

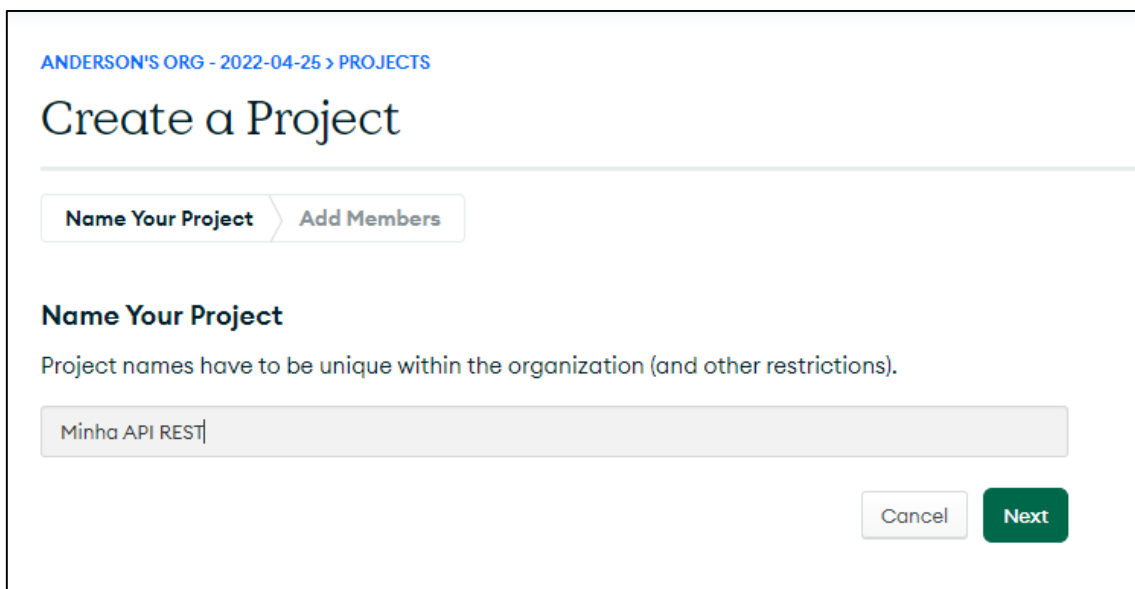
