Academia de Ensino Superior de Mafra
Cursos Técnicos Superiores Profissionais (CTESP)

ipt Instituto Politécnico de Tomar

PROJECTO INTEGRADO

AUTOR(ES):

RUI BAPTISTA – 81752

TIAGO DIAS - 81748

DISCIPLINA:

PROJECTO INTEGRADO

ACADEMIA DE ENSINO SUPERIOR DE MAFRA, 15 de janeiro de 2023

Índice

1 - RESUMO/ABSTRACT

Este trabalho focou-se em automatizar as rotinas principais de uma exploração avícola, mas que poderá ser extensível a muitas outras áreas, através de IoT.

Usou-se um micro controlador e sensores genéricos para apresentar uma solução de automação com baixo investimento. Deixa também em aberto a hipótese de acrescentar novas funcionalidades conforme será sugerido ao longo deste documento.

2 - INTRODUÇÃO

O objectivo inicial do projecto foi o desenvolvimento de uma solução completa para a automatização de uma exploração avícola, mas que facilmente permite automatizar qualquer outro tipo de exploração agrícola.

Envolve as unidade curriculares de:

- IoT
- Redes
- Sistemas Operativos
- Administração de sistemas e serviços
- Programação
- Tecnologias de Internet

Este projecto teve também como pretensão apresentar uma solução de automação que fosse flexível o suficiente para poder ser adaptada a outras necessidades.

Neste caso concreto afim de manter a exploração a funcionar, as luzes serão ligadas quando escurecer, o aquecimento será ligado quando a temperatura baixar da temperatura definida e o telhado será aberto quando exceder outro valor que também é definível. Fica de fora nesta primeira versão os aspersores que irão na próxima evolução permitir aumentar a humidade no ar caso este esteja muito seco e/ou quente.

3 - DESENVOLVIMENTO

Este projecto divide-se em varias áreas de desenvolvimento.

A parte do projecto de IoT tem como componentes principais o ESP32, controlador este que foi escolhido em virtude de ter sido trabalhado em contexto de aula e logo estarmos mais familiarizados com ele e alem disso tem boas capacidades de comunicação por WiFi e uma boa relação preço/capacidade.

O ESP32 vai processar os sinais de entrada dos sensores de temperatura/humidade(DHT11) e do sensor de luminosidade (LDR). O sensor DHT11 foi escolhido logo no inicio do planeamento deste projecto em virtude de ser um sensor com um custo baixo mas que por sua vez também de resolução baixa, logo numa próxima evolução do projecto este sensor será substituído por o BME280 que além de medir a temperatura e a humidade com mais fiabilidade, mede também a pressão atmosférica.

Em termos de saídas nesta primeira versão do projecto o ESP32 fará actuar, através de sinais de output, a iluminação, o aquecimento e o motor que abrirá o tecto.

A iluminação, como sabemos o valor de luminosidade (através do sensor de luminosidade) , é activada em dois blocos, sendo na próxima evolução do projecto actuada por percentagem, bastando para isso que as lâmpadas suportem a função DIM, conseguindo-se assim uma poupança energética significativa alem de um ambiente mais adequado aos animais.

O aquecimento será activado através de um relay, podendo também mais tarde ser escalado em várias fases, sendo o valor de temperatura que faz activar esta função programável no código através da variável “heatOn”. A possibilidade de o aquecimento poder ser dividido em várias fases permite uma maior economia de energia, permitindo assim que ele seja accionado consoante a temperatura esteja baixa ou muito baixa.

O telhado será aberto através de um servo motor que permite um posicionamento exacto evitando assim neste primeira fase do projecto ter informação adicional da posição em que se encontra o telhado, mas que na próxima evolução será essencial afim de prevenir falhas do sistema.

Todas estas constantes que precisam de ser definidas para que o ESP possa saber os valores a partir dos quais ele vai executar uma acção estão definidas nesta primeira versão "hardcoded" no inicio do programa, mas que na próxima versão será possível alterar através da aplicação móvel, algo que não foi possível na janela de tempo existente para esta primeira versão.

```
const int tempOpenRoof = 20;

const int heatOn = 20;
const int lightOn = 750;
const int readingSensorInterval = 1500;
const int roofOpeningAngle = 40;
```

A nível de comunicação com a base de dados optou-se nesta primeira versão por um método HTTP (GET), que consiste em enviar através de um endereço URL (como se pode ver no exemplo mais abaixo) o valor das variáveis que queremos actualizar na tabela da base de dados, sendo que estes dados são inseridos na tabela já do lado do servidor através de um script PHP.

http://192.168.1.100/db/arduino.php?
temp=19.00&humidade=70.00&sensorLuz=752&luzes=0&aquecimento=1&telhado=0

