**Relatorio como criar uma api para consumir os dados do mysql.**

**Gabriel Ferreira de Lucas – Dev**

[**https://github.com/gflucas/COMO-CRIAR-UMA-API-PARA-CONSUMIR-OS-DADOS-DO-MYSQL/tree/main**](https://github.com/gflucas/COMO-CRIAR-UMA-API-PARA-CONSUMIR-OS-DADOS-DO-MYSQL/tree/main)

Para que seja possível realizar a conexão do NodeJS com o MySQL, é necessário que o banco de dados esteja instalado e configurado corretamente no Docker. Certifique-se de que a versão do MySQL compatível com a aplicação esteja instalada para isso eu utilizei a última versão em todas as imagens e, em seguida, crie um usuário para o banco de dados. Lembre-se de criar um usuario com privilégios de administrador, como o usuário root, que permitirá a conexão com o banco de dados com a API NodeJS.

Além disso criei um banco de dados chamado: **nodejsql**

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

E criei a tabela pessoa, da qual a API utilizar.

CREATE TABLE pessoa (

  id INT,

  nome VARCHAR(50),

  telefone VARCHAR(20),

  email VARCHAR(100)

);

E fiz a população da tabela:

INSERT INTO pessoa (id, nome, telefone, email)

VALUES

  (1, 'Tony Stark', '(21) 555-1234', 'tony.stark@starkindustries.com'),

  (2, 'Steve Rogers', '(21) 555-5678', 'steve.rogers@avengers.com'),

  (3, 'Thor Odinson', '(21) 555-9012', 'thor.odinson@asgard.com'),

  (4, 'Natasha Romanoff', '(21) 555-3456', 'natasha.romanoff@shield.com'),

  (5, 'Clint Barton', '(21) 555-7890', 'clint.barton@avengers.com'),

  (6, 'Bruce Banner', '(21) 555-2345', 'bruce.banner@starkindustries.com'),

  (7, 'Peter Parker', '(21) 555-6789', 'peter.parker@dailybugle.com'),

  (8, 'Carol Danvers', '(21) 555-1234', 'carol.danvers@usaf.mil'),

  (9, 'Wanda Maximoff', '(21) 555-5678', 'wanda.maximoff@avengers.com'),

  (10, 'Vision', '(21) 555-9012', 'vision@avengers.com');

Feito isso agora irmos criar um container NodeJS para rodar nossa API.

Startar o Docker

Agora iremos criar uma pasta que ser compartilhada com o NodeJS, abra o terminal de comando do Windows (Power shell ou Prompt de comando) e digite o seguinte comando:

**mkdir K:\nodejsql** (você pode criar a pasta com o nome que achar melhor)

Texto

Descrição gerada automaticamente

Agora iremos criar os arquivos necessários para que a nossa aplicação funcione

**package.json/index.js/dbconexao.js/Dockerfile**

**dbconexao.js**

Esse código é usado para conectar o programa NodeJS com o banco de dados MySQL. Ele usa uma ferramenta chamada "módulo mysql" do NodeJS para criar uma conexão com o banco de dados. Para estabelecer essa conexão, é preciso informar dados como endereço do banco de dados, número da porta, nome do usuário, senha e o nome do banco de dados que se quer acessar.

tive o seguinte erro abaixo, resolvi colocando em vez de localhost o ip da minha maquina host: '192.168.1.131',na hora de configurar a conexao no arquivo.

original: Error: connect ECONNREFUSED 127.0.0.1:3306

2023-03-13 16:51:03 at TCPConnectWrap.afterConnect [as oncomplete] (node:net:1494:16) {

2023-03-13 16:51:03 errno: -111,

2023-03-13 16:51:03 code: 'ECONNREFUSED',

2023-03-13 16:51:03 syscall: 'connect',

2023-03-13 16:51:03 address: '127.0.0.1',

2023-03-13 16:51:03 port: 3306,

2023-03-13 16:51:03 fatal: true

const mysql = require('mysql');

const conexao = mysql.createConnection({

  host: '192.168.1.131',

  port: 3306,

  user: 'root',

  password: '123',

  database: 'nodejsql'

});

conexao.connect((*err*) => {

  if (*err*) {

    console.error('Erro ao conectar com o MySQL: ', *err*);

  } else {

    console.log('Conexão estabelecida com o MySQL!');

  }

});

*module*.*exports* = conexao;

**index.js**

Esse código é usado para criar uma aplicação web que funciona como uma API. Quando essa API recebe uma requisição HTTP com o parâmetro "id", ele faz uma busca no banco de dados MySQL e retorna os dados da pessoa correspondente a esse "id".

const express = require('express');

const app = express();

const conexao = require('./dbconexao.js');

app.get('/pessoa/:id', (*req*, *res*) => {

  const id = *req*.params.id;

  const query = `SELECT nome, telefone, email FROM pessoa WHERE id = ${id}`;

  conexao.query(query, (*err*, *results*) => {

    if (*err*) {

      throw *err*

    } else {

*res*.send(*results*[0]);

    }

  });

});

console.log(conexao)

app.listen(3000, () => {

  console.log('API rodando na porta 3000!');

});

**Dockerfile**

O Dockerfile é uma ferramenta que usamos para criar nossas próprias imagens. Ele funciona como uma espécie de "receita" para construir um container, permitindo que possamos criar um ambiente personalizado para o nosso projeto, seja ele pessoal ou empresarial. Em resumo, o Dockerfile é uma forma de criar imagens de containers únicas e personalizadas para o nosso projeto.

