Lista de Exercícios 1 - Recapitulando o Backend (3º Ano)

---

1. Explique com suas palavras:

O que é um **endpoint** e qual sua importância em uma API?

---

2. Relacione com o CRUD:

Dado o seguinte endpoint:

```
DELETE /produtos/:id
```

📌 Qual operação do CRUD esse endpoint representa? Como seria uma requisição para excluir um produto com ID **10** usando JavaScript (`fetch`)?

---

3. Complete o código:

O código abaixo deveria cadastrar um novo usuário na API, mas está incompleto. Complete os trechos que faltam:

```
async function createUser(name) {
    const response = await fetch(, {
        method: "POST",
        headers: { "Content-Type": "?"},
        body: JSON.stringify({ name:})
    });

    if (response.ok) {
        console.log("Usuário criado com sucesso!");
    } else {
        console.error("Erro ao criar usuário.");
    }
}
```

---

4. Identifique e corrija o erro:

O código abaixo deveria buscar a lista de usuários da API, mas contém um erro. Identifique e corrija.

```
fetch("http://localhost:3000/users")
  .then(response ⇒ response.json())
  .then(data ⇒ console.log(data))
  .catch(error ⇒ console.log(Erro: ${error}));
```

---

5. Descreva o que acontece nesse trecho:

O que cada linha do código abaixo faz?

```
fetch("http://localhost:3000/users/5", { method: "DELETE" })
  .then(response ⇒ response.json())
  .then(data ⇒ console.log("Usuário removido:", data))
  .catch(error ⇒ console.error("Erro:", error));
```

---

6. Problema com atualização de usuário:

O código abaixo deveria atualizar o nome de um usuário, mas **sempre retorna erro 404 (Not Found)**. O que pode estar errado?

```
async function updateUser(id, newName) {
  const response = await fetch(`http://localhost:3000/users?id=${id}`, {
    method: "PUT",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify({ name: newName })
  });

  return await response.json();
}
```

---

7. Complete a requisição:

Preencha os espaços em branco para buscar um usuário pelo ID e exibir seu nome no console.

```
async function getUserById(id) {
  const response = await fetch();
  const user = await response.json();
  console.log("");
}
```

8. Código quebrado:

O seguinte código deveria listar usuários, mas **não exibe nada na tela**. Qual pode ser o problema?

```
async function fetchUsers() {
  const response = await fetch("http://localhost:3000/users");
  const users = await response.json();

  users.forEach(user => {
    document.getElementById("userTable").innerHTML += `
      <tr>
        <td>${user.id}</td>
        <td>${user.name}</td>
      </tr>
    `;
  });
}
fetchUsers();
```

9. Debugging de API:

Você fez uma requisição **POST** para cadastrar um usuário, mas nada acontece. Nem sucesso, nem erro aparecem no console.

O que você faria para depurar esse problema? Liste **pelo menos 3 passos**.

10. Coloque para funcionar!

Aqui está um código **incompleto e com erros**. Corrija os problemas para que ele funcione corretamente.

```
const express = require("express");
const app = express();
app.use(express.json());

let produtos = []; // Lista de produtos vazia

// Endpoint para listar produtos (GET)
app.get("/produtos", (req, res) => {
   res.json(produto);
});

// Endpoint para adicionar produto (POST)
app.post("/produto", (req, res) => {
   const novoProduto = { id: produtos.length + 1, nome: req.nome };
   produtos.push(novoProduto);
   res.status(201).json(novoProduto);
});

// Iniciar servidor
app.listen(3000);
```