



INSTITUTO FEDERAL

Brasília
Campus Brasília

TECNOLOGIA EM SISTEMAS PARA INTERNET

Claudio Lucas de Oliveira Franco

Jean Pereira Ribeiro

Marcos Vinicius Lira de Oliveira

Rafael Souza Nunes

**RELATÓRIO DE PRÁTICA INTEGRADA
DE
CIÊNCIA DE DADOS E INTERNET DAS COISAS**

Brasília - DF

23/12/2022

Sumário

| | |
|---------------------------------|-----------|
| 1. Objetivos | 3 |
| 2. Descrição do problema | 4 |
| 3. Desenvolvimento | 5 |
| 3.1 Código implementado | 9 |
| 4. Considerações finais | 13 |
| Referências | 14 |

1. Objetivos

O objetivo deste relatório é descrever o processo de desenvolvimento da Prática Integrada de Introdução a Ciência de Dados e Internet das Coisas. O projeto escolhido pelo grupo foi o da “Caixa para controle de umidade de filamento para impressora 3D”.

Para a sprint 1, os objetivos foram: montar o hardware e configurá-lo para funcionar adequadamente. Após a finalização dessa sprint, iniciou-se a sprint 2, em que os objetivos eram: definir o contexto de aplicação, coletar e preparar os dados.

2. Descrição do problema

Em relação ao projeto escolhido, o problema envolve a umidade no filamento que resulta em problemas de impressão e perdas do material. Para resolver esse problema, é necessário evitar ou remover a umidade no material para que se tenha uma impressão adequada (TECNOCUBO, 2022).

Nesse sentido, o contexto de aplicação desse projeto é orientar o leitor a realizar boas praticas visando eludir problemas ao iniciar uma impressão.

Os materiais geralmente utilizados para impressão 3D absorvem de forma natural a umidade. Os mais utilizados são o filamento PLA (ácido polilático), filamento Nylon, filamento ABS (acrilonitrila butadieno estireno) entre outros. O que pode acarretar em alguns problemas como uma peça quebradiça, camadas não uniformes, rompimento do filamento ou entupimento do bico.

Figura 1 – A esquerda um vaso impresso em PLA com umidade e à direita com o mesmo material em perfeito estado.



Fonte: 3DLab, 2016.

Temos algumas soluções disponíveis para resolver este problema, entretanto a mais viável é o armazenamento em ambiente adequado, inserindo ao nosso contexto vamos utilizar uma caixa de plástico com desumidificador para armazenar de forma correta o filamento visando garantir uma vida útil maior aos materiais e consequentemente constituindo impressões com melhores resultados.

3. Desenvolvimento

Para a realização do projeto, seguimos o passo a passo mostrado no artigo “Sensores DHT11 e DHT22: Guia Básico dos Sensores de Umidade e Temperatura” (MURTA, 2019).

Os materiais utilizados foram:

- 1 Arduino Uno R3 + Cabo Usb para Arduino
- 1 Protoboard 400 Pontos
- 3 Jumpers – Macho/Macho de 20cm
- 1 Sensor de Umidade e Temperatura DHT11

Os materiais foram adquiridos pela internet. Após a entrega, montamos o hardware de acordo com o Diagrama Fritzing a seguir:

Figura 2 – Diagrama Fritzing demonstrando o esquema de montagem do sensor DHT11 com o Arduino.

