```
//----------------Bibliotecas-----------------------------
#include <RTClib.h>
#include <DHT.h>
#include <WiFi.h>
#include <WebServer.h>

//--------------------------Definição dos pinos-------------------------
#define RELE1_PIN 27
#define RELE2_PIN 26
#define RELE3_PIN 25
#define RELE4_PIN 32

#define SOIL_HUMIDITY_PIN A0 // Pino analógico para o sensor de umidade do solo
#define DHT_PIN 14      // Pino ao qual o sensor DHT11 está conectado
#define DHT_TYPE DHT11  // Tipo de sensor DHT
DHT dht(DHT_PIN, DHT_TYPE);
RTC_DS1307 rtc;

//--------------------Conexão na rede local sem fio---------------------
const char* ssid = "TP-LINK_1766";
const char* password = "95653662";
IPAddress ip(192, 168, 0, 112);
IPAddress gateway(192, 168, 1, 1); // Endereço do gateway
IPAddress subnet(255, 255, 255, 0); // Máscara de sub-rede
WebServer server(80);

//------------------------Variáveis globais-------------------------
int current_hour; // Será a hora de acordo com o rtc
int current_minute; // Variável para armazenar o minuto atual
int aux = 0;// usada na calibração
```

```
//Valores atuais recebido dos sensores
float soil_humidity = 0;
float air_temperature = 0;
float air_humidity = 0;
int leituraSeca = 0;
int leituraUmida = 0;

//horários
int last_hour = 0; //será atualizado quando a rega acontecer
int last_minute = 0;
int next_hour = 0; //será definido no site
int next_minute = 0;
bool regaAtiva = false; // Variável para controlar se a rega está ativa
unsigned long tempoInicioRega; // armazena o tempo em que a rega foi iniciada
unsigned long duracaoRega = 60000; // Duração da rega em milissegundos

//-------------------------função que controla a rega de acordo com as horas-------------------------
void LigarRega() {
  Serial.print("Verificando condição de rega: ");
  Serial.print(current_hour);
  Serial.print(":");
  Serial.print(current_minute);
  Serial.print(" - ");
  Serial.print(next_hour);
  Serial.print(":");
  Serial.println(next_minute);

  // Verifica se está na hora de ligar a rega
  if (current_hour == next_hour && current_minute == next_minute && !regaAtiva) {
    Serial.println("Rega ligada");
    digitalWrite(RELE1_PIN, LOW);
```

```
    delay(2000);
    digitalWrite(RELE2_PIN, LOW);
    tempoInicioRega = millis(); // Registra o tempo de início da rega
    regaAtiva = true;
    last_hour = current_hour;
    last_minute = current_minute;
  }

  // Verifica se a rega deve ser desligada após a duração especificada
  if (regaAtiva && millis() - tempoInicioRega >= duracaoRega) {
    Serial.println("Rega desligada");
    digitalWrite(RELE2_PIN, HIGH);
    delay(2000);
    digitalWrite(RELE1_PIN, HIGH);
    regaAtiva = false;
  }
}

//--------------------------Funções de controle manual-------------------------
void ligarRegaManual() {
  Serial.println("Rega manual ligada");
  digitalWrite(RELE1_PIN, LOW);
  delay(2000);
  digitalWrite(RELE2_PIN, LOW);
  regaAtiva = true;
  tempoInicioRega = millis();
  last_hour = current_hour;
  last_minute = current_minute;
}

void desligarRegaManual() {
```

```
  Serial.println("Rega manual desligada");
  digitalWrite(RELE2_PIN, HIGH);
  delay(2000);
  digitalWrite(RELE1_PIN, HIGH);
  regaAtiva = false;
}

//--------------------------Calibrando os sensores-------------------------
void calibrarSensorUmidadeSolo() {
  if (aux == 0){
    Serial.println("Insira o sensor em solo seco e aguarde...");
    delay(10000);
    leituraSeca = analogRead(SOIL_HUMIDITY_PIN);
    Serial.println("Leitura seca concluida");
