NODE

**Backend REST 1 - Preparação do ambiente/Criação do Projeto**

**Criação do projeto**

Ambiente de desenvolvimento:

* VSCode
* MySQL (ou XAMPP)
* MySQL Workbench
* Node.js

Criação do projeto:

Caso seja utilizado o GitHub:

* Criar o repositório no GitHub
* Executar os procedimentos indicados no GitHub para criação do repositório local
* Criar um arquivo .gitignore para projetos Node.js

Na pasta do projeto, abrir o terminal e executar:

**npm init -y**

Abrir o arquivo package.json:

**editar a descrição e o autor**

Instalar as primeiras dependências:

**npm install express mysql dotenv cors body-parser**

Instalar outra dependência, mas apenas para o ambiente dev:

**npm install nodemon --save-dev**

**Criar a pasta src na raiz do projeto**

**Na pasta raiz criar o arquivo variaveis.env:**

PORT=3000

**Na pasta src criar o arquivo server.js:**

Adicionar no package.json, na seção scripts:

**“start”: “nodemon ./src/server.js”,**

Obs: o nademon, sempre que houver uma modificação nos arquivos do projeto fará um “refresh” do servidor.

**Backend REST 2 - Criação do servidor e rotas**

**Na pasta src criar o arquivo routes.js:**

const express = require('express');

const router = express.Router();

module.exports = router;

**Alterar o arquivo server.js:**

require('dotenv').config({path: 'variaveis.env'});

const express = require('express');

const cors = require('cors');

const bodyParser = require('body-parser');

const routes = require('./routes');

const server = express();

server.use(cors());

server.listen(process.env.PORT, () => {

console.log(`Servidor rodando em: http://localhost:${process.env.PORT}`);

});

**Executar a aplicação:**

npm start

**Backend REST 3 - Banco de dados**

**Criação do banco e tabelas**

CREATE DATABASE db\_expo;

USE db\_expo;

CREATE TABLE usuario(

id int PRIMARY KEY auto\_increment,

email VARCHAR(50) NOT NULL UNIQUE,

nome VARCHAR(50) NOT NULL,

nascimento DATE NOT NULL

);

**Carga de teste nas tabelas**

USE db\_expo;

INSERT INTO usuario(email, nome, nascimento) [VALUES('usuario\_p@hotmail.com](mailto:VALUES('usuario_p@hotmail.com)', 'Primeiro Usuario', '2021-11-06');

INSERT INTO usuario(email, nome, nascimento) [VALUES('usuario\_d@hotmail.com](mailto:VALUES('usuario_d@hotmail.com)', 'Segundo Usuario', '1980-03-10');

INSERT INTO usuario(email, nome, nascimento) [VALUES('usuario\_t@hotmail.com](mailto:VALUES('usuario_t@hotmail.com)', 'Outro Usuario', '1971-10-14');

SELECT \* FROM usuario;

**Backend REST 4 - Implementação**

**No arquivo variaveis.env adicionar as configurações para acesso ao banco de dados:**

DB\_HOST=localhost

DB\_USER=root

DB\_PASS=

DB\_NAME=db\_expo

**Na pasta src, criar a pasta dao**

**Na pasta src, criar a pasta service**

No arquivo server.js, antes do listen, adicionar:

**server.use(‘api/‘, routes);**

**No arquivo routes.js, criar as primeiras rotas para demonstrar o funcionamento:**

router.get('/usuarios', async function(req, res){

res.end();

});

router.get('/usuarios/:id');

**Alterar a implementação acima para demonstrar o retorno de um json pelo endpoint usuarios:**

router.get('/usuarios', async function(req, res){

res.json([{

code: 0,

message: 'Endpoint válido'

}]);

});

**Na pasta src, criar o arquivo db.js:**

const mysql = require('mysql');

const connection = mysql.createConnection({

host: process.env.DB\_HOST,

user: process.env.DB\_USER,

password: process.env.DB\_PASS,

database: process.env.DB\_NAME

});

connection.connect((error) => {

if(error) throw error;

console.log(`Conectado ao banco de dados ${process.env.DB\_NAME}`);

});

module.exports = connection;

**Na pasta dao, criar o arquivo usuarioDao.js:**

const db = require('../db');

listarUsuarios: () => {

return new Promise((aceito, rejeitado) => {

db.query('SELECT \* FROM usuario', (error, results) => {

if(error){

rejeitado(error);

return;

}

aceito(results);

})

});

};

module.exports = listarUsuarios;

**Na pasta service, criar o arquivo usuarioService.js:**

const usuarioDao = require('../dao/usuarioDao');

listar: async(req, res) => {

let retorno = {error: '', result: []};

let usuarios = await usuarioDao.listarUsuarios();

for(let i in usuarios){

retorno.result.push({

id: usuarios[i].id,

email: usuarios[i].email,

nome: usuarios[i].nome,

nascimento: usuarios[i].nascimento

});

}

res.json(retorno);

}

**Alterar a rota de usuários em routes.js:**

router.get('/usuarios', usuarioService.listar);