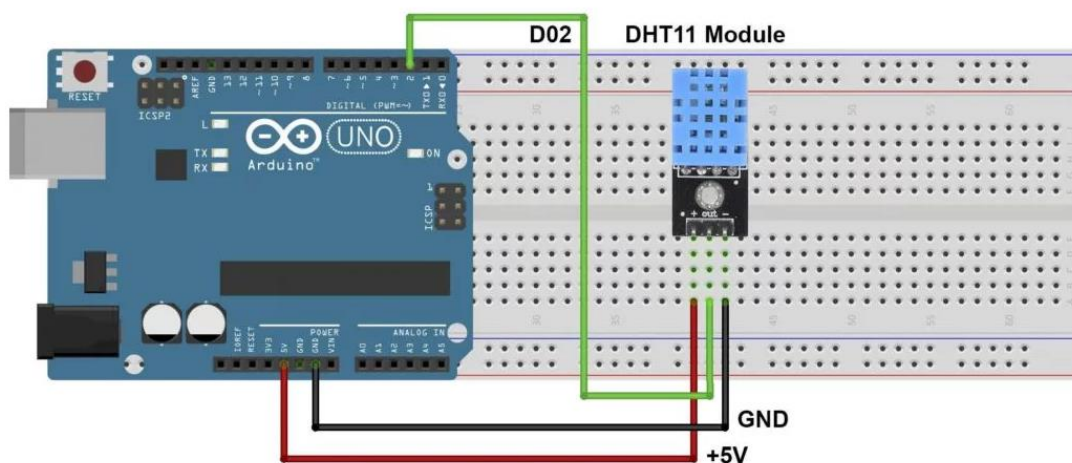
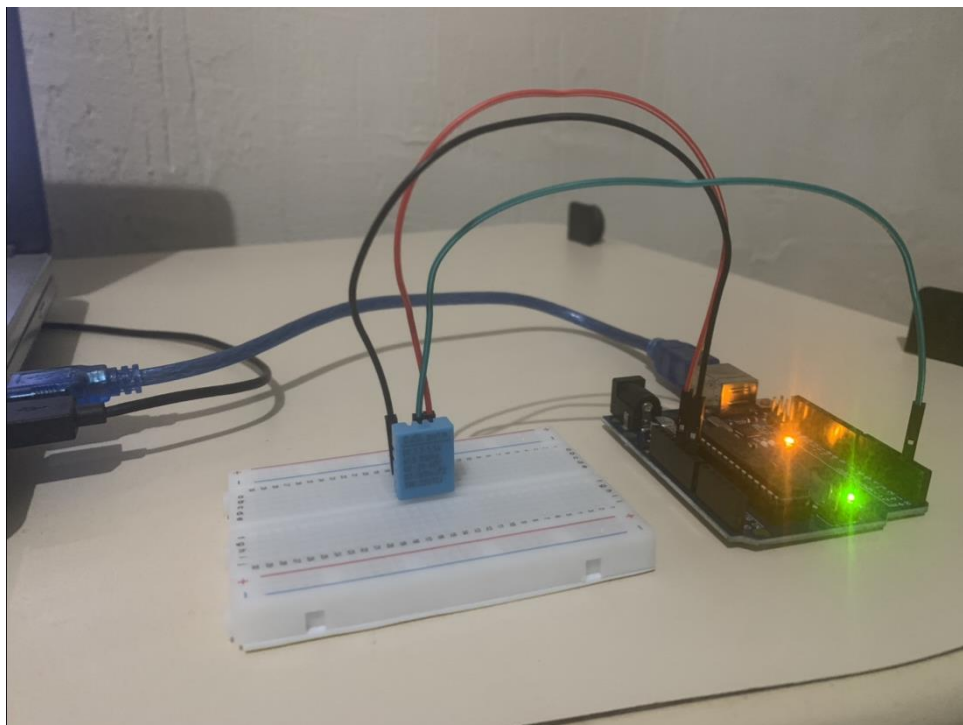


Diagrama Fritzing – Arduino e sensor DHT11

Fonte: Blog Eletrogate, 2009.