    delay(1000);
    Serial.println("Insira o sensor em solo úmido e aguarde...");
    delay(10000);
    leituraUmida = analogRead(SOIL_HUMIDITY_PIN);
    Serial.println("Leitura umida concluida");
    delay(1000);
    Serial.println("Calibração concluída.");
    aux = 1;
  }
}

//--------------------------Atualiza os valores dos sensores--------------------------
void updateSensorsValue() {
  air_humidity = dht.readHumidity();
  air_temperature = dht.readTemperature();
  soil_humidity = analogRead(SOIL_HUMIDITY_PIN);
```

```
  if (isnan(air_humidity) || isnan(air_temperature)) {
    Serial.println("Falha na leitura do DHT!");
    return;
  }

  DateTime now = rtc.now();
  current_hour = now.hour();
  current_minute = now.minute();

  // Conversão do valor do sensor de umidade do solo para umidade percentual
  soil_humidity = map(soil_humidity, leituraSeca, leituraUmida, 0, 100);
  Serial.print("Temp: ");
  Serial.print(air_temperature);
  Serial.print(" Hum: ");
  Serial.print(air_humidity);
  Serial.print(" Soil: ");
  Serial.println(soil_humidity);
}

//----------------função que conecta na internet-----------------------------
void handleRoot() {
  updateSensorsValue(); // Atualiza os valores dos sensores
  // Inicia a criação da página HTML
  String page = "<html><head><title>Controle do Viveiro</title>";
  page += "<meta charset='UTF-8'>"; // Define a codificação de caracteres para UTF-8
  page += "<style>"; // Inicia a seção de estilo CSS
  page += "body { font-family: Arial, sans-serif; margin: 0; padding: 0; }"; // Estilo para o corpo da página
  page += ".container { padding: 20px; }"; // Estilo para o contêiner da página
  page += "h1 { background-color: #4CAF50; color: white; padding: 10px; }"; // Estilo para o cabeçalho
```

```
  page += ".section { border: 1px solid black; padding: 10px; margin-bottom: 20px; }"; // Estilo para as seções

  page += ".section h2 { margin-top: 0; }"; // Estilo para os subtítulos das seções

  page += "input[type='submit'] { background-color: #4CAF50; color: white; border: none; padding: 10px 20px; cursor: pointer; border-radius: 12px; }"; // Estilo para os botões de envio

  page += "input[type='submit']:hover { background-color: #45a049; }"; // Estilo para os botões de envio quando o mouse está sobre eles

  page += "</style>";

  page += "</head><body><div class='container'>";

  page += "<h1>Controle   do Viveiro de Mudas</h1>";


  // Seção de sensores

  page += "<div class='section'>";

  page += "<h2>Sensores</h2>";

  page += "<p>Temperatura do ar: " + String(air_temperature) + " °C</p>";

  page += "<p>Umidade do ar: " + String(air_humidity) + " %</p>";

  page += "<p>Umidade do solo: " + String(soil_humidity) + " %</p>";

  page += "</div>";


  // Seção de horários

  page += "<div class='section'>";

  page += "<h2>Horários</h2>";

  page += "<p>Hora atual: " + String(current_hour) + ":" + String(current_minute) + "</p>";

  page += "<p>Próxima rega às: " + String(next_hour) + ":" + String(next_minute) + "</p>";

  page += "<p>Última rega às: " + String(last_hour) + ":" + String(last_minute) + "</p>";

  page += "</div>";


  // Seção de controle de rega

  page += "<div class='section'>";

  page += "<h2>Controle de Rega</h2>";

  page += "<p>Rega ativa: " + String(regaAtiva ? "Sim" : "Não") + "</p>";

  page += "<p>Duração da rega: " + String(duracaoRega / 1000) + " segundos</p>";
```

```
  page += "</div>";

  // Seção de configuração
  page += "<div class='section'>";
