Imersão Alura Back End / aula-3

Entendendo async e await em JavaScript

**async:** Essa palavra-chave transforma uma função regular em uma função assíncrona, permitindo que você use a palavra-chave await dentro dela para lidar com operações assíncronas de forma mais síncrona.

**await**: Essa palavra-chave faz com que o JavaScript espere que uma Promise seja resolvida antes de continuar a execução da função. É como fazer uma pausa na execução até que o resultado da operação assíncrona esteja pronto.

**Em resumo**:

**async**: Declara que uma função irá lidar com operações assíncronas.

**await:** Pausa a execução da função até que uma Promise seja resolvida.

Exemplo:

**JavaScript**

async function fetchData() {

const response = await fetch('https://api.example.com/data');

const data = await response.json();

return data; 1

1.

meen.academy

meen.academy

}

Use code with caution.

**Neste exemplo**:

**fetchData** é uma função assíncrona, declarada com **async**.

**await fetch('https://api.example.com/data')** espera que a requisição seja feita e retorna uma **Promise**.

**await response.json()** espera que a resposta seja convertida para JSON.

O resultado final, data, é retornado pela função.

**Por que usar async e await?**

**Código mais limpo:** Torna o código mais fácil de ler e entender, evitando callbacks aninhados.

**Gerenciamento de erros:** Permite usar try...catch para lidar com erros de forma mais clara.

**Melhora a legibilidade:** Faz o código parecer mais síncrono, mesmo quando lidando com operações assíncronas.

**Em resumo:**

**async** e **await** são ferramentas poderosas em JavaScript que facilitam o trabalho com operações assíncronas, tornando o código mais limpo e fácil de manter.

**Atividade**

**Claro! Que tal criar uma nova coleção no MongoDB chamada "users" (usuários) e implementar uma funcionalidade para adicionar novos usuários à sua aplicação? Aqui estão os passos que você pode seguir:**

**Criar a coleção "users": Acesse o MongoDB Atlas e crie uma nova coleção chamada "users" dentro do seu banco de dados "imersao-instabytes".**

**Definir o modelo do usuário: Decida quais campos você gostaria que cada usuário tivesse. Por exemplo, você pode incluir username, email e password.**

**Implementar a rota para adicionar usuários: Crie uma nova rota no seu arquivo postRoutes.js que permita adicionar um novo usuário. Use o método POST para receber os dados do usuário.**

**Criar a função no controller: No seu postsController.js, crie uma função que receba os dados do usuário e os insira na coleção "users" usando o método insertOne() do MongoDB.**

**Testar a funcionalidade: Utilize uma ferramenta como Postman ou Insomnia para enviar uma requisição POST para a nova rota que você criou, com os dados do usuário no corpo da requisição.**

**Esse exercício vai te ajudar a praticar a conexão com o banco de dados, a criação de rotas e a manipulação de dados! Se precisar de ajuda, não hesite em perguntar! 😊**

function buscarPostPorId(*id*){

    return posts.findIndex((*post*) => {

        return *post*.id == *Number*(*id*)

    })

}

app.get("/posts/:id", (*req*, *res*) => {

    const index = buscarPostPorId(*req*.params.id)

*res*.status(200).json(posts[index]);

});

import express from 'express';

import conectarAoBanco from './src/config/dbconfig.js';

const conexao = await conectarAoBanco(process.env.STRING\_CONEXAO)

const posts = [

    {

      id: 1, descricao: "Uma foto teste",

      imagem: "https://placecats.com/millie/300/150"

    },

    {

        id: 2, descricao: "Paisagem montanhosa",

      imagem: "https://source.unsplash.com/random/300x150/?mountain"

    },

    {

        id: 3, descricao: "Cachorro brincando",

      imagem: "https://source.unsplash.com/random/300x150/?dog,play"

    },

    {

        id: 4, descricao: "Comida deliciosa",

      imagem: "https://source.unsplash.com/random/300x150/?food"

    },

    {

        id: 5, descricao: "Cidade à noite",

      imagem: "https://source.unsplash.com/random/300x100/?city,night"

    },

    {

        id: 6, descricao: "Pessoas sorrindo",

      imagem: "https://source.unsplash.com/random/300x150/?people,smile"

    }

  ];

const app = express();

app.use(express.json());//Aqui nós estamos dizendo que tudo que foi reconhecido por Json. Estamos indicando que nossa aplicação usa essa funcionalidade de converter a estrutura em Json.

// Agora o express está pronto para receber o que vem texto do array posts e transformar em Json denovo para usar.

app.listen(3000, () => {

    console.log("Servidor executando...." );

});

async function getTodosPosts() {

  const db = conexao.db ("imersao-instabytes")

  const colecao = db.collection("posts")

  return colecao.find().toArray()

}

app.get("/posts", async (*req*, *res*) => {

    const posts = await getTodosPosts()

*res*.status(200).json(posts);

});