

void cincominut(){
  minutos_pausa =5;
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
  server.send(204,"");
  }

  void seteminut(){
  minutos_pausa =7;
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
  server.send(204,"");
  }

  void dezminut(){
  minutos_pausa =10;
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
  server.send(204,"");
  }
```

```
void quinzeminut(){
minutos_pausa =15;
EEPROM.write(address_pausa,minutos_pausa);
EEPROM.commit();
server.send(204,"");
}

void vinteminut(){
minutos_pausa =20;
EEPROM.write(address_pausa,minutos_pausa);
EEPROM.commit();
server.send(204,"");
}

void trintaminut(){
minutos_pausa =30;
EEPROM.write(address_pausa,minutos_pausa);
EEPROM.commit();
server.send(204,"");
}

//incrementar e decrementar os minutos de pausa

void mas_um(){
minutos_pausa =minutos_pausa + 1;
if(minutos_pausa>45){
  minutos_pausa=45;
  }
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
server.send(204,"");
}

  void mas_tres(){
minutos_pausa = minutos_pausa+3;
if(minutos_pausa>45){
  minutos_pausa=45;
  }
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
server.send(204,"");
}

  void mas_cinco(){
```

```
minutos_pausa = minutos_pausa+5;
if(minutos_pausa>45){
  minutos_pausa=45;
  }
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
server.send(204,"");
}

  void meno_um(){
minutos_pausa = minutos_pausa -1;
if(minutos_pausa<1){
  minutos_pausa=1;
  }
  if(minutos_pausa>45){
  minutos_pausa=1;
  }
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
server.send(204,"");
}


 void meno_tres(){
minutos_pausa = minutos_pausa-3;
if(minutos_pausa<1){
  minutos_pausa=1;
  }
  if(minutos_pausa>45){
  minutos_pausa=1;
  }
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
server.send(204,"");
}

 void meno_cinco(){
minutos_pausa = minutos_pausa-5;
if(minutos_pausa<1){
  minutos_pausa=1;
  }
  if(minutos_pausa>45){
  minutos_pausa=1;
  }
```

```
  EEPROM.write(address_pausa,minutos_pausa);
  EEPROM.commit();
server.send(204,"");
}




void logica_botao(){

 int reading = digitalRead(buttonPin);

 if (reading != lastButtonState){
  // reset the debouncing timer
  lastDebounceTime = millis();
}

 if ((millis() - lastDebounceTime) > debounceDelay) {

  if (reading != buttonState) {
    buttonState = reading;

    // only toggle the LED if the new button state is HIGH
    if (buttonState == LOW) {
    if(Banho == habilitado && Estado_Bot == desligado ){
      Estado_Bot = iniciar_banho;
       digitalWrite(rele, HIGH);

    }

    else if(Banho == habilitado && (Estado_Bot == iniciar_banho||Estado_Bot == continu
      Estado_Bot = pausar_banho;
      minutos_pausa=EEPROM.read(address_pausa);
      tempo_de_pausa = minutos_pausa*60;
      digitalWrite(rele, LOW);
    }

    else if(Banho == habilitado && Estado_Bot == pausar_banho ){
      Estado_Bot = continuar_banho;
      digitalWrite(rele, HIGH);
    }
    }
```

```
  }
}

 lastButtonState = reading;
  }


void logica_tempo(){
  if(Banho == habilitado && Estado_Bot == iniciar_banho){
    minutos=EEPROM.read(address_tempo);
    tempo = minutos*60;
    Estado_Bot = continuar_banho;
    }

    else if(Banho == habilitado && Estado_Bot == continuar_banho){
    tempo = tempo-1;
    if(tempo<60 && tempo>20){
      digitalWrite(Led_Aviso,HIGH);
      }
    else if(tempo < 20){

    if (led_S == LOW) {
    led_S = HIGH;
  } else {
    led_S = LOW;
  }
    digitalWrite(Led_Aviso, led_S);

      }
    }

    else if(Banho == habilitado && Estado_Bot == pausar_banho && tempo_de_pausa>0){
    tempo = tempo;
    tempo_de_pausa=tempo_de_pausa-1;
    }
    else if(Banho == habilitado && Estado_Bot == pausar_banho && tempo_de_pausa<=0){
    Banho = desabilitado;
    Estado_Bot = desligado;
    tempo=0;
    minutos_espera=EEPROM.read(address_espera);
    tempo_espera = (int)minutos_espera*60;
    digitalWrite(rele, LOW);
      }
```

```
    if(tempo == 0 && Banho == habilitado){
      Banho = desabilitado;
      Estado_Bot = desligado;
      minutos_espera=EEPROM.read(address_espera);
      tempo_espera = (int)minutos_espera*60;
      digitalWrite(rele, LOW);
      digitalWrite(Led_Aviso, LOW);
      }

    else if(tempo == 0 && Banho == desabilitado){
      if(tempo_espera > 0){
      tempo_espera = tempo_espera - 1;
      }
      else if(tempo_espera <= 0){
        Banho = habilitado;
        minutos=EEPROM.read(address_tempo);
        tempo = (int)minutos*60;
        minutos_espera=EEPROM.read(address_espera);
        tempo_espera=(int)minutos_espera*60;
        }
      }

    }



void setup() {

  pinMode(buttonPin, INPUT);
  pinMode(rele, OUTPUT);
  pinMode(Led_Aviso, OUTPUT);

  digitalWrite(rele, LOW);
  digitalWrite(Led_Aviso, LOW);


  botao.attach(0.2, logica_botao);
  tyme.attach(1, logica_tempo);


  delay(1000);
  Serial.begin(115200);
  Serial.println();
  Serial.print("Configuring access point...");
```

```
  /* You can remove the password parameter if you want the AP to be open. */
  WiFi.softAP(ssid, password);

  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(myIP);
  server.on("/", handleRoot);
  server.begin();
  Serial.println("HTTP server started");

    EEPROM.begin(512);

armazenado =EEPROM.read(address_tempo);
  if(armazenado>45 ||armazenado<1 ){
    minutos=7;
    EEPROM.write(address_tempo,minutos);
    EEPROM.commit();
    }
    else{
      minutos=armazenado;
      }

armazenado =EEPROM.read(address_espera);
   if(armazenado>45 ||armazenado<1 ){
    minutos_espera=5;
    EEPROM.write(address_espera,minutos_espera);
    EEPROM.commit();
    }
    else{
      minutos_espera=armazenado;
      }

armazenado =EEPROM.read(address_pausa);
    if(armazenado>45 ||armazenado<1 ){
    minutos_pausa=3;
    EEPROM.write(address_pausa,minutos_pausa);
    EEPROM.commit();
    }
    else{
      minutos_pausa=armazenado;
      }
```

```
}

void loop() {
  server.handleClient();
}
```

5 REFERÊNCIAS

[1] DIAGRAMA SENSOR ULTRASSÔNICO, disponível em: https://www.hackster.io/mpho_mphego/automated-standup-desk-640adf

[2] EXEMPLO ACCESS POINT ESP8266, disponível em: https://www.youtube.com/watch?v=fcmb_3aBfH4

[3] CURSO HTML, disponível em: https://www.youtube.com/watch?v=epDCjksKMoklist=PLHz_AreHm4dlAnJ_jJtV29RFxnPHDuk9o

[4] TEORIA DO SENSOR, disponível em: https://www.youtube.com/watch?v=ZejQOX69K5M

[5] DIY SENSOR ULTRASSÔNICO, disponível em: https://www.youtube.com/watch?v=e_cyxoqvWe8t=