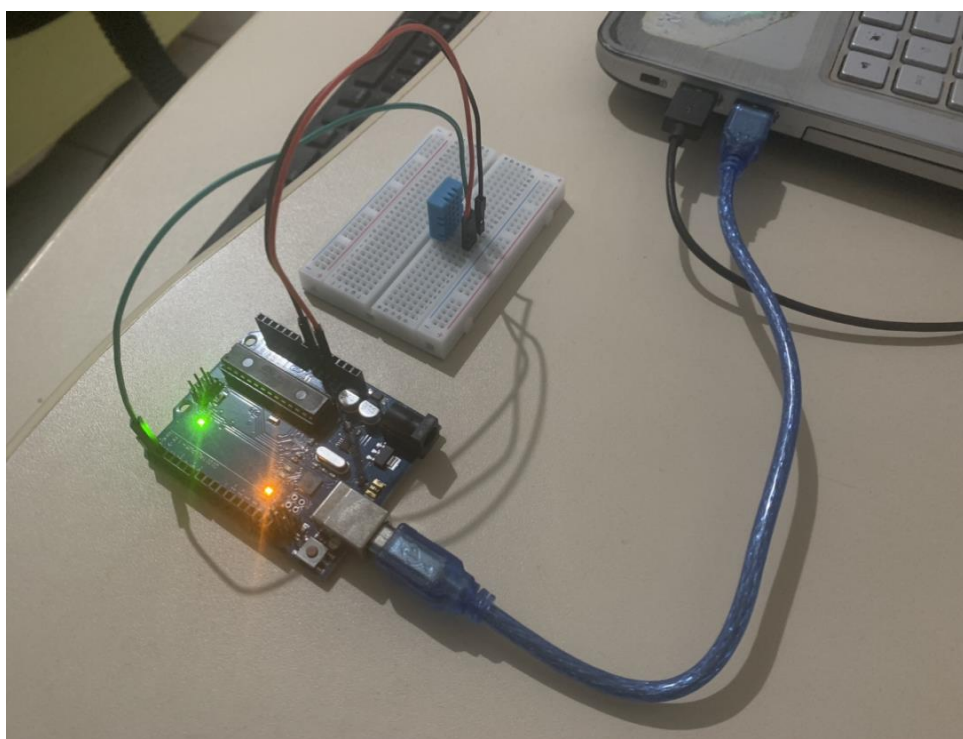
As imagens abaixo mostram o hardware montado:

Figura 3: Hardware montado (visto de frente)



Fonte: Elaborado pelos autores

Figura 4 – Hardware montado (visto por cima)



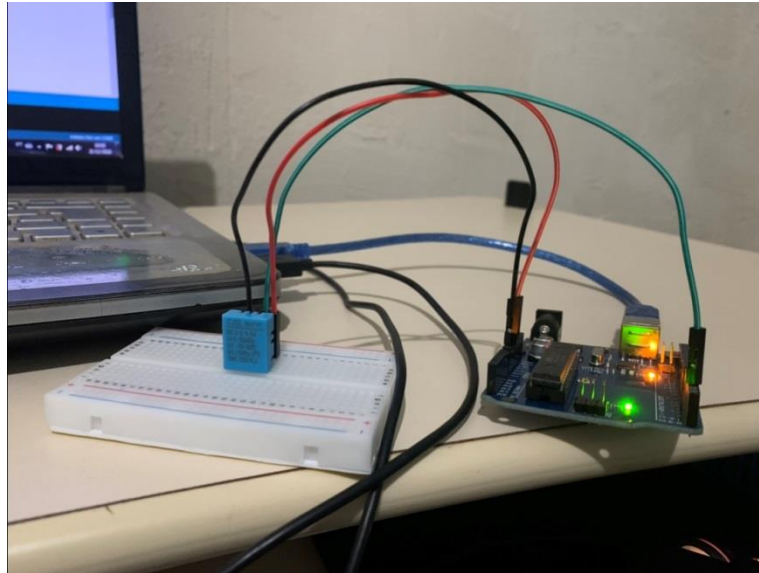
Fonte: Elaborado pelos autores

Para a etapa de coleta e preparação dos dados, o processo desenvolvido foi o seguinte:

Definimos o contexto e os dados que seriam coletados. Ao entender o problema que teríamos que resolver (evitar a umidade no filamento), decidimos coletar os dados de umidade e temperatura em 3 situações diferentes:

1 - Na coleta 1, os dados coletados vieram de uma sala normal, simulando a situação do filamento ser deixado desprotegido em um cômodo.

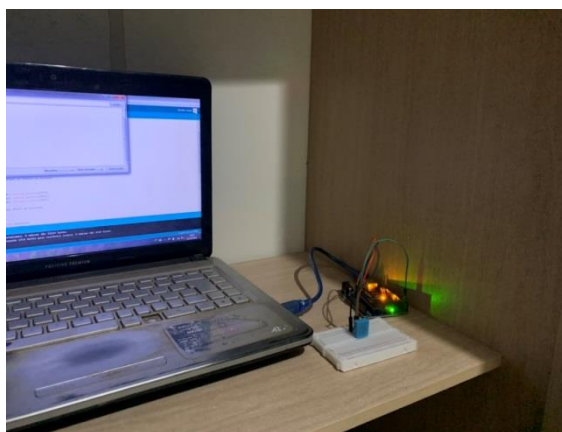
Figura 5 – Coleta 1



Fonte: Elaborado pelos autores.

