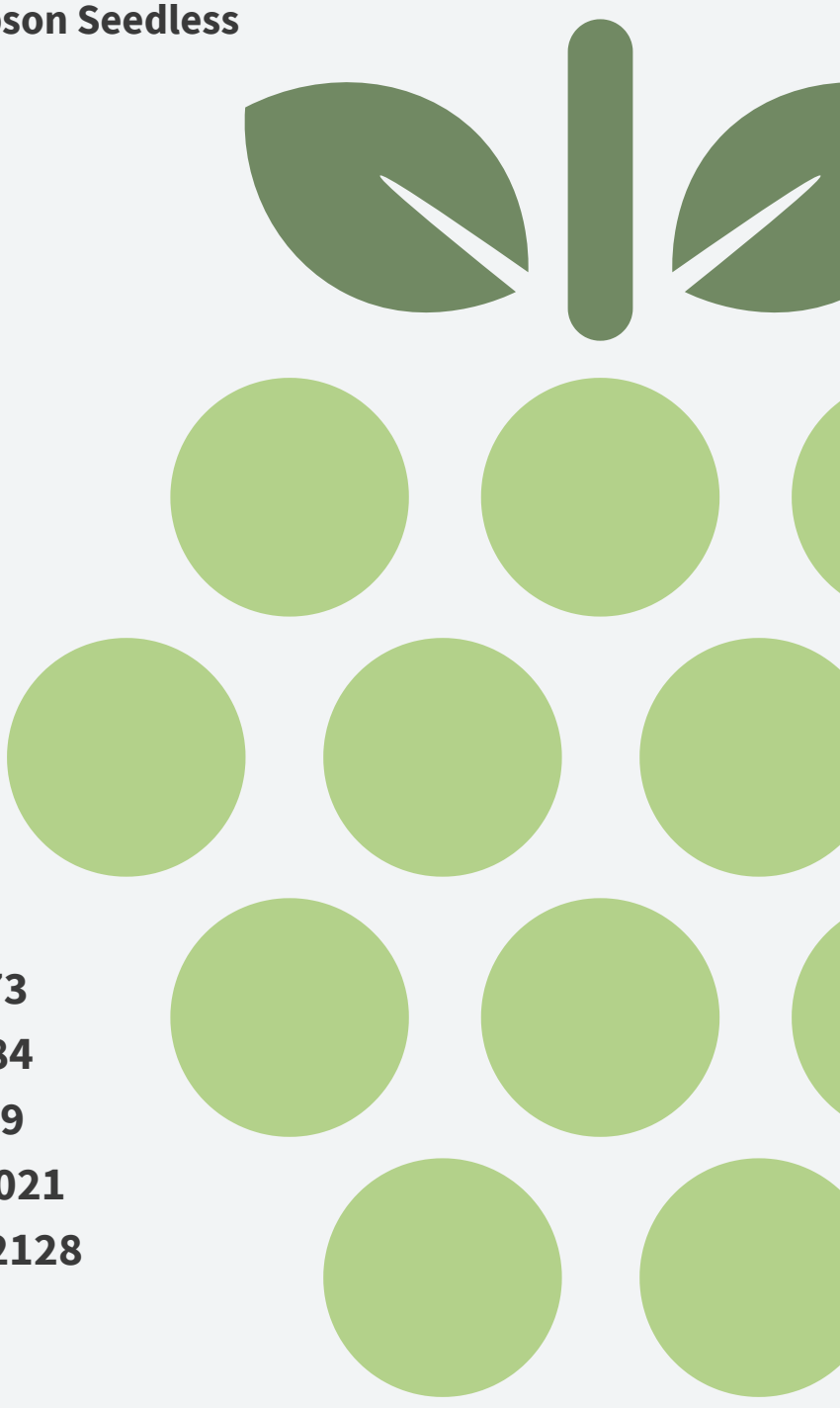
CTRL+V

Relatório Técnico - Arq. Comp.

