# Realizado por Viviane, e Gilmar

# **Como Construir uma Aplicação Node.js com o Docker**

## **Pré-requisitos**

Para seguir este tutorial, você vai precisar ter instalado em seu sistema operacional:

* Docker versão atual
* Node.js e npm versão atual.

## **Passo 1 — Instalando as Dependências da Sua Aplicação**

Na pasta onde será criado o seu projeto, crie um arqui com o nome: **package.json**, dentro do arquivo será colocardo as informações do seu projeto bem como as dependências nodejs: ( em **"description":** coloque a descrição do seu projeto, e em **“author”:** coloque o nome do autor do projeto.

{

"name": "docker\_web\_app",

"version": "1.0.0",

"description": "Node.js on Docker",

"author": "First Last <first.last@example.com>",

"main": "server.js",

"scripts": {

"start": "node server.js"

},

"dependencies": {

"express": "^4.16.1"

}}

Após ter criado o **packager.json,** e execute em seu terminal dentro do diretório onde foi criado o projeto o seguinte comando: **npm install.** Isso permitirá a instalação de todas as dependência necessárias para o projeto.

Em seguida, crie um outro arquivo no diretório chmado: **server.js,** este arquivo irá definir as requisições web para sua aplicação, usando o **Express.js.**

'use strict';const express = require('express');// Constantsconst PORT = 8080;const HOST = '0.0.0.0';// Appconst app = express();

app.get('/', (req, res) => {

res.send('Hello World');});

app.listen(PORT, HOST);

console.log(`Running on http://${HOST}:${PORT}`);

## **Passo 2 — Criando o Dockerfile**

Dentro do diretório do seu projeto crie um arquivo vazio chamado: **Dockerfile,** digitando o seguinte comando em seu terminal:

touch Dockerfile

Em seguida abra o arquivo **Dockerfile,** em seu editor de texto.

O qual permitirá será incluir os parametros do docker.

FROM node:10

# Cria o diretório da aplicação

WORKDIR /usr/src/app

# Instala as dependências

#Realiza a copia do package.json

# where available (npm@5+)

COPY package\*.json ./RUN npm install

# If you are building your code for production

# RUN npm ci --only=production

# Bundle app source

COPY . .

EXPOSE 8080

CMD [ "node", "server.js" ]

## **Passo 3 —** arquivo .dockerignore e Construindo sua imagem

Crei em seu diretório um arquivo chmado: **.dockerignore,** lembrando que este arquivo deverá ser criado no mesmo diretório do **Dockerfile.** Em seguida abra o aquivo **.dockerignore,**  e coloque as seguinte informações:

node\_modules

npm-debug.log

Logo após iremos criar a imagem do Docker.

No diretório principal do seu projeto onde está o **Dockerfile,** execute o seguinte comando no terminal:

docker build -t <your username>/node-web-app .

Subistitua <your username> pelo usuário do dockerHub.

A sua imagem poderá ser lista pelo docker:

$ docker images# Example

REPOSITORY TAG ID CREATED

node 10 1934b0b038d1 5 days ago<your username>/node-web-app latest d64d3505b0d2 1 minute ago

## **Passo 4 — Executando a imagem docker**

No diretório do seu projeto execute o seguinte comando no terminal:

docker run -p 49160:8080 -d <your username>/node-web-app

Verificando a saida de sua aplicação:

# Get container ID

$ docker ps

# Print app output

$ docker logs <container id>

# Example

Running on http://localhost:8080

## **Passo 4 — Testando sua aplicação**

Para realizar o teste de sua aplicação execute o seguinte comando no terminal:

$ docker ps

# Example

ID IMAGE COMMAND ... PORTS

ecce33b30ebf <your username>/node-web-app:latest npm start ... 49160->8080

O docker irá passar o parâmetro da porta 8080 para 49160 a qual foi mapeada pelo docker.

Digite em seu navegador web o seguinte endereço: **localhost:49160,** se correu tudo bem irá sugir na tela a seguinte mensagem: Hello World