

## Crie uma API RESTful com Node.js e MongoDB | CRUD com Node, Express e Mongoose

Criar a pasta APINODEJS digitar `npm init -y`  
Agora vamos puxar os pacotes adicionais

```
npm install express nodemon mongoose
```

Adicionar o Pacote

Index.js

```
// config inicial chamar o express vai procurar o módulo
```

```
const express = require('express')
```

```
const app = express() // Inicializar as apps
```

```
//forma de ler JSON UTILIZAR MIDDLEWARES
```

```
app.use( //criando o MIDDLEWARES
```

```
express.urlencoded({
```

```
extended: true,
```

```
}),
```

```
)
```

```
app.use(express.json())
```

```
//rota inicial GET pegar algo so servidor endpoint
```

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

```
//mostrar requisição mostrar a resposta que vai ser JSON
```

```
res.json({ message: 'Oi Express'})
```

```
})
```

```
//entregar a porta
```

```
app.listen(3000)
```

Package.JSON

```
{
```

```
"name": "arquivo",
```

```
"version": "1.0.0",
```

```
"main": "index.js",
```

```
"scripts": {  
  "start": "nodemon ./index.js localhost 3001"  
},  
"keywords": [],  
"author": "",  
"license": "ISC",  
"description": "",  
"dependencies": {  
  "express": "^4.19.2",  
  "mongoose": "^8.3.1",  
  "nodemon": "^3.1.0"  
}  
}
```

Digite NPM START

```
const express = require('express')
```

`require('express')`: `require` é uma função do Node.js usada para carregar módulos. Quando você passa

Index

Chamar na Postman o Json

<https://www.postman.com/downloads/>

Continuação

Classe pacote Models/Person

Index.js

Abrir o mongo DB e conectar na aplicação então configurar o mongo DB

Entrar no Site e baixar

Só dar next

No Compass já tenho um URI isso é uma string de conexão do Mongo DB

Aperte Coneect

Vamos cria um banco de dados  
Insert Document

Insira um dado via JSON

Adicione

// depois do db

```
const mongoose = require('mongoose')
```

Se der erro instalar o npm install --save mongoose

//cONEXÃO NA MÁQUINA LOCAL CLS SE FUNCIONAR VAI NO THEN SENÃO APONTA O ERRO

```
mongoose.connect('mongodb://localhost:27017/ARQUIVO')
```

```
.then(() => {
```

```
  console.log('Conectou ao banco!')
```

```
  app.listen(3000)
```

```
})
```

```
.catch((err) => console.log(err))
```

Criar uma pasta model

Como iremos trabalhar com os dados no Banco

Criar uma pasta model entidade Person

Agora vamos enviar Dados via Json

```
{  
  "name": "Thaigo",  
  "salary": 8000,  
  "approved": false  
}
```

ee

Atualização do Tutorial: Implementando POST, GET, PUT e DELETE com Postman e MongoDB

#### 1. Abrindo o Postman e Testando o POST

Abra o Postman e selecione o método POST.

Cole a URL: `http://localhost:3000/person`

Clique em Body > Raw > escolha JSON como tipo de dado.

Adicione um corpo JSON como este:

```
{
  "name": "Carlos",
  "salary": 5000,
  "approved": true
}
```

Clique em SEND para enviar.

#### 2. Criando o GET para buscar todos os registros

No Postman, selecione o método GET.

Cole a URL: `http://localhost:3000/person`

Clique em SEND. Você verá todos os registros cadastrados.

#### 3. Criando o GET por ID

Depois de fazer o POST, o MongoDB gera um ID.

Copie esse ID do retorno da resposta.

Agora selecione o método GET e cole a URL: `http://localhost:3000/person/ID_DO_DOCUMENTO`

Substitua ID\_DO\_DOCUMENTO pelo ID real. Clique em SEND.

#### 4. Criando o PUT para atualizar dados

No Postman, selecione o método PUT.

Cole a URL: `http://localhost:3000/person/ID_DO_DOCUMENTO`

Clique em Body > Raw > JSON e modifique o JSON como quiser.

Exemplo:

```
{
  "name": "Carlos Silva",
  "salary": 5500,
  "approved": true
}
```

Clique em SEND para atualizar.

#### 5. Criando o DELETE para remover dados

No Postman, selecione o método DELETE.

Cole a URL: `http://localhost:3000/person/ID_DO_DOCUMENTO`

Clique em SEND. O registro será deletado do MongoDB.