Este é o código PHP usado para comunicação com o SQL Server

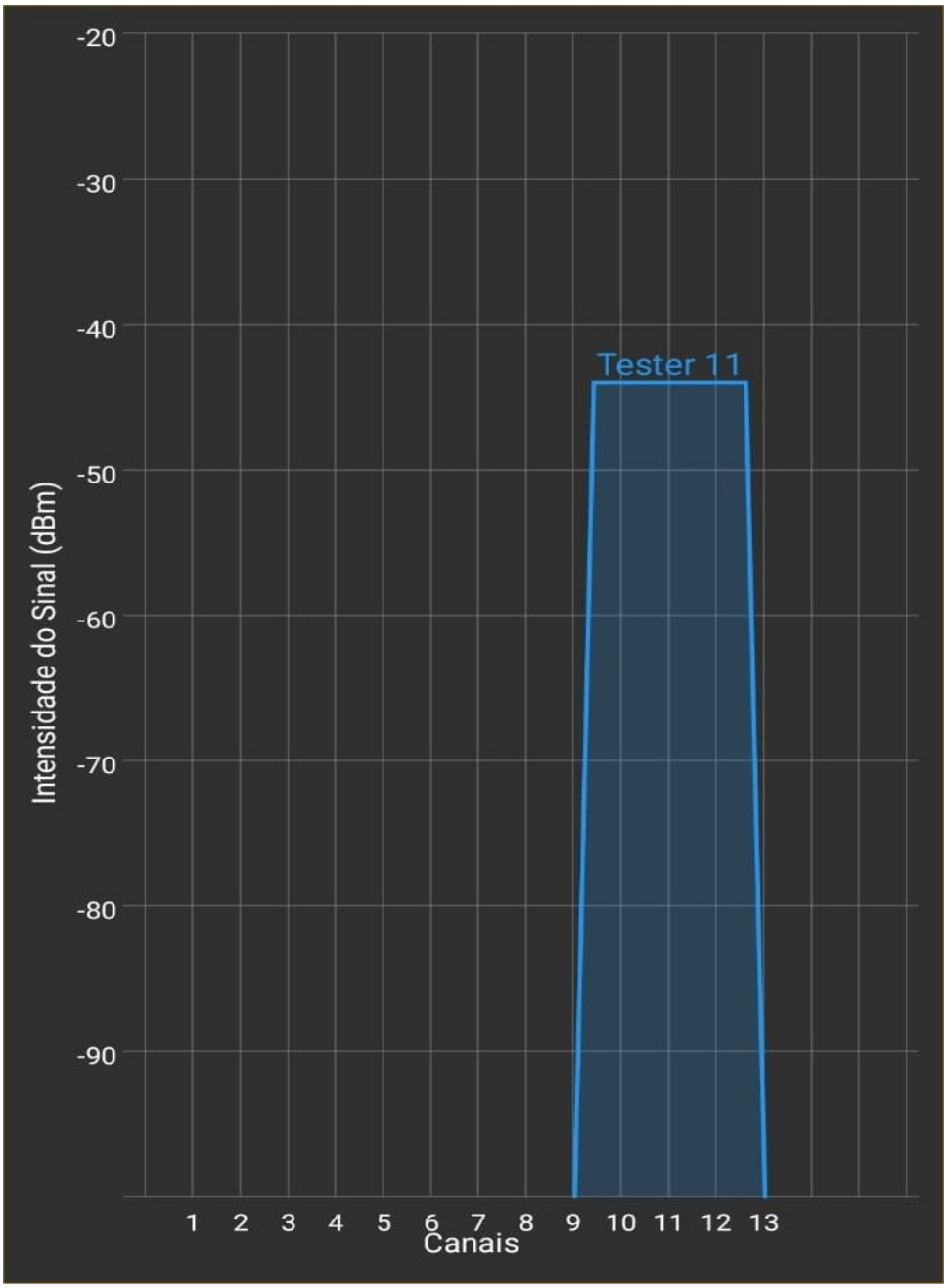
```
<?php
include_once('config.php');

// Obter os parâmetros da requisição GET
$temp=$_GET['temp'];
$humidade=$_GET['humidade'];
$sensorLuz=$_GET['sensorLuz'];
$luzes=$_GET['luzes'];
$aquecimento=$_GET['aquecimento'];
$telhado=$_GET['telhado'];
```

```
// Prepara a query SQL para selecionar as leituras mais recentes
$sql = 'INSERT INTO info(temp,humidade,sensorLuz,luzes,aquecimento,telhado)
VALUES(' .$temp. ',' .$humidade. ',' .$sensorLuz. ',' .$luzes. ',' .
$aquecimento. ',' .$telhado. ')';

// Estabelece uma conexão com a base de dados
$conn = connection_db();

// Executa a query SQL e armazena o resultado em uma variável
$resultado = sqlsrv_query($conn, $sql);
sqlsrv_free_stmt($resultado);
sqlsrv_close($conn);

//?temp=55&humidade=55&sensorLuz=0&luzes=0&aquecimento=0&telhado=0
?>
```