2 - Na coleta 2, os dados coletados vieram de dentro de um armário fechado, simulando a situação do filamento ser guardado em um armário.

Figura 6 – Coleta 2 (imagem 1)



Fonte: elaborado pelos autores

Figura 7 – Coleta 2 (imagem 2)



Fonte: elaborado pelos autores

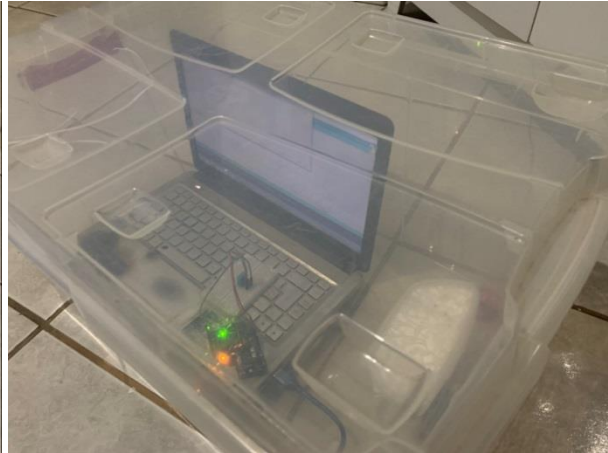
3 - Na coleta 3, os dados coletados vieram de dentro de uma caixa de plástico fechada contendo um desumidificador, simulando a situação do filamento ser guardado em um ambiente projetado para a diminuição da umidade.

Figura 8 – Coleta 3 (imagem 1)



Fonte: Elaborado pelos autores

Figura 9 – Coleta 3 (imagem 2)



Fonte: Elaborado pelos autores

Uma observação é que na coleta 3, a caixa precisava ser fechada para coletar os dados, então precisamos colocar o sensor junto com a fonte de energia, no caso um notebook. O calor emitido pelo notebook pode ter contribuído para o aumento da temperatura e consequente diminuição da umidade. Para coletas futuras, tentaremos deixar apenas o sensor dentro da caixa e, se for necessário, encontrar outra forma de aumentar a temperatura.

Com essas três coletas, o nosso objetivo era coletar os dados e verificar se haveria diferença na temperatura e umidade em cada ambiente. Para isso, decidimos coletar a temperatura e umidade de cada ambiente a cada 30 segundos durante 30 minutos.

Os resultados foram os seguintes:

- 1 – Na coleta 1, a temperatura variou entre 23,3 °C e 23,5 °C e a umidade variou entre 81% e 82%. Ou seja, se manteve estável durante os 30 minutos.
- 2 – Na coleta 2, a temperatura variou entre 23,3 °C e 23,5 °C e a umidade variou entre 80% e 81%. Ou seja, se manteve estável durante os 30 minutos.
- 3 – Na coleta 3, a medição de temperatura iniciou-se em 23,8 °C e terminou em 26,5 °C. A umidade inicial foi 79% e terminou em 67%. Ou seja, houve variação nos dados.

Os dados completos de cada coleta em formato CSV estão disponíveis na pasta “sprint 2” no GitHub, que pode ser acessado através desse link: <https://github.com/infocbra/pratica-integrada-cd-e-ic-2022-2-g2-cmrj>

Ao analisar os resultados, como o nosso objetivo era criar uma caixa que diminuísse o máximo possível a umidade, concluímos que a caixa projetada cumpriu o seu objetivo, ao reduzir a umidade do ar de 79% para 67% em 30 minutos, com tendência a continuar caindo caso a medição ocorresse por um período maior.