Controle de temperatura e umidade na produção de uvas Thompson Seedless



1ADSA - 2025/2

Everton Barbosa | 01252061
Giovana Branquinho | 01252073
João Henrique Sapia | 01252084
Larissa Lie Okamoto | 01252039
Leandro Almeida Silva | 01252021
Sabrina da Silva Araujo | 01252128

SOBRE O PROJETO

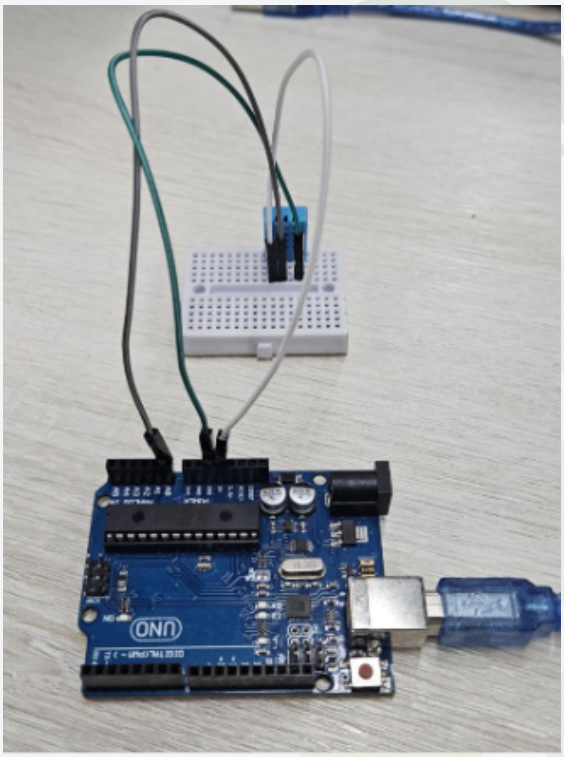
Tema do projeto:
Controle de Temperatura e Umidade das Uvas *Thompson Seedless* – *Indoor*

Sensor:
DHT11 (Temperatura e Umidade)