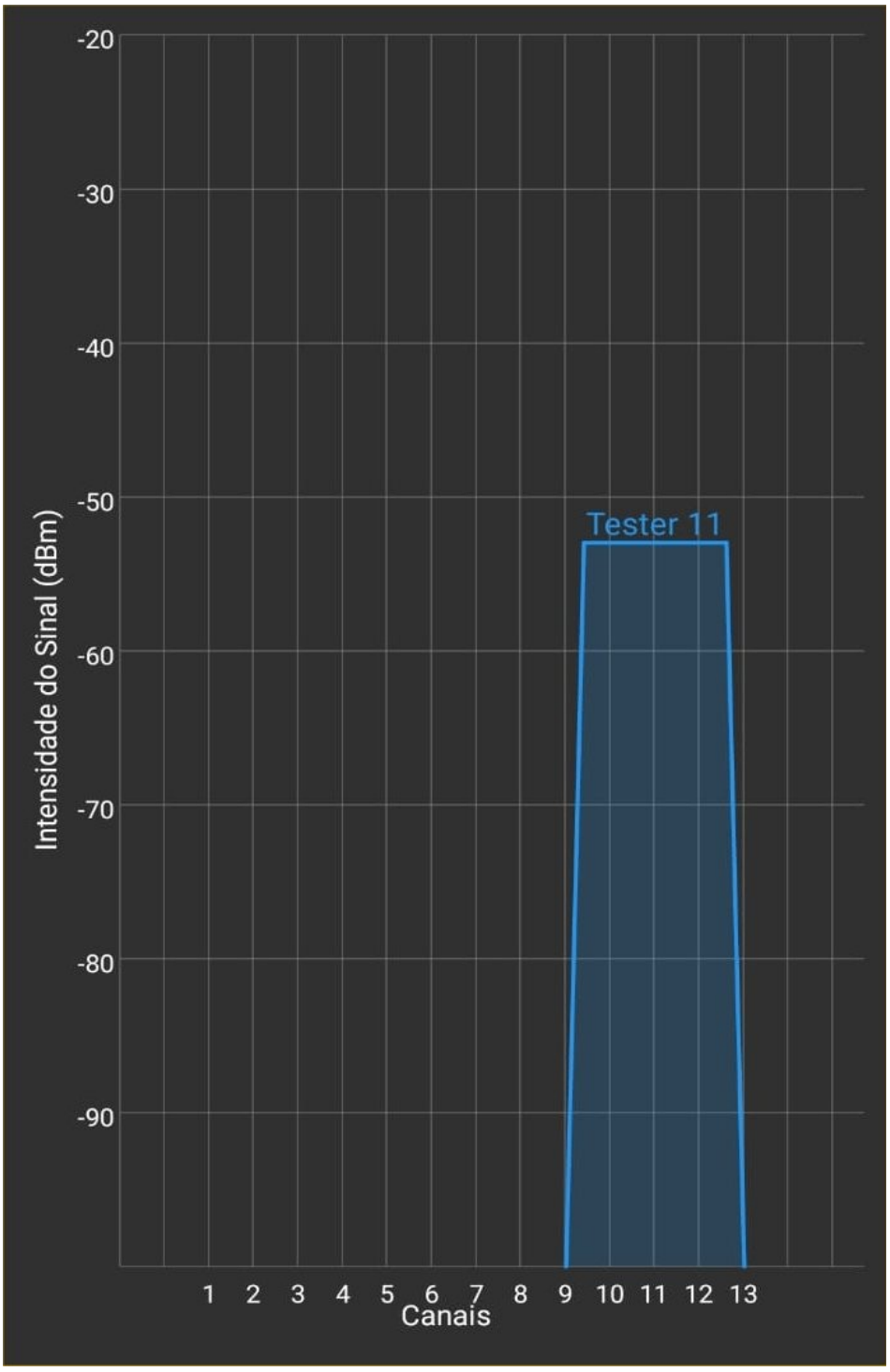
Mais uma vez, na próxima versão também este método sofrerá alterações sendo substituído por uma comunicação MQTT, que é mais eficiente na transmissão de dados e na própria gestão da ligação que é estabelecida cada vez que é enviado um pacote de dados.

A nível de ligações de rede e no contexto para onde este projecto foi desenvolvido usou-se o WiFi visto que onde será implementado consiste em pavilhões abertos com poucos obstáculos à propagação do sinal, bastando por isso que o Access Point seja instalado a meio do pavilhão tanto em altura como em largura, afim do sinal ser propagado da maneira mais eficiente possível, reduzindo-se assim tanto o numero de Access Points como de cablagem necessária.

A correcta localização do equipamento de WiFi é suficiente para ter ganhos expressivos na quantidade de sinal, devido à reflexão do sinal no tecto ou chão, fazendo com que haja atenuação de sinal conforme se mostra nos próximos dois gráficos.



Neste gráfico o Access Point foi instalado afastado da parede e a meia distancia entre o tecto e o chão.



Neste gráfico o Access Point foi instalado encostado à parede e ao tecto .

Do lado do servidor foi instalado o Windows Server 2019 com Web Server que será uma das alternativas para visualizar os dados do ESP32 apenas com o delay de tempo que está definido no código do ESP32 para envio dos dados dos sensores.

Foi necessário também instalar o PHP no Windows Server de modo a conseguir utilizar scripts.

A base de dados optou-se por utilizar o Microsoft SQL Server Express, tanto por ser gratuito como por ser uma solução com muita utilização no meio empresarial , logo com bom suporte.

Além disso ao utilizarmos o SQL Server e sendo este ainda pouco utilizado com o ESP32, foi um desafio extra conseguir introduzir os dados dos sensores na base de dados, devido ao facto de a maioria da literatura existente ser para o MySQL inclusive com bibliotecas já preparadas, algo que aquando da redação deste documento não estava disponível para o SQL Server.