3.1 Código implementado

Para a sprint 1, o código implementado para o recebimento dos dados do sensor contendo os comentários explicando o funcionamento de cada parte foi o seguinte:

Figura 10 – Código (parte 1)



```
sketch_dec08a | Arduino 1.8.19
Arquivo Editar Sketch Ferramentas Ajuda

sketch_dec08a $
#include <Adafruit_Sensor.h>           // Biblioteca DHT Sensor Adafruit
#include <DHT.h>
#include <DHT_U.h>
#define DHTTYPE DHT11                  // Sensor DHT11
#define DHTPIN 2                       // Pino do Arduino conectado no Sensor(Data)
DHT_Unified dht(DHTPIN, DHTTYPE);      // configurando o Sensor DHT - pino e tipo
uint32_t delayMS;                     // variável para atraso no tempo
void setup()
{
  Serial.begin(9600);                  // monitor serial 9600 bps
  dht.begin();                        // inicializa a função
  Serial.println("Usando o Sensor DHT");
  sensor_t sensor;
```

Fonte: Elaborado pelos autores

Figura 11 – Código (parte 2)

```

dht.begin(); // inicializa a função
Serial.println("Usando o Sensor DHT");
sensor_t sensor;
dht.temperature().getSensor(&sensor); // imprime os detalhes do Sensor de Temperatura
Serial.println("-----");
Serial.println("Temperatura");
Serial.print ("Sensor: "); Serial.println(sensor.name);
Serial.print ("Valor max: "); Serial.print(sensor.max_value); Serial.println(" *C");
Serial.print ("Valor min: "); Serial.print(sensor.min_value); Serial.println(" *C");
Serial.print ("Resolucao: "); Serial.print(sensor.resolution); Serial.println(" *C");
Serial.println("-----");
dht.humidity().getSensor(&sensor); // imprime os detalhes do Sensor de Umidade
Serial.println("-----");
Serial.println("Umidade");
Serial.print ("Sensor: "); Serial.println(sensor.name);
Serial.print ("Valor max: "); Serial.print(sensor.max_value); Serial.println("%");
Serial.print ("Valor min: "); Serial.print(sensor.min_value); Serial.println("%");
Serial.print ("Resolucao: "); Serial.print(sensor.resolution); Serial.println("%");
Serial.println("-----");

```

Fonte: Elaborado pelos autores

Figura 12 – Código (parte 3)

```

delayMS = sensor.min_delay / 1000; // define o atraso entre as leituras
}
void loop()
{
    delay(delayMS); // atraso entre as medições
    sensors_event_t event; // inicializa o evento da Temperatura
    dht.temperature().getEvent(&event); // faz a leitura da Temperatura
    if (isnan(event.temperature)) // se algum erro na leitura
    {
        Serial.println("Erro na leitura da Temperatura!");
    }
    else // senão
    {
        Serial.print("Temperatura: "); // imprime a Temperatura
        Serial.print(event.temperature);
        Serial.println(" *C");
    }
    dht.humidity().getEvent(&event); // faz a leitura de umidade
    if (isnan(event.relative_humidity)) // se algum erro na leitura
    {