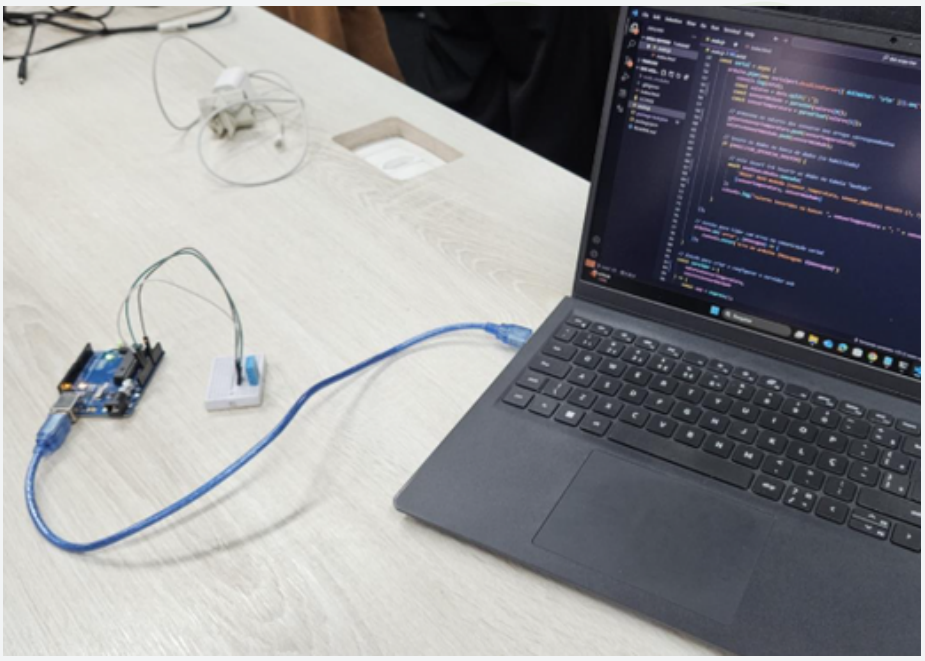
INTRODUÇÃO

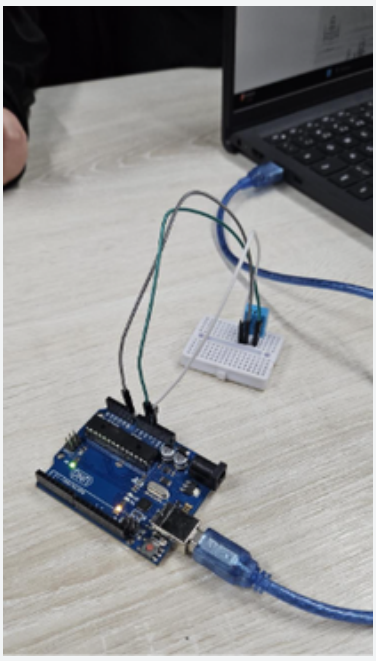
A Ctrl+V é uma empresa que visa o desenvolvimento de um sistema de monitoramento de temperatura e umidade nas produções da uva *Thompson Seedless* indoor em estufas. Coletando dados e gerando insights durante todo o processo de produção. A coleta de dados será direcionada a um dashboard com o objetivo de auxiliar o produtor na tomada de decisões relacionadas ao controle de temperatura e umidade da sua produção.

ARQUITETURA DE MONTAGEM DO SENSOR



Passo 1: Colocar o Sensor DHT11 na protoboard.
Passo 2: Ligar o jumper (da cor branca), na protoboard (no lado esquerdo) e no pino 5V no Arduino.
Passo 3: Ligar o jumper (da cor cinza), na protoboard (no lado esquerdo central) e na entrada A0 no Arduino.
Passo 4: Ligar o jumper (da cor verde), na protoboard (no lado direito) e no pino GND no Arduino.
Passo 5: Conectar o USB no Arduino e no Notebook.





ARQUITETURA DO SISTEMA

Por meio da estrutura do sensor mencionada na página anterior, é realizada a captação dos dados referentes à temperatura e umidade no ambiente, que são recebidos pela API, podendo ser armazenados em um banco de dados por meio de uma conexão feita no código da própria API, como pode ser observado abaixo:

```
API executada com sucesso na porta 3300
A leitura do arduino foi iniciada na porta COM3 utilizando Baud Rate de 9600
49.00;21.10;
valores inseridos no banco:  21.1, 49
49.00;21.10;
valores inseridos no banco:  21.1, 49
49.00;21.10;
valores inseridos no banco:  21.1, 49
49.00;21.10;
valores inseridos no banco:  21.1, 49
58.00;21.10;
valores inseridos no banco:  21.1, 58
69.00;21.20;
valores inseridos no banco:  21.2, 69
75.00;21.30;
valores inseridos no banco:  21.3, 75
79.00;21.50;
valores inseridos no banco:  21.5, 79
81.00;21.60;
valores inseridos no banco:  21.6, 81
83.00;21.70;
valores inseridos no banco:  21.7, 83
84.00;21.90;
valores inseridos no banco:  21.9, 84
84.00;22.00;
valores inseridos no banco:  22, 84
```

```sql
CREATE DATABASE testeapi;
USE testeapi;

CREATE TABLE medida (
idSensor INT PRIMARY KEY AUTO_INCREMENT,
sensor_Temperatura FLOAT,
sensor_Umidade INT
);

SELECT * from medida;
```

```
Result Grid | Filter Rows: | Edit:
```

idSensor	sensor_Temperatura	sensor_Umidade
1	23.2	64
2	23.2	65
3	23.2	65
4	23.2	65
5	23.2	65
6	23.1	65
7	23.1	65
8	23.1	65

```
// conexão com o banco de dados MySQL
let poolBancoDados = mysql.createPool(
    {
        host: 'localhost',
        user: 'root',
        password: '65279818',
        database: 'testeapi',
        port: 3306
    }
).promise();
```

CÓDIGO DO PROJETO

Uma parte do código do Arduino foi reestruturado, com o intuito de adequá-lo ao padrão de formatação definido na API, que segue um padrão de: **valor; valor**.

```
Serial.print("Umidade: ");
Serial.print(umidade);
Serial.print(" % ");
Serial.print("Temperatura: ");
Serial.print(temperatura);
Serial.println(" ºC");
```

