Arquitetura Computacional com API
Gabriella Lodi de Azevedo Antunes
Grupo 04 – 1CCOB

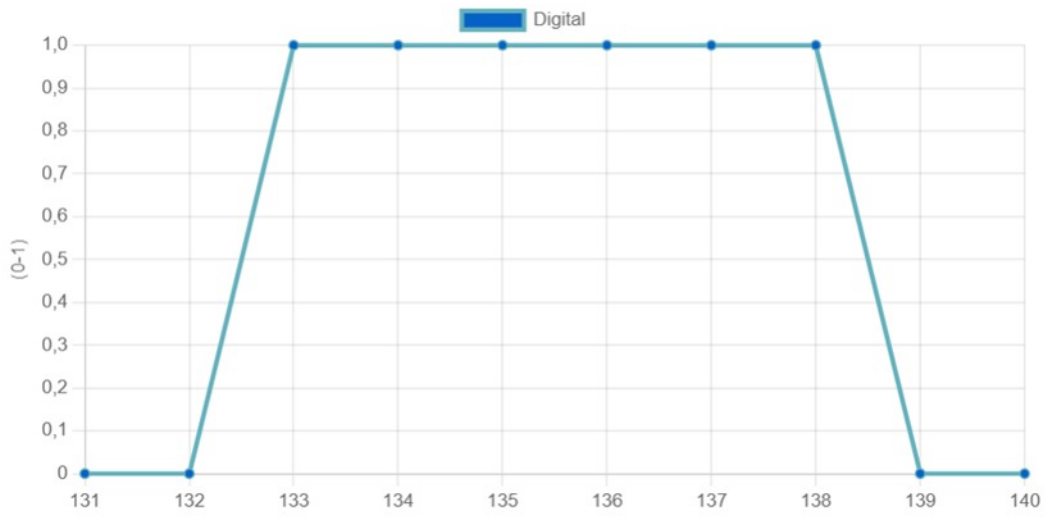
Questões avaliação prática:

1. Inicie o servidor NODE.

Para iniciar o servidor node, é necessário inserir o comando npm start no terminal.

```
Gabriellas-MacBook-Pro:~ gabriellalodiantunes$ npm start
```

2. Exiba os gráficos do projeto do grupo no index.html.



3. Qual comando é utilizado para instalar pacotes dos módulos NODE?

Para a instalação do pacote dos módulos NODE, se usa o comando npm install no terminal.

```
Gabriellas-MacBook-Pro:~ gabriellalodiantunes$ npm install
```

4. Demonstre no main.js como as variáveis do servidor de serviço foram definidas.

```
// importa os bibliotecas necessários
const serialport = require('serialport');
const express = require('express');
const mysql = require('mysql2');

// constantes para configurações
const SERIAL_BAUD_RATE = 9600;
const SERVIDOR_PORTA = 3300;

// habilita ou desabilita a inserção de dados no banco de dados
const HABILITAR_OPERACAO_INSERIR = true;

// função para comunicação serial
const serial = async (
    valoresSensorDigital
) => {

    // conexão com o banco de dados MySQL
    let poolBancoDados = mysql.createPool(
        {
            host: 'localhost',
            user: 'parkwise',
            password: 'Projeto123!',
            database: 'parkwisePI',
            port: 3307
        }
    ).promise();

    // lista as portas seriais disponíveis e procura pelo Arduino
    const portas = await serialport.SerialPort.list();
    const portaArduino = portas.find((porta) => porta.vendorId == 2341 && porta.productId == 43);
    if (!portaArduino) {
        throw new Error('O arduino não foi encontrado em nenhuma porta serial');
    }

    // configura a porta serial com o baud rate especificado
    const arduino = new serialport.SerialPort(
        {
            path: portaArduino.path,
            baudRate: SERIAL_BAUD_RATE
        }
    );
```

5. Qual é a distinção entre as portas 3300 e 3306?

A porta 3300 seria a porta do servidos aonde os dados serão inseridos, já a 3306 seria a porta local do servidor do Windows.

6. No index.html, como é feita a chamada à API externa chart.js? Onde é realizada essa chamada e onde um novo gráfico é criado?