A estrutura da tabela implementada no SQL Server é a seguinte :

```
[id] [int] IDENTITY(1,1) NOT NULL,
[temp] [float] NULL,
[humidade] [float] NULL,
[sensorLuz] [bit] NULL,
[luzes] [bit] NULL,
[aquecimento] [bit] NULL,
[telhado] [bit] NULL,
[reading_time] [datetime] NOT NULL
```

Para mostrar os valores da base de dados na pagina Web criada para o efeito utiliza-se um script PHP que vai ler a tabela da base de dados e mostrar os resultados em HTML conforme podemos ver na imagem abaixo.

ID	Temperatura	humidade	aquecimento	sensorluz	luzes	telhado	Data/horas
504	19.4°C	64%	on	on	off	off	2023-03-14 11:49:47
503	19.4°C	65%	on	off	on	off	2023-03-14 11:49:32
502	19.8°C	67%	on	on	off	off	2023-03-14 11:49:17
501	20.2°C	70%	off	on	off	on	2023-03-14 11:49:02
500	20.2°C	71%	off	on	off	on	2023-03-14 11:48:47
499	20.6°C	71%	off	on	off	on	2023-03-14 11:48:32

Esta parte do script PHP é a mais relevante:

```
<?php
    include_once('db/config.php'); // Inclui o arquivo config.php que contém as informações de configuração da base de dados.
    if (isset($_GET["readingsCount"])){ // Verifica se a variável readingsCount foi definida na URL.
     $dados = $_GET["readingsCount"]; // Atribui o valor de readingsCount a uma variável.
     $dados = trim($dados); // Remove espaços em branco do início e do fim da string.
     $dados = stripslashes($dados); // Remove barras invertidas adicionadas por addslashes().
     $dados = htmlspecialchars($dados); // Converte caracteres especiais em entidades HTML.
      $readings_count = $_GET["readingsCount"]; // Atribui o valor de readingsCount a uma variável.
    }
   else {
```

```
    $readings_count = 20; // Define o valor padrão para readingsCount se não foi definido na URL.
  }
    $last_reading = getLastReadings(); // Obtém a última leitura da tabela info da base de dados.
    $last_reading_temp = $last_reading["temp"]; // Obtém o valor da temperatura da última leitura.
   $last_reading_humi = $last_reading["humidade"]; // Obtém o valor da humidade da última leitura.
    $last_reading_time = $last_reading["reading_time"]; // Obtém o valor da data/hora da última leitura.


  // Converte a variável $last_reading_time em uma string PHP.
  $last_reading_time = $last_reading_time->format("Y-m-d H:i:s");
  // Obtém as estatísticas de temperatura e humidade para o número de leituras especificado.
  $min_temp = minReading($readings_count, 'temp');
  $max_temp = maxReading($readings_count, 'temp');
  $avg_temp = avgReading($readings_count, 'temp');
  $min_humi = minReading($readings_count, 'humidade');
  $max_humi = maxReading($readings_count, 'humidade');
  $avg_humi = avgReading($readings_count, 'humidade');
?>


// Prepara a query SQL para selecionar as leituras mais recentes
$sql = 'SELECT TOP '.$readings_count.' * FROM info ORDER BY reading_time DESC';
// Estabelece uma conexão com a base de dados
$conn = connection_db();
// Executa a query SQL e armazena o resultado em uma variável
$resultado = sqlsrv_query($conn, $sql);
// Verifica se a query foi executada com sucesso
if ($resultado) {
  // Percorre cada linha do resultado e exibe as informações na tabela HTML
  while ($row = sqlsrv_fetch_array($resultado, SQLSRV_FETCH_ASSOC)) {
    // Armazena os valores de cada coluna em variáveis separadas para facilitar o acesso
```

```
        $row_id = $row["id"];
        $row_temp = $row["temp"];
        $row_humidade = $row["humidade"];
        $row_reading_time = $row["reading_time"]->format("Y-m-d H:i:s");
        // Converte os valores de 0/1 em "on" ou "off" para exibir na tabela
        $aquecimento = intval($row['aquecimento']) == 0 ? "off" : "on";
        $sensorluz = intval($row['sensorLuz']) == 0 ? "off" : "on";
        $luzes = intval($row['luzes']) == 0 ? "off" : "on";
        $telhado = intval($row['telhado']) == 0 ? "off" : "on";
        // Exibe os valores na tabela HTML
        echo '<tr>
                <td>' . $row_id . '</td>
                <td>' . $row_temp . '&deg;C</td>
                <td>' . $row_humidade . '%</td>
                <td>' . $aquecimento . '</td>
                <td>' . $sensorluz . '</td>
                <td>' . $luzes . '</td>
                <td>' . $telhado. '</td>
                <td>' . $row_reading_time . '</td>
              </tr>';
    }
    // Fecha a tag da tabela HTML
    echo '</table>';
    // Liberta o resultado da query e fecha a conexão com a base de dados
    sqlsrv_free_stmt($resultado);
    sqlsrv_close($conn);
}
?>
<script>
// Definição da função a ser executada quando a página é carregada
window.onload = function(){
    // Esconde o elemento com o ID "ring"
```

```
    document.getElementById("ring").style.display = "none";
    // Exibe o elemento com o ID "page"
    document.body.style.display = "block";
}
// Leitura dos valores da temperatura e humidade obtidos via PHP
var temp = <?php echo $last_reading_temp; ?>;
var humidade = <?php echo $last_reading_humi; ?>;

// Chamada das funções para exibir os valores obtidos no ecra
setTemperature(temp);
setHumidity(humidade);
// Definição da função para exibir o valor da temperatura
function setTemperature(curVal){
    // Definição dos valores mínimo e máximo para a escala da temperatura em Celsius
    var minTemp = -5.0;
    var maxTemp = 38.0;
    // Definição dos valores mínimo e máximo para a escala da temperatura em Fahrenheit
    // var minTemp = 23;
    // var maxTemp = 100;
    // Escalonamento do valor da temperatura obtido para a escala de exibição no ecra
    var newVal = scaleValue(curVal, [minTemp, maxTemp], [0, 180]);
    // Atualização da exibição da temperatura no ecra
    $('.gauge--1 .semi-circle--mask').attr({
        style: '-webkit-transform: rotate(' + newVal + 'deg);' +
        '-moz-transform: rotate(' + newVal + 'deg);' +
        'transform: rotate(' + newVal + 'deg);'
    });
    $("#temp").text(curVal + ' ºC');
}
// Definição da função para exibir o valor da humidade
function setHumidity(curVal){
    // Definição dos valores mínimo e máximo para a escala da humidade
```

```
    var minHumi = 0;
    var maxHumi = 100;
    // Escalonamento do valor da humidade obtido para a escala de exibição na tela
    var newVal = scaleValue(curVal, [minHumi, maxHumi], [0, 180]);
    // Atualização da exibição da humidade na tela
    $('.gauge--2 .semi-circle--mask').attr({
        style: '-webkit-transform: rotate(' + newVal + 'deg);' +
        '-moz-transform: rotate(' + newVal + 'deg);' +
        'transform: rotate(' + newVal + 'deg);'
    });
    $("#humi").text(curVal + ' %');
}

// Definição da função para escalonar o valor da temperatura ou humidade
function scaleValue(value, from, to) {
    // Cálculo do fator de escala para o valor a ser escalonado
    var scale = (to[1] - to[0]) / (from[1] - from[0]);
    // Limitação do valor a ser escalonado dentro dos limites de escala definidos
    var capped = Math.min(from[1], Math.max(from[0], value)) - from[0];
    // Escalonamento do valor propriamente dito
    return ~~(capped * scale + to[0]);
```