```

Fonte: Elaborado pelos autores

Figura 13 – Código (parte 4)

```

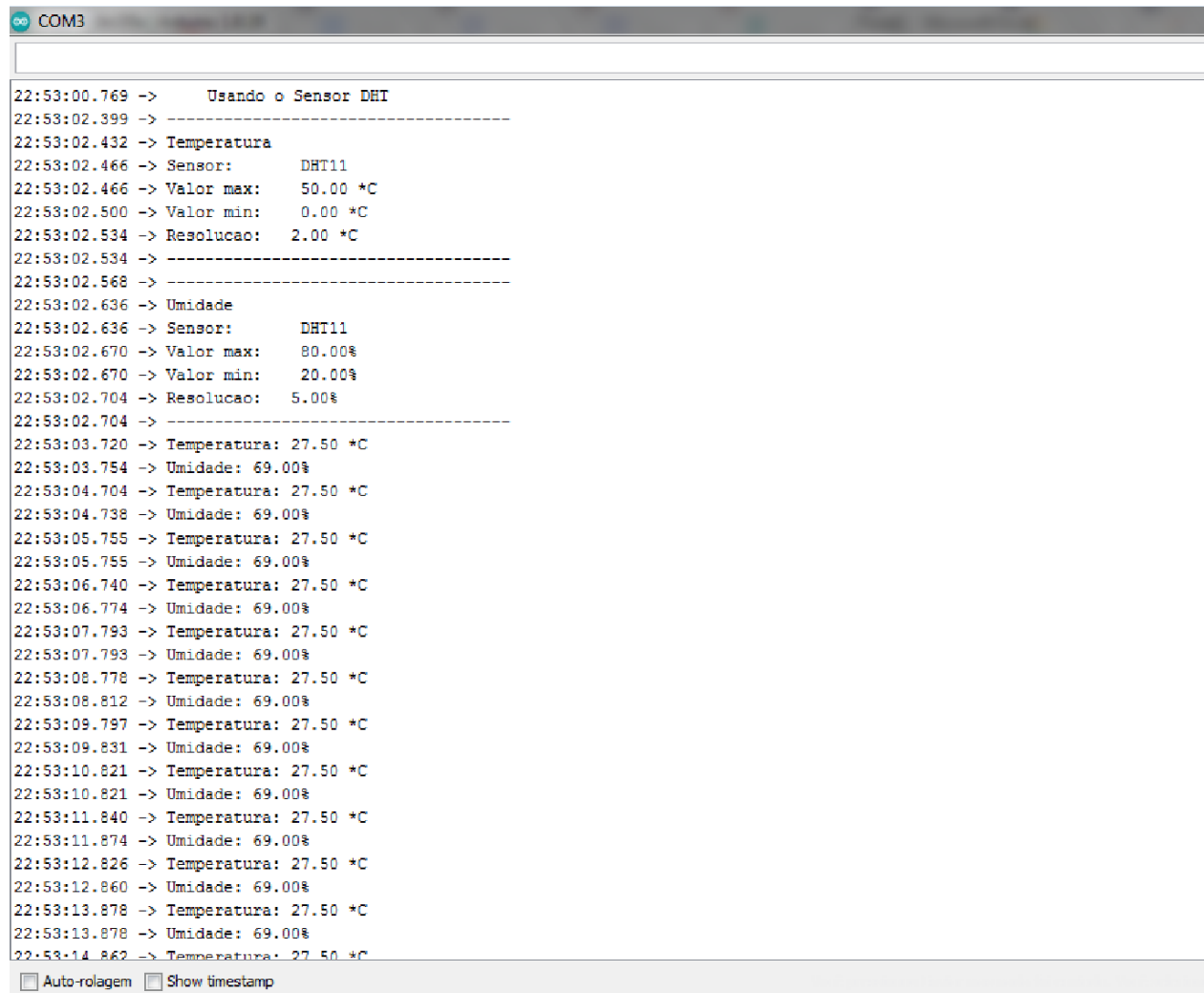
        Serial.println("Erro na leitura da Umidade!");
    }
    else // senão
    {
        Serial.print("Umidade: "); // imprime a Umidade
        Serial.print(event.relative_humidity);
        Serial.println("%");
    }
}

```

Fonte: Elaborado pelos autores

A imagem a seguir mostra a saída desse código e os dados que foram enviados pelo sensor. Na imagem é possível ver informações sobre o sensor, a hora da coleta e os dados de temperatura e a umidade.

Figura 14 – Saída do código



```
COM3
22:53:00.769 -> Usando o Sensor DHT
22:53:02.399 -> -----
22:53:02.432 -> Temperatura
22:53:02.466 -> Sensor:      DHT11
22:53:02.466 -> Valor max:   50.00 *C
22:53:02.500 -> Valor min:   0.00 *C
22:53:02.534 -> Resolucao:   2.00 *C
22:53:02.534 -> -----
22:53:02.568 -> -----
22:53:02.636 -> Umidade
22:53:02.636 -> Sensor:      DHT11
22:53:02.670 -> Valor max:   80.00%
22:53:02.670 -> Valor min:   20.00%
22:53:02.704 -> Resolucao:   5.00%
22:53:02.704 -> -----
22:53:03.720 -> Temperatura: 27.50 *C
22:53:03.754 -> Umidade: 69.00%
22:53:04.704 -> Temperatura: 27.50 *C
22:53:04.738 -> Umidade: 69.00%
22:53:05.755 -> Temperatura: 27.50 *C
22:53:05.755 -> Umidade: 69.00%
22:53:06.740 -> Temperatura: 27.50 *C
22:53:06.774 -> Umidade: 69.00%
22:53:07.793 -> Temperatura: 27.50 *C
22:53:07.793 -> Umidade: 69.00%
22:53:08.778 -> Temperatura: 27.50 *C
22:53:08.812 -> Umidade: 69.00%
22:53:09.797 -> Temperatura: 27.50 *C
22:53:09.831 -> Umidade: 69.00%
22:53:10.821 -> Temperatura: 27.50 *C
22:53:10.821 -> Umidade: 69.00%
22:53:11.840 -> Temperatura: 27.50 *C
22:53:11.874 -> Umidade: 69.00%
22:53:12.826 -> Temperatura: 27.50 *C
22:53:12.860 -> Umidade: 69.00%
22:53:13.878 -> Temperatura: 27.50 *C
22:53:13.878 -> Umidade: 69.00%
22:53:14.862 -> Temperatura: 27.50 *C
☐ Auto-rolagem ☐ Show timestamp
```

Fonte: Elaborado pelos autores.