FROM node:latest

WORKDIR /app

COPY package\*.json ./

RUN npm install

COPY . .

EXPOSE 3000

CMD [ "npm", "start" ]

**package.json**

É um arquivo de configuração utilizado para estipular e configurar dependências do seu projeto (quais outros pacotes ele vai precisar para ser executado) e scripts automatizados. Através dele conseguimos deixar claro uma "receita" para executar um projeto.

{

  "name": "nome-do-projeto",

  "version": "1.0.0",

  "description": "Descrição do projeto",

  "main": "index.js",

  "scripts": {

    "start": "node index.js",

    "test": "echo \"Error: no test specified\" && exit 1"

  },

  "author": "Nome do autor",

  "license": "ISC",

  "dependencies": {

    "express": "^4.17.1",

    "mysql": "^2.18.1"

  }

}

Para criar todos esses arquivos basta abrir um arquivo TXT dentro da pasta criada anteriormente e salve esses códigos com seus respectivos nomes.

Agora abra o seu terminal de comando digite o seguinte comando:

**Cd K:nodejsql**



Você deverá acessar a pasta que foi criada anteriormente para poder dar prosseguimento a instalação

Agora que você acessou a pasta digite o seguinte comando abaixo:

**docker build -t xxxxx . (não pode esquecer do ponto no final)**

**(xxxxx = nome de sua preferência)**

**docker build -t nodejsql01 . (esse foi o nome que eu utilizei para criar minha imagem)**

O comando docker build do docker cria uma imagem a partir de um Dockerfile e de um contexto. O contexto da construção é o conjunto de arquivos em um PATH ou URL de local especificado. O PATH é um diretório em seu sistema de arquivos local. O URL é um local do repositório Git.

O parâmetro "-t" é usado para dar um nome e uma identificação

O ponto final (.) significa que o arquivo com as instruções para criar a imagem, chamado Dockerfile, está no mesmo diretório onde você está executando o comando.

Texto

Descrição gerada automaticamente

Agora vamos instalar o **npm install express**

O que é o npm install express?

O npm é o Gerenciador de Pacotes do Node (Node Package Manager) que vem junto com ele e que é muito útil no desenvolvimento node. Por anos, o Node tem sido amplamente usado por desenvolvedores JavaScript para compartilhar ferramentas, instalar vários módulos e gerenciar suas dependências.

Texto

Descrição gerada automaticamente

Agora vamos instalar **npm install mysql@latest**

O comando npm install mysql@latest serve para instalar a versão mais nova do pacote MySQL, que é um conjunto de ferramentas para trabalhar com bancos de dados relacionais, dentro de um projeto criado com NodeJS.

Agora iremos utilizar essa linha de comando para executar um contêiner Docker para um aplicativo Node.js

**docker run --name nodejsql01 -p 3000:3000 -v k:\nodejsql:/app dockernodejs**

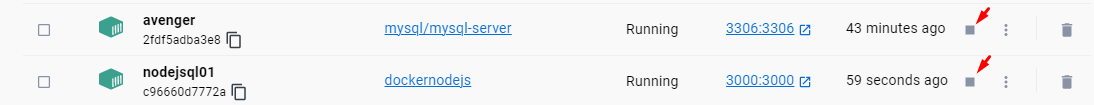
**nodejsql01**= nome que você dará ao seu container

**k:\nodejsql**= pasta raiz do seu trabalho, criada na primeira etapa

**dockernodejs**= nome da imagem criada com o docker build

**docker run --name nodejsql02 -p 3000:3000 -v k:\nodejsql02:/app servernodejs** (fiz dessa maneira)

Podemos ver na imagem abaixo que o container está sendo executado de maneira correta, tanto o NodeJS como o MySql



Ao startar o container podemos ver que a conexao foi estabelecida



Ao dar o comando **npm start** ou **node index.js** no terminal de comando



Agora acesse o seu navegador e digite: [**http://seu-ip/pessoa/ID**](http://seu-ip/pessoa/ID)

[**http://192.168.1.131:3000/pessoa/3**](http://192.168.1.131:3000/pessoa/3%20/)  **/** [**http://localhost/pessoa/8**](http://localhost/pessoa/8)

caso você não saiba seu IP utilize o comando :**ipconfig**

Va até a linha IPV4 e verifique o número do seu IP conforme a imagem abaixo

Uma imagem contendo Padrão do plano de fundo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**FONTES DE PESQUISA UTILIZADAS:**

<https://expressjs.com/en/starter/hello-world.html>

<https://www.w3schools.com/nodejs/nodejs_mysql.asp>

<https://stackoverflow.com/questions/66038165/error-connect-econnrefused-127-0-0-15432-when-connecting-with-nodejs-program>

<https://cursos.alura.com.br/forum/topico-error-connect-econnrefused-127-0-0-1-3306-196274>

<https://www.youtube.com/watch?v=g5ij7NIPR2s&ab_channel=LuizTools>

<https://www.luiztools.com.br/post/como-usar-nodejs-mysql/>

<https://www.luiztools.com.br/post/criando-uma-webapi-com-nodejs-e-mysql/>

<https://www.w3resource.com/node.js/nodejs-mysql.php>

<https://expressjs.com/pt-br/guide/database-integration.html>

<https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction>

<https://pt.linkedin.com/pulse/construindo-uma-api-com-nodejs-e-express-yesmin-marie-soret-lahoud>

<https://www.treinaweb.com.br/blog/criando-rotas-com-express-js>

<https://cursos.alura.com.br/forum/topico-nao-retorna-erro-ao-digitar-um-id-inexistente-229580>