**Desafio Node - 1**

🚀 **Sobre o desafio**

Crie uma aplicação que fará o cadastro dos pedidos de uma hamburgueria, e você deve utilizar Node e Express.

**Rotas**

**POST /order**: A rota deve receber o pedido do cliente, o nome do cliente e o valor do pedido, essas informações devem ser passadas dentro do corpo(body) da requisição, e com essas informações você deve registrar o novo pedido dentro de um array no seguinte formato: { id: "ac3ebf68-e0ad-4c1d-9822-ff1b849589a8", order: "X- Salada, 2 batatas grandes, 1 coca-cola", clientName:"José", price: 44.50, status:"Em preparação" }. Não se esqueça que o ID deve ser gerado por você, dentro do código utilizando UUID V4, assim que o pedido é criado, você deve sempre colocar o status como "Em preparação".

**GET /order**: Rota que lista todos os pedidos já feitos.

**PUT /order/:id**: Essa rota deve alterar um pedido já feito. Pode alterar,um ou todos os dados do pedido.O id do pedido deve ser enviado nos parâmetros da rota.

**DELETE /order/:id**: Essa rota deve deletar um pedido já feito com o id enviado nos parâmetros da rota.

**GET /order/:id**: Essa rota recebe o id nos parâmetros e deve retornar um pedido específico.

**PATCH /order/:id**: Essa rota recebe o id nos parâmetros e assim que ela for chamada, deve alterar o status do pedido recebido pelo id para "Pronto".

Exemplo

Se eu chamar a rota POST /order repassando { order: "X- Salada, 2 batatas grandes, 1 coca-cola", clienteName:"José", price: 44.50 }, o array deve ficar assim:

[

{

id: "ac3ebf68-e0ad-4c1d-9822-ff1b849589a8",

order: "X- Salada, 2 batatas grandes, 1 coca-cola",

clienteName:"José",

price: 44.50,

status:"Em preparação"

}

];

Se eu chamar a rota PATCH /order/ac3ebf68-e0ad-4c1d-9822-ff1b849589a8, o array deve ficar assim:

[

{

id: "ac3ebf68-e0ad-4c1d-9822-ff1b849589a8",

order: "X- Salada, 2 batatas grandes, 1 coca-cola",

clienteName:"José",

price: 44.50,

status:"Pronto"

}

];

Middlewares

Crie um middleware que será utilizado em todas rotas que recebem o parâmetro ID, então ele deve verificar se o ID passado existe. Se não existir retorne um erro, caso contrário permita a requisição continuar normalmente;

Crie um middleware que é chamado em todas requisições que tenha um console.log que mostra o método da requisiçao(GET,POST,PUT,DELETE, etc) e também a url da requisição.

Exemplo

[GET] - /order

📅 Entrega

Após finalizar o desafio, suba o projeto para o seu github e compartilhe no Club, para todos verem que você conseguiu.

Feito com ♥ by Code Club - Rodolfo Mori

const { request } = require('express')

const express = require('express')

const uuid = require('uuid')

const port = 3000

const app = express()

app.use(express.json())

/\*

- Query params => meusite.com/user?name=rodolfo&age=28 // FILTROS

- Route params => /users/2 // BUSCAR, DELETAR OU ATUALIZAR ALGO ESPECÍFICO

- Resquest Body => { "name" : "Rodolfo", "age": "28" }

- GET => Buscar informações no back-end

- POST => Criar informação no back-end

- PUT / PACH => Alternar/Atualizar informação no back-end

- DELETE => Deletar informação no back-end

- Middleware => INTERCEPTADOR => Tem o poder de parar ou alterar dados da requisição

\*/

const users = []

const checkUserId = (request, response, next) => {

const { id } = request.params

const index = users.findIndex(user => user.id === id)

if (index < 0) {

return response.status(404).json({ Error: "User not found" })

}

request.userIndex = index

request.userId = id

next()

}

//app.use(checkUserId)

app.get('/users/', (request, response) => {

return response.json(users)

})

app.post('/users/', (request, response) => {

const { name, age } = request.body

const user = { id: uuid.v4(), name, age }

users.push(user)

return response.status(201).json(user)

})

app.put('/users/:id', checkUserId, (request, response) => {

const { name, age } = request.body

const index = request.userIndex

const id = request.userId

const updateUser = { id, name, age }

users[index] = updateUser

return response.json(updateUser)

})

app.delete('/users/:id', checkUserId, (request, response) => {

const index = request.userIndex

users.splice(index, 1) // indice "index", deleta apenas a posição um "index".

return response.status(204).json()

})

app.listen(port, () => {

console.log(`🚀 Server started on port ${port}`)

})