Para a sprint 2, o código usado foi praticamente o mesmo, contendo apenas uma modificação no tempo de coleta dos dados. Antes, os dados do sensor eram lidos a cada 1 segundo. Com essa modificação, os dados são lidos a cada 30 segundos.

Figura 15 – Modificação no atraso entre as leituras

```

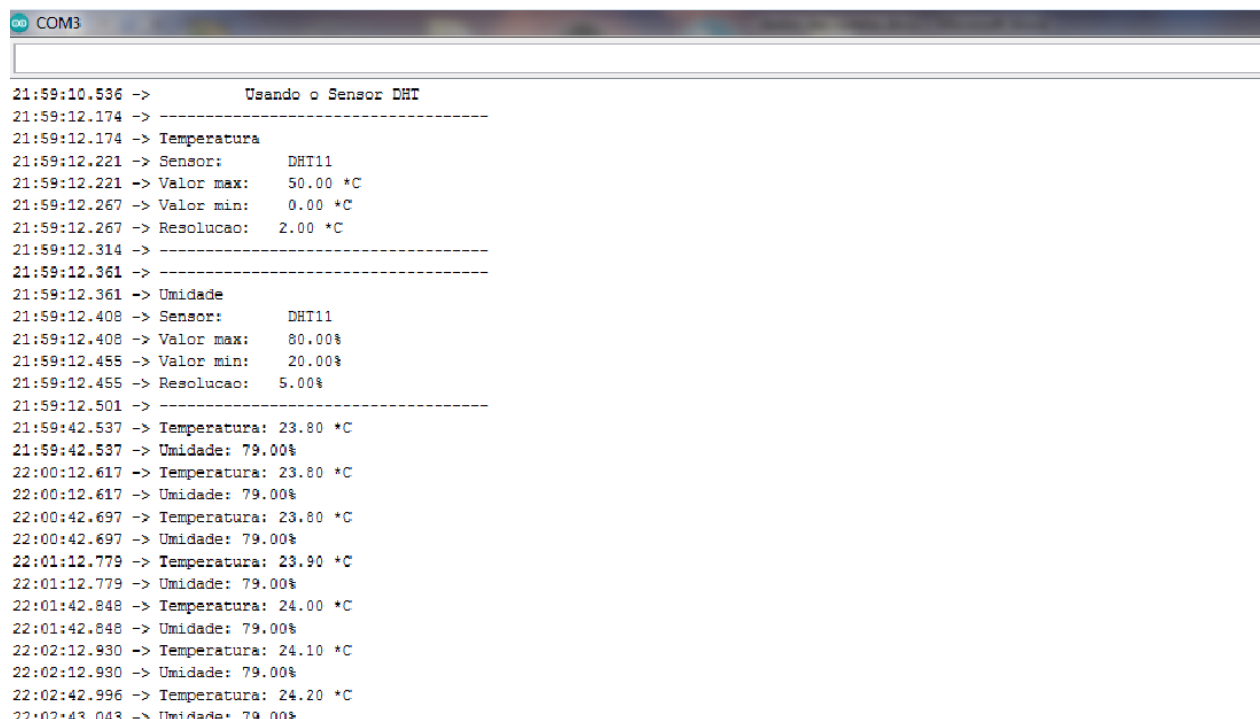
delayMS = sensor.min_delay / 33.3;           // define o atraso entre as leituras
}
void loop()
{
    delay(delayMS);                           // atraso entre as medições
    sensors_event_t event;                     // inicializa o evento da Temperatura
    dht.temperature().getEvent(&event);        // faz a leitura da Temperatura
    if (isnan(event.temperature))              // se algum erro na leitura
    {
        Serial.println("Erro na leitura da Temperatura!");
    }
    else                                       // senão
    {
        Serial.print("Temperatura: ");        // imprime a Temperatura
        Serial.print(event.temperature);
        Serial.println(" *C");
    }
    dht.humidity().getEvent(&event);           // faz a leitura de umidade
    if (isnan(event.relative_humidity))        // se algum erro na leitura
    {

```

Fonte: Elaborado pelos autores.