Foi criada também uma aplicação em android studio que permite ver no dispositivo móvel os dados da base de dados. Na próxima evolução da aplicação ela vai permitir alterar os parâmetros definidos no ESP32 evitando assim que tenha de ser alterado no código.

4 - CONCLUSÕES

Este projecto foi interessante na medida que foram envolvidos conhecimentos em varias áreas para realizar o trabalho com o respetivo desenvolvimento e aprendizagem.

Tem potencial para continuar o desenvolvimento consoante o que foi referido neste documento com possibilidade de implementação em várias áreas de negocio.

5 - ANEXOS

GRÁFICO GANTT

	M de Ta	Nome da Tarefa	Duração	Início	Conclusão	Predecessoras	Nomes de Recursos
1		**Projecto - Automação de exploração avicola**	**44 dias**	**Ter 03/01/23**	**Sex 03/03/23**		
2		**Planeamento**	**7 dias**	**Seg 09/01/23**	**Ter 17/01/23**		
3		Pesquisa dos recursos necessarios	5 dias	Seg 09/01/23	Sex 13/01/23		
4		Pedido de orcamentos	2 dias	Seg 16/01/23	Ter 17/01/23	3	
5		**Aquisição dos recursos**	**3 dias**	**Qua 18/01/23**	**Sex 20/01/23**		
6		Encomenda do material IoT	1 dia	Qua 18/01/23	Qua 18/01/23	4	
7		Encomenda do material para a maquete	2 dias	Qui 19/01/23	Sex 20/01/23	4	
8		**Montagem de todos os recursos necessários**	**5 dias**	**Seg 23/01/23**	**Sex 27/01/23**		
9		Elaboração do espaço de trabalho	1 dia	Seg 23/01/23	Seg 23/01/23		
10		Montagem da Maquete	2 dias	Ter 24/01/23	Qua 25/01/23	7	
11		Montagem do ESP32 e sensores	2 dias	Qui 26/01/23	Sex 27/01/23	6	
12		**Programação IoT**	**8 dias**	**Seg 30/01/23**	**Qua 08/02/23**		
13		Elaboração do codigo ESP32 para leitura de sensores	4 dias	Seg 30/01/23	Qui 02/02/23	11	
14		Elaboração do codigo ESP32 para accionamento das saidas	4 dias	Sex 03/02/23	Qua 08/02/23	11	
15		**Instalação de Recursos do Server**	**3 dias**	**Seg 20/02/23**	**Qua 22/02/23**		
16		Instalação e configuração do Windows Server 2019	1 dia	Qui 09/02/23	Qui 09/02/23		
17		Instalação e configuração do MYSQL	1 dia	Sex 10/02/23	Sex 10/02/23	16	
18		Projectar e implementar as tabelas necessarias	1 dia	Sex 10/02/23	Sex 10/02/23	17	
19		Implementar os serviços WEB SERVER	1 dia	Sex 10/02/23	Sex 10/02/23	16	
20		**Programação do Software necessário**	**9 dias**	**Seg 13/02/23**	**Qui 23/02/23**		
21		Elaboração do software de integração do projecto	7 dias	Seg 13/02/23	Ter 21/02/23		
22		Elaboração da web page para cliente final	2 dias	Qua 22/02/23	Qui 23/02/23	19	
23		Integração de todos os modulos	4 dias	Sex 24/02/23	Qua 01/03/23	13;14;21;22	
24		Redação do relatorio	4 dias	Ter 28/02/23	Sex 03/03/23		