ANTES

```
#include "DHT.h"

#define TIPO_SENSOR DHT11
const int  PINO_SENSOR_DHT11 = A0;

DHT sensorDHT(PINO_SENSOR_DHT11, TIPO_SENSOR);

void setup() {
  Serial.begin(9600);
  sensorDHT.begin();
}

void loop() {
  float umidade = sensorDHT.readHumidity();
  float temperatura = sensorDHT.readTemperature();

  if (isnan (temperatura) || isnan (umidade)){
    Serial.println ("Erro ao ler os dados do sensor");

  }else{
    Serial.print(umidade);
    Serial.print(";");
    Serial.print(temperatura);
    Serial.println(";");

  }

  delay(2000);
}
```

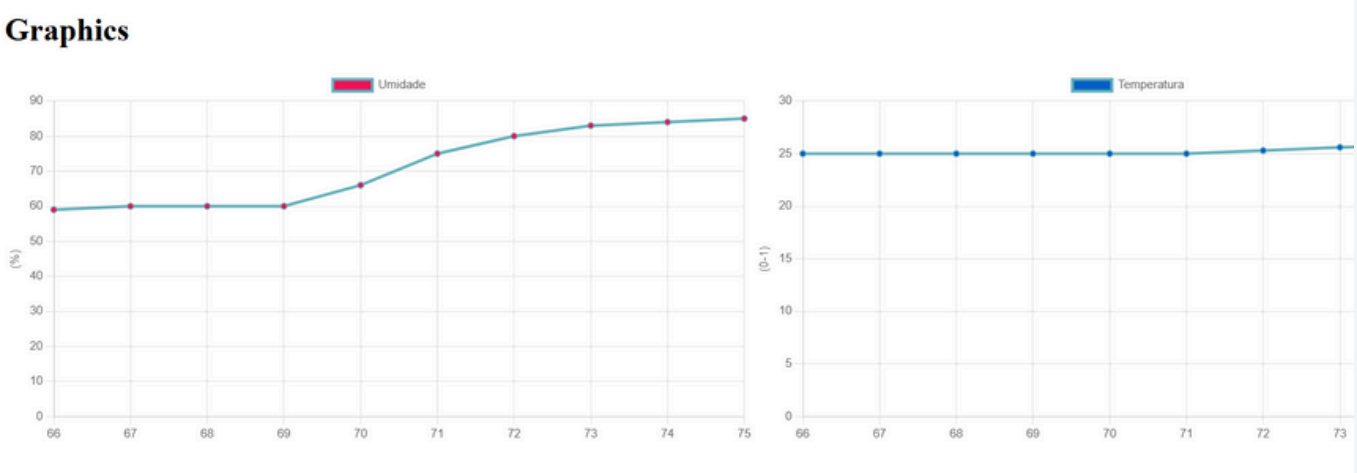
DEPOIS

```
// processa os dados recebidos do Arduino
arduino.pipe(new serialport.ReadlineParser({ delimiter: '\r\n' })).on('data', async (data) =>
    console.log(data);
    const valores = data.split(';');
    const sensorUmidade = parseInt(valores[0]);
    const sensorTemperatura = parseFloat(valores[1]);

    // armazena os valores dos sensores nos arrays correspondentes
```

O código da API foi reestruturado para ficar mais coeso com nossa regra de negócio. Todas as partes do código em que antes encontrava-se o trecho de texto "**sensorDigital**", agora encontra-se "**sensorUmidade**", o mesmo vale para os trechos de "**sensorAnalogico**" que viraram "**sensorTemperatura**", como pode ser visto na imagem acima.

RESULTADOS INICIAIS



Por meio da captação recebida, a API utiliza-se dos históricos de dados armazenados para a criação dos gráficos visualizados acima, que são atualizados de forma dinâmica. O histórico desses valores pode ser encontrado de forma definitiva no banco de dados e também pode ser observado nos seus respectivos Endpoints, bem como no CMD por meio do Node.js (como já visto mais cedo), mas somente enquanto a API estiver em execução.