A figura a seguir mostra um exemplo de saída desse código utilizado na sprint 2. Essa foi a saída registrada na coleta 3 que foi explicada anteriormente. Na imagem é possível ver informações sobre o sensor, a hora da coleta e os dados de temperatura e a umidade.

Figura 16 – Saída do código da sprint 2



```

21:59:10.536 -> Usando o Sensor DHT
21:59:12.174 -> -----
21:59:12.174 -> Temperatura
21:59:12.221 -> Sensor:      DHT11
21:59:12.221 -> Valor max:    50.00 *C
21:59:12.267 -> Valor min:    0.00 *C
21:59:12.267 -> Resolucao:    2.00 *C
21:59:12.314 -> -----
21:59:12.361 -> -----
21:59:12.361 -> Umidade
21:59:12.408 -> Sensor:      DHT11
21:59:12.408 -> Valor max:    80.00%
21:59:12.455 -> Valor min:    20.00%
21:59:12.455 -> Resolucao:    5.00%
21:59:12.501 -> -----
21:59:42.537 -> Temperatura: 23.80 *C
21:59:42.537 -> Umidade: 79.00%
22:00:12.617 -> Temperatura: 23.80 *C
22:00:12.617 -> Umidade: 79.00%
22:00:42.697 -> Temperatura: 23.80 *C
22:00:42.697 -> Umidade: 79.00%
22:01:12.779 -> Temperatura: 23.90 *C
22:01:12.779 -> Umidade: 79.00%
22:01:42.848 -> Temperatura: 24.00 *C
22:01:42.848 -> Umidade: 79.00%
22:02:12.930 -> Temperatura: 24.10 *C
22:02:12.930 -> Umidade: 79.00%
22:02:42.996 -> Temperatura: 24.20 *C
22:02:43.043 -> Umidade: 79.00%

```

Fonte: elaborado pelos autores

O código utilizado pode ser encontrado no Github através do link:
<https://github.com/infocbra/pratica-integrada-cd-e-ic-2022-2-g2-cmrj>

4. Considerações finais

Para a realização da sprint 1, a principal dificuldade encontrada foi a de conseguir os materiais necessários para a realização do projeto. Fizemos um pedido pela internet e a entrega chegou poucos dias antes do fechamento da sprint, o que nos deu pouco tempo para a montagem do hardware e desenvolvimento do código. Apesar das dificuldades encontradas, ficamos satisfeitos por atingir o objetivo dessa sprint de montar o hardware e configurá-lo para funcionar adequadamente.

Na sprint 2, a principal dificuldade foi em montar a caixa utilizada para o controle de umidade e coletar os dados dentro dela. Devido ao fato da caixa precisar ser fechada para coletar os dados, a solução encontrada foi a de colocar o sensor e a fonte de energia (nesse caso um notebook) dentro da caixa e fechá-la. Nesse caso, o calor emitido pelo notebook contribuiu para o aumento da temperatura e diminuição da umidade. Para a próxima sprint, os objetivos para a melhoria do projeto é melhorar o desempenho da caixa para ela reduzir ainda mais a umidade e conseguir colocar apenas o sensor dentro da caixa e a fonte de energia fora, para não interferir na temperatura.

Referências

MURTA, José Gustavo Abreu. Sensores DHT11 e DHT22: Guia Básico dos Sensores de Umidade e Temperatura. **Blog Eletrogate**, 2019. Disponível em: <https://blog.eletrogate.com/sensores-dht11-dht22/>. Acesso em: 10 dez.2022.

Umidade no Filamento. **Tecnocubo**, c2022. Disponível em: <https://www.tecnocubo.com.br/pagina/umidade-no-filamento.html>. Acesso em: 10 dez.2022.

Como resolver umidade no filamento. **3DLab**, 2016. Disponível em: <https://3dlab.com.br/umidade-no-filamento/>. Acesso em: 22 dez.2022.