FUNDAÇÃO ESCOLA DE COMÉRCIO ÁLVARES PENTEADO

– FECAP

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ALANIS CALHEIRA VIEIRA - 20021179

FERNANDA VIANA HIGINO - 22023553

GLENDA KAYLAINE HENRIQUE - 22023256

THAINA BEATRIZ DE SOUSA E SILVA - 22023297

VINICIUS LACERDA VIANA - 22023786

PROJETO INTERDISCIPLINAR

São Paulo

2022

ALANIS CALHEIRA VIEIRA - 20021179

FERNANDA VIANA HIGINO - 22023553

GLENDA KAYLAINE HENRIQUE - 22023276

THAINA BEATRIZ DE SOUSA E SILVA - 22023297
VINICIUS LACERDA VIANA - 22023786

PROJETO INTERDISCIPLINAR

Trabalho apresentado como parte dos requisitos necessários à obtenção de nota.

Orientador: David de Oliveira Lemes

São Paulo

2022

Sumário

1 Sobre o projeto

2 Por que um alimentador automático

3 Como será executado?

4 Como montar?

5 O que o código faz?

6 Código HTML

7 Código ESP

1 Sobre o projeto

A proposta do trabalho tem como objetivo prototipar um alimentador automático, a princípio, para cães e gatos.

O, “Hora do Lanche!” identifica através da hora inserida pelo usuário, e libera a comida para o pet, apenas nos horários selecionados.

2 Por que um alimentador automático?

Em 2018, o “Instituto Pet Brasil”, contabilizou cerca de 54,2 milhões de cães e 23,9 milhões de gatos como animais de estimação. Desde então o número vem crescendo, comprovando cada vez mais a busca por *pets* para fazer companhia nos lares, principalmente por parte das pessoas que moram sozinhas.

A tarefa de manter o pet bem alimentado se torna uma preocupação quando se tem uma rotina agitada, dessa forma, se faz necessário um alimentador automático, onde o usuário pode definir o horário e a quantidade de ração a ser despejada, mantendo a mente livre para se dedicar a correria do dia-a-dia e o seu companheiro de quatro patas saudável.

3 Como será executado?

Para a elaboração do alimentador, será usado um ESP8266, motor 28byj-48 e a biblioteca ntp. As informações serão salvas no ESP e os dados serão carregados para as variáveis corretas, iniciando o alimentador. A ração somente será liberada nas horas cadastradas pelo usuário.

Após ter definido o horário e a quantidade desejada, o ESP fará a verificação da hora através da internet e então o motor será ativado, girando a hélice acoplada para despejar a comida. A quantidade de giros da hélice controlará a comida que cai no comedouro.

4 Como montar?

O alimentador poderá ser montado em casa e com materiais de fácil acesso, possibilitando a aquisição para todos aqueles que desejam obter um para facilitar o seu dia e do seu animalzinho de estimação.

Os itens necessários para a construção são:

- Jumpers 3 Macho / Macho e 6 Macho / Fêmea.
- Motor de passo 28byj-48
- Drive ULN2003
- ESP8266
- Protoboard
- Arduino
- Fonte de 5Volts
- Cabo USB
- Resistor
- Cola
- Papelão
- Tesoura

A estrutura poderá ser feita com papelão ou com plástico (como caixas de sorvete ou caixas organizadoras, a escolha ficará a critério do cliente)

5 O que o código faz?

O ESP será conectado na internet, faz um upload da página e nela terá dois textboxs[1] para hora e minuto que só aceitam números e um botão para cada textbox. Essas informações são enviadas para duas variáveis no programa para serem salvas e dentro do ESP duas das bibliotecas são <NTPClient.h> e <WiFiUdp.h> que vão puxar um horario do site https://www.ntppool.org/pt/, e salvar em 3 variaveis int.

E depois que o usuário inserir o horário no código temos um if que vai comparar se a hora e o minuto digitado baterem com a hora, minuto e segundo pego do servidor e no caso dos segundos a razão dele é no if uma das comparações ela pede que pede que os segundo sejam = 0.
if(hourClient == currentHour && minuteClient == currentMinute && currentSecond == 0)

E dentro do if temos um for que realizar a ligação do motor e assim girando para soltar a comida.

```
for(int i = 0; i<10; i++){ myStepper.step(stepsPerRevolution); }
```

[1] Textbox: O TextBox controle fornece um estilo de formato único para texto exibido ou inserido no controle.