Planeamento do projecto executado no Microsoft Project e revisto semanalmente.

Código usado para o ESP32:

```
// Import required libraries
#include "WiFi.h"
#include "ESPAsyncWebServer.h"
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <Servo.h>
#include <HTTPClient.h>

// Replace with your network credentials
const char* ssid = "Tester"; //Tester
const char* password = "ipt12345";  //ipt12345

const int lightsPin = 5; // pin for lights
const int heatPin = 23;  // pin  for heating
const int sensorPin = 33; // pin for light sensor

const int tempOpenRoof = 20;
const int heatOn = 20;
const int lightOn = 750;
const int readingSensorInterval = 15000;
const int roofOpeningAngle = 60;

int luzes = 0;
int aquecimento =0;
int telhado = 0;

int lightVal;   // light reading

Servo myservo; // create servo object to control a servo
int pos = 0;    // variable to store the servo position
```

```
#define DHTPIN 27     // Digital pin connected to the DHT sensor

#define DHTTYPE    DHT11     // DHT 11
//#define DHTTYPE    DHT22     // DHT 22 (AM2302)
//#define DHTTYPE    DHT21     // DHT 21 (AM2301)

DHT dht(DHTPIN, DHTTYPE);

String readDHTTemperature() {
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  // Read temperature as Celsius (the default)
  float temp = dht.readTemperature();
  // Check if any reads failed and exit early (to try again).
  if (isnan(temp)) { Serial.println("Failed to read from DHT sensor!"); return "--"; }
  else { Serial.println(temp); return String(temp); }
}

String readDHTHumidity() {
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  if (isnan(h)) { Serial.println("Failed to read from DHT sensor!"); return "--"; }
  else { Serial.println(h); return String(h); }
}

void setup(){

  // Serial port for debugging purposes
  pinMode (lightsPin, OUTPUT);
  pinMode (heatPin, OUTPUT);

  myservo.attach(13);  // attaches the servo on pin 13 to the servo object

  Serial.begin(115200);
```

```
  dht.begin();

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi..");
  }

  // Print ESP32 Local IP Address
  Serial.println(WiFi.localIP());

}

void loop()
{
  //Check WiFi connection status
  if(WiFi.status()== WL_CONNECTED){
    WiFiClient client;

  float temp = dht.readTemperature();
  if (temp < heatOn )
  {
    digitalWrite (heatPin, HIGH);
    aquecimento = 1;
  }
  else
  {
    digitalWrite (heatPin, LOW);
    aquecimento = 0;
  }

  float humidade = dht.readHumidity();
```

```
 lightVal = analogRead(sensorPin); // read the current light levels

 if(lightVal  <  lightOn ) {
    digitalWrite (lightsPin, HIGH); // turn on light
    luzes = 1;
 }
 //otherwise, it is bright
 else {
  digitalWrite (lightsPin, LOW); // turn off light
  luzes = 0;
 }

// test lines to ldr sensor response
// reads the input on analog pin (value between 0 and 4095)
 int analogValue = analogRead(sensorPin);

 Serial.print("Analog Value = ");
 Serial.print(analogValue);   // the raw analog reading

 if (temp < tempOpenRoof )
 {
  myservo.write(0);
  telhado =0;
 }
 else
 {
  myservo.write(roofOpeningAngle);
  telhado =1;
 }

 HTTPClient http;

 lightVal = analogRead(sensorPin); // read the current light levels
```

```
  String url = "http://192.168.1.100/db/arduino.php?";
  url += "temp="+ String(temp);
  url += "&humidade="+ String(humidade);
  url += "&sensorLuz="+String(lightVal);
  url += "&luzes="+String(luzes);
  url += "&aquecimento="+String(aquecimento);
  url += "&telhado="+String(telhado);

  Serial.println(url);

  http.begin(url);

  int httpCode = http.GET();

  if (httpCode > 0) {
    String payload = http.getString();
    Serial.println(payload);
  } else {
    Serial.println("Falha na requisição");
  }
  http.end();

  delay(readingSensorInterval);
  }
}
```