A chamada é feita no <head>, mostrando o caminho que a API chart.js será usada. Ela então, possibilitará a criação de um novo gráfico a partir da tag <canvas>, tendo um id próprio e uma função específica para que o gráfico apareça.

```
<head>
    <title>Graphics</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
```

```
<h1>Graphics</h1>
<div style="display: flex;">
    <div style="width: 50%;">
        <!-- <canvas id="sensorAnalogico"></canvas> -->
    </div>
    <div style="width: 50%;">
        <canvas id="sensorDigital"></canvas>
    </div>
</div>
```

```
<script>
    var sensorDigital = new Chart(document.getElementById('sensorDigital').getContext('2d'), {
        type: 'line',
        data: {
            datasets: [{
                label: 'Digital',
                borderColor: '#63B1BC',
                backgroundColor: '#0762C8'
            }]
        },
        options: {
            scales: {
                x: {
                    beginAtZero: true
                },
                y: {
                    title: {
                        display: true,
                        text: '(0-1)'
                    },
                    beginAtZero: true
                }
            }
        }
    });
```

7. Onde é possível ajustar o tamanho e o tipo de gráfico gerado?

É possível mudar o tamanho do gráfico a partir do “width” do HTML, e o tipo do gráfico pode ser mudado no “type: ‘line’”, onde pode-se colocar, no lugar do line, bar, e entre outros tipos de gráficos.

```
<h1>Graphics</h1>
<div style="display: flex;">
    <div style="width: 50%;">
        <!-- <canvas id="sensorAnalogico"></canvas> -->
    </div>
    <div style="width: 50%;">
        <canvas id="sensorDigital"></canvas>
    </div>
</div>

var sensorDigital = new Chart(document.getElementById('sensorDigital').getContext
('2d'), {
    type: 'line',
    data: {
        datasets: [{
            label: 'Digital',
            borderColor: '#63B1BC',
            backgroundColor: '#0762C8'
        }]
```

8. O que é representado pelo método 'get' no código main.js?

O método ‘get’ no código main.js serve para “pegar” determinada informação. Então, no código abaixo, o get é usado conjuntamente ao app, representando o sensor digital e pegando as informações que esse sensor está absorvendo, para também colocar os gráficos dentro da API.

```
app.get('/sensores/digital', (_, response) => {
    return response.json(valoresSensorDigital);
});
```

9. Por que é gerado um arquivo JSON e para que ele é utilizado?

O arquivo JSON serve para o gerenciamento das informações sobre o projeto, dependências (que seria o modules_node) e scripts de execução. O JSON, então, é utilizado a partir do response.json, lendo uma stream de Response e retornando um resultado desejado.

O arquivo package.JSON é uma descrição da api, utilizado para que se tenha informações objetivas do nome da api, colaboradores e licenças.

O arquivo package-lock.json é o local onde está atribuído os módulos que a api consome, indicando todos os módulos que o node.js irá instalar para a utilização da api.

10. Considerando que no código.ino a saída é:

    a. **DHTH_temp; DHTH_umid; Luminosidade; LM35_temp; chave**, explique como essa estrutura de dados (na forma de lista) é adicionada como um vetor na APINode. Demonstre como o código captura essa lista e a divide ordenadamente dentro de um vetor.

    Essa estrutura de dados é adicionada como um vetor na APInode com a intenção de armazenar, em uma lista, os dados capturados pelos sensores. No caso do código acima, é possível notar três sensores: o de temperatura e umidade (DHTH), o de luminosidade (LDR) e o de temperatura (LM35).

    const valoresSensorAnalogico = [DHTH_temp; DHTH_umid; Luminosidade; LM35_temp; chave]

    Dentro dessa função, é atribuido uma constante que está armazenando os dados que o sensor captura:

    const valoresDHTH = [DHTH_temp; DHTH_umid]

    const valoresLuminosidade = [Luminosidade]

    const valoresLM35 = [LM35_temp]

    const valoresChave = [chave]

    Dando como exemplo no código da API, o vetor faz somente referência a um sensor, que seria o sensor TCHT5000, nomeado como "valoresSensorDigital" a partir de uma variável.

```
// função principal assíncrona para iniciar a comunicação serial e o servidor web
(async () => {
    // arrays para armazenar os valores dos sensores
    const valoresSensorDigital = [];
```

Os dados capturados são então processados nessa linha de código, sendo organizados em posição 0 (que seria o DHTH_temp), até a posição 3, que seria a o LM35_temp.

```
// 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 sensorDigital = parseInt(valores[0]);
```

const sensorDHTH_temp = parseFloat(valores[0]);

const sensorDHTH _umid= parseFloat(valores[1]);

const sensorLuminosidade = parseInt(valores[2]);

const sensorLM35 = parseInt(valores[3]);

const sensorChave = parseInt(valores[4]);