  page += "<h2>Configuração</h2>";
  page += "<form action='/set_schedule' method='GET'>"; // Formulário para definir o próximo horário de rega
  page += "Próxima hora de rega: <input type='number' name='hour' min='0' max='23' value='" + String(next_hour) + "'><br>";
  page += "Próximo minuto de rega: <input type='number' name='minute' min='0' max='59' value='" + String(next_minute) + "'><br>";
  page += "Tempo de rega (segundos): <input type='number' name='time' min='1' value='" + String(duracaoRega / 1000) + "'><br>";
  page += "<input type='submit' value='Atualizar horário de rega'>";
  page += "</form>";
  page += "</div>";

  // Seção de controle manual
  page += "<div class='section'>";
  page += "<h2>Rega Manual</h2>";
  page += "<form action='/ligar_rega_manual' method='POST'>";
  page += "<input type='submit' value='Ligar Rega Manual'>";
  page += "</form>";
  page += "<form action='/desligar_rega_manual' method='POST'>";
  page += "<input type='submit' value='Desligar Rega Manual'>";
  page += "</form>";
  page += "</div>";

  page += "</div></body></html>";
  server.send(200, "text/html", page); // Envia a página HTML para o cliente
}
```

```
//----------------função que atualiza o horário de rega-----------------------------
void handleSetSchedule() {
  if (server.hasArg("hour") && server.hasArg("minute") && server.hasArg("time")) {
    next_hour = server.arg("hour").toInt();
    next_minute = server.arg("minute").toInt();
    duracaoRega = server.arg("time").toInt() * 1000;
  }
  server.sendHeader("Location", "/");
  server.send(303);
}

//----------------função que controla a função manual da rega-----------------------------
void handleLigarRegaManual() {
  ligarRegaManual();
  server.sendHeader("Location", "/");
  server.send(303);
}

void handleDesligarRegaManual() {
  desligarRegaManual();
  server.sendHeader("Location", "/");
  server.send(303);
}

//----------------setup-----------------------------
void setup() {
  Serial.begin(115200);

  //--------------------------Definição os pinos--------------------------
  pinMode(RELE1_PIN, OUTPUT);
  pinMode(RELE2_PIN, OUTPUT);
```

```
pinMode(RELE3_PIN, OUTPUT);
pinMode(RELE4_PIN, OUTPUT);
pinMode(SOIL_HUMIDITY_PIN, INPUT);
        digitalWrite(RELE1_PIN, HIGH);
        digitalWrite(RELE2_PIN, HIGH);
        digitalWrite(RELE3_PIN, HIGH);
        digitalWrite(RELE4_PIN, HIGH);
calibrarSensorUmidadeSolo();

//------------------------------Inicialização DHT e RTC------------------------------
Serial.println("Inicializando o sensor DHT...");
dht.begin(); // Inicializa o sensor DHT
delay(2000);
if (!rtc.begin()) {
  Serial.println("Erro ao iniciar o RTC!");
  while (1);
}

//------------------------------Conexão WIFI------------------------------
WiFi.config(ip, gateway, subnet);
WiFi.begin(ssid, password);
Serial.println("Conectando ao WiFi...");
while (WiFi.status() != WL_CONNECTED) {
  delay(1000);
  Serial.print(".");
}
Serial.println();
Serial.println("WiFi conectado");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
```

```
  server.on("/", handleRoot);
  server.on("/set_schedule", handleSetSchedule);
  server.on("/ligar_rega_manual", HTTP_POST, handleLigarRegaManual);
  server.on("/desligar_rega_manual", HTTP_POST, handleDesligarRegaManual);
  server.begin();
  Serial.println("Servidor HTTP iniciado");
}

//----------------loop-----------------------------
void loop() {
  updateSensorsValue();
  LigarRega();
  server.handleClient();
  delay(1000);
}
```