Script PHP para introdução de dados na base de dados:

```
<?php
include_once('config.php');
// Obter os parâmetros da requisição GET
$temp=$_GET['temp'];
$humidade=$_GET['humidade'];
$sensorLuz=$_GET['sensorLuz'];
$luzes=$_GET['luzes'];
$aquecimento=$_GET['aquecimento'];
$telhado=$_GET['telhado'];
// Prepara a query SQL para selecionar as leituras mais recentes
$sql = 'INSERT INTO info(temp,humidade,sensorLuz,luzes,aquecimento,telhado) VALUES('
.$temp. ',' .$humidade. ',' .$sensorLuz. ',' .$luzes. ',' .$aquecimento. ',' .$telhado. ')';
// Estabelece uma conexão com o banco de dados
$conn = connection_db();
// Executa a query SQL e armazena o resultado em uma variável
$resultado = sqlsrv_query($conn, $sql);
sqlsrv_free_stmt($resultado);
sqlsrv_close($conn);
//?temp=55&humidade=55&sensorLuz=0&luzes=0&aquecimento=0&telhado=0
?>
```

Script PHP para visualização da base de dados:

```
<?php
   include_once('db/config.php'); // Inclui o arquivo config.php que contém as informações de configuração da base de dados.
   if (isset($_GET["readingsCount"])){ // Verifica se a variável readingsCount foi definida na URL.
    $dados = $_GET["readingsCount"]; // Atribui o valor de readingsCount a uma variável.
    $dados = trim($dados); // Remove espaços em branco do início e do fim da string.
    $dados = stripslashes($dados); // Remove barras invertidas adicionadas por addslashes().
    $dados = htmlspecialchars($dados); // Converte caracteres especiais em entidades HTML.
     $readings_count = $_GET["readingsCount"]; // Atribui o valor de readingsCount a uma variável.
   }
   else {
     $readings_count = 20; // Define o valor padrão para readingsCount se não foi definido na URL.
   }
   $last_reading = getLastReadings(); // Obtém a última leitura da tabela info da base de dados.
   $last_reading_temp = $last_reading["temp"]; // Obtém o valor da temperatura da última leitura.
   $last_reading_humi = $last_reading["humidade"]; // Obtém o valor da humidade da última leitura.
   $last_reading_time = $last_reading["reading_time"]; // Obtém o valor da data/hora da última leitura.
```

```
    // Converte a variável $last_reading_time em uma string PHP.
    $last_reading_time = $last_reading_time->format("Y-m-d H:i:s");
    // Obtém as estatísticas de temperatura e humidade para o número de leituras especificado.
    $min_temp = minReading($readings_count, 'temp');
    $max_temp = maxReading($readings_count, 'temp');
    $avg_temp = avgReading($readings_count, 'temp');
    $min_humi = minReading($readings_count, 'humidade');
    $max_humi = maxReading($readings_count, 'humidade');
    $avg_humi = avgReading($readings_count, 'humidade');
?>
<!DOCTYPE html>
<html>
    <head><meta http-equiv="Content-Type" content="text/html; charset=utf-8">
        <link rel="stylesheet" type="text/css" href="css/style.css">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <link rel="icon" href="images/chiken.ico">
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
        <title>Chiken</title>
    </head>
    <div id="ring"></div>
    <header class="header">
        <h1>📊 Chicken informações</h1>
        <form method="get">
            <input type="number" name="readingsCount" min="1" placeholder="Número de
leituras (<?php echo $readings_count; ?>)">
```

```
        <input type="submit" value="UPDATE">
      </form>
  </header>
<body>
  <p>Ultima leitura ás: <?php echo $last_reading_time; ?></p>
  <section class="content">
          <div class="box gauge--1">
          <h3>TEMPERATURA</h3>
         <div class="mask">
                      <div class="semi-circle"></div>
                      <div class="semi-circle--mask"></div>
                    </div>
                <p style="font-size: 30px;" id="temp">--</p>
                <table cellspacing="5" cellpadding="5">
                  <tr>
                    <th colspan="3"><?php echo $readings_count; ?> leituras de
Temperaturas  </th>
                </tr>
                    <tr>
                      <td>Min</td>
              <td>Max</td>
              <td>Média</td>
            </tr>
            <tr>
              <td><?php echo $min_temp; ?> &deg;C</td>
```

```
            <td><?php echo $max_temp; ?> &deg;C</td>
            <td><?php echo round($avg_temp, 2); ?> &deg;C</td>
        </tr>
    </table>
</div>
<div class="box gauge--2">
    <h3>HUMIDADE</h3>
    <div class="mask">
        <div class="semi-circle"></div>
        <div class="semi-circle--mask"></div>
    </div>
    <p style="font-size: 30px;" id="humi">--</p>
    <table cellspacing="5" cellpadding="5">
        <tr>
            <th colspan="3"> <?php echo $readings_count; ?> leituras de humidade</th>
        </tr>
        <tr>
            <td>Min</td>
            <td>Max</td>
            <td>Média</td>
        </tr>
        <tr>
            <td><?php echo $min_humi; ?> %</td>
            <td><?php echo $max_humi; ?> %</td>
```

```
                <td><?php echo round($avg_humi, 2); ?> %</td>