6 Código para o HTML

```
const char index_html[] PROGMEM = R"=====(
<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head>
        <meta charset="utf-8" />
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <script>
        function submitMessage() { //evia uma notificação que a informação foi salva no esp com sucesso
        alert("Sua informação foi salva no esp");
        setTimeout(function(){ document.location.reload(false); }, 500);
        }</script>
        <title>Hora Do Lanche</title>
        <style>
        body {
        font-family: Arial, Helvetica, sans-serif;
        background: linear-gradient(to right, rgb(29, 136, 207), rgb(110, 25, 143));
        text-align: center;
        color: #fff;
        }

        .container {
        position: absolute;
        top: 50%;
        left: 50%;
        transform: translate(-50%,-50%);
        width: 50%;
```

```
        background-color: rgba(0, 0, 0, 0.5);
        padding: 1em;
        border-radius: 10px;
        }

        button {
        background-color: slateblue;
        color: #fff;
        border: none;
        padding: 1em;
        border-radius: 10px;
        box-shadow: 1px 1px 6px black;
        cursor: pointer;
        }

        button:hover {
                background-color: steelblue;
        }

        .final-step,
        .second-step {
        display: none;
        }

        input {
        padding: 5px;
        border-radius: 5px;
        border: none;
        outline: none;
        }

        #resultado {
        font-size: 25px;
        }
        </style>
        <title>Bem vindo a HORA DO LANCHE</title>
</head>
```

```
<body>
        <p>Digite a Hora e o Minuto desejado para alimentação</p>
<form action="/get" target="hidden-form">
        Hora: <input type="number" name="inputHour" min="0" max="23" maxlength="2"
onkeypress="return (event.charCode !=8 && event.charCode ==0 || (event.charCode >=
48 && event.charCode <= 57))" >
        <input type="submit" value="Cadastrar" onclick="submitMessage()">
  </form><br>
  <form action="/get" target="hidden-form">
        Minutos: <input type="number" name="inputMinute" min="0" max="59"
maxlength="2" onkeypress="return (event.charCode !=8 && event.charCode ==0 ||
(event.charCode >= 48 && event.charCode <= 57))">
        <input type="submit" value="Cadastrar" onclick="submitMessage()">
  </form>
  <iframe style="display:none" name="hidden-form"></iframe>

</body>
</html>
)=====";
```

7 Código para o ESP

```
#include <Arduino.h> //Bibliotecas
#include <ESP8266WiFi.h>
#include <ESPAsyncTCP.h>
#include <ESPAsyncWebServer.h>
#include <Stepper.h>
#include <Wire.h>
#include "index_html.h"
#include <NTPClient.h>
#include <WiFiUdp.h>

AsyncWebServer server(80);

// Nome da rede wifi e senha
const char* ssid = "coloque aqui o nome da red";
const char* password = "e a senha dela";
```

```
//variaveis para puxar informação do html e transforma ela em int para poder ser comparada com a hora
const char* PARAM_Hour = "inputHour";
const char* PARAM_Minute = "inputMinute";

int hourClient;
int minuteClient;

String minuteC;
String hourC;

//Pinos para usar o Stepper D1 D5 D2 D6
const int stepsPerRevolution = 200;
Stepper myStepper(stepsPerRevolution, D1, D5, D2, D6);

//Define NTP
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org"); //Site para puxar a hora

void notFound(AsyncWebServerRequest *request) {
  request->send(404, "text/plain", "Not found");
}

void setup() {
  myStepper.setSpeed(60); //Define a velocidade do motor
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  if (WiFi.waitForConnectResult() != WL_CONNECTED) {
        Serial.println("Erro na conexão");
        return;
  }
  Serial.println();
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP());
```

```
  timeClient.begin(); //inicializa o NTP para pegar o tempo
  timeClient.setTimeOffset(-10800); //ajusta para o horario do brasil GMT -3, -3*60*60

  // Manda a pagina para a Web
  server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
        request->send_P(200, "text/html", index_html);
  });

  // Manda o pedido para o ESP
  server.on("/get", HTTP_GET, [] (AsyncWebServerRequest *request) {
        String inputMessage;
        // Pegar o input da hora
        if (request->hasParam(PARAM_Hour)) {
        hourC = request->getParam(PARAM_Hour)->value();

        }
        // Pegar o input do Minuto
        else if (request->hasParam(PARAM_Minute)) {
        minuteC = request->getParam(PARAM_Minute)->value();
        }
  });
  server.onNotFound(notFound);
  server.begin();
}

 void loop() {
  timeClient.update(); //
  hourClient = hourC.toInt(); // hora pega do site
  minuteClient = minuteC.toInt(); // minuto
  int currentHour = timeClient.getHours(); //hora pega do servidor
  int currentMinute = timeClient.getMinutes(); //minuto
  int currentSecond = timeClient.getSeconds(); //segundos
  Serial.print(hourClient); //essa linhas seguintes servem para mostra o horario do servidor
 e horaio inserido pelo usario
  Serial.print(":");
  Serial.print(minuteClient);
  Serial.println();
```

```
  Serial.println(currentHour);
  Serial.print(":");
  Serial.println(currentMinute);
  Serial.print(":");
  Serial.println(currentSecond);
  Serial.println();
  delay(1000);
  if(hourClient == currentHour && minuteClient == currentMinute && currentSecond == 0){
//if e for para quando a hora e o minuto forem igual aos inseridos no site
        for(int i = 0; i<10; i++){
        myStepper.step(stepsPerRevolution); //roda o motor
        yield(); //usado para nao deixar o watch dog resetar o eps
        }
        }
}
```