            </tr>
        </table>
    </div>
</section>
<?php
// Exibe um título indicando a quantidade de leituras a serem mostradas
echo '<h2> Ver as Últimas ' . $readings_count . '  Leituras </h2>';
// Inicia a criação de uma tabela HTML para exibir as leituras
echo '<table cellspacing="5" cellpadding="5" id="tableReadings">';
echo '<tr>
        <th>ID</th>
        <th>Temperatura</th>
        <th>humidade</th>
        <th>aquecimento</th>
        <th>sensorluz</th>
        <th>luzes</th>
        <th>telhado</th>
        <th>Data/horas</th>
    </tr>';
// Prepara a query SQL para selecionar as leituras mais recentes
$sql = 'SELECT TOP '.$readings_count.' * FROM info ORDER BY reading_time DESC';
// Estabelece uma conexão com o banco de dados
$conn = connection_db();
```

```
// Executa a query SQL e armazena o resultado em uma variável
$resultado = sqlsrv_query($conn, $sql);
// Verifica se a query foi executada com sucesso
if ($resultado) {
    // Percorre cada linha do resultado e exibe as informações na tabela HTML
    while ($row = sqlsrv_fetch_array($resultado, SQLSRV_FETCH_ASSOC)) {
        // Armazena os valores de cada coluna em variáveis separadas para facilitar o acesso
        $row_id = $row["id"];
        $row_temp = $row["temp"];
        $row_humidade = $row["humidade"];
        $row_reading_time = $row["reading_time"]->format("Y-m-d H:i:s");
        // Converte os valores de 0/1 em "on" ou "off" para exibir na tabela
        $aquecimento = intval($row['aquecimento']) == 0 ? "off" : "on";
        $sensorluz = intval($row['sensorLuz']) == 0 ? "off" : "on";
        $luzes = intval($row['luzes']) == 0 ? "off" : "on";
        $telhado = intval($row['telhado']) == 0 ? "off" : "on";
        // Exibe os valores na tabela HTML
        echo '<tr>
                <td>' . $row_id . '</td>
                <td>' . $row_temp . '&deg;C</td>
                <td>' . $row_humidade . '%</td>
                <td>' . $aquecimento . '</td>
                <td>' . $sensorluz . '</td>
                <td>' . $luzes . '</td>
```

```
            <td>' . $telhado. '</td>
            <td>' . $row_reading_time . '</td>
          </tr>';
    }
    // Fecha a tag da tabela HTML
    echo '</table>';
    // Liberta o resultado da query e fecha a conexão com a base de dados
    sqlsrv_free_stmt($resultado);
    sqlsrv_close($conn);
}
?>
<script>
// Definição da função a ser executada quando a página é carregada
window.onload = function(){
    // Esconde o elemento com o ID "ring"
    document.getElementById("ring").style.display = "none";
    // Exibe o elemento com o ID "page"
    document.body.style.display = "block";
}
// Leitura dos valores da temperatura e humidade obtidos via PHP
var temp = <?php echo $last_reading_temp; ?>;
var humidade = <?php echo $last_reading_humi; ?>;

// Chamada das funções para exibir os valores obtidos no ecrã
```

```
setTemperature(temp);
setHumidity(humidade);
// Definição da função para exibir o valor da temperatura
function setTemperature(curVal){
    // Definição dos valores mínimo e máximo para a escala da temperatura em Celsius
    var minTemp = -5.0;
    var maxTemp = 38.0;
    // Definição dos valores mínimo e máximo para a escala da temperatura em Fahrenheit
    // var minTemp = 23;
    // var maxTemp = 100;
    // Escalonamento do valor da temperatura obtido para a escala de exibição no ecrã
    var newVal = scaleValue(curVal, [minTemp, maxTemp], [0, 180]);
    // Atualização da exibição da temperatura no ecrã
    $('.gauge--1 .semi-circle--mask').attr({
        style: '-webkit-transform: rotate(' + newVal + 'deg);' +
        '-moz-transform: rotate(' + newVal + 'deg);' +
        'transform: rotate(' + newVal + 'deg);'
    });
    $("#temp").text(curVal + ' ºC');
}
// Definição da função para exibir o valor da humidade
function setHumidity(curVal){
    // Definição dos valores mínimo e máximo para a escala da humidade
    var minHumi = 0;
```

```
    var maxHumi = 100;
    // Escalonamento do valor da humidade obtido para a escala de exibição no ecrã
    var newVal = scaleValue(curVal, [minHumi, maxHumi], [0, 180]);
    // Atualização da exibição da humidade na tela
    $('.gauge--2 .semi-circle--mask').attr({
        style: '-webkit-transform: rotate(' + newVal + 'deg);' +
        '-moz-transform: rotate(' + newVal + 'deg);' +
        'transform: rotate(' + newVal + 'deg);'
    });
    $("#humi").text(curVal + ' %');
}
// Definição da função para escalonar o valor da temperatura ou humidade
function scaleValue(value, from, to) {
    // Cálculo do fator de escala para o valor a ser escalonado
    var scale = (to[1] - to[0]) / (from[1] - from[0]);
    // Limitação do valor a ser escalonado dentro dos limites de escala definidos
    var capped = Math.min(from[1], Math.max(from[0], value)) - from[0];
    // Escalonamento do valor propriamente dito
    return ~~(capped * scale + to[0]);
}
</script>
</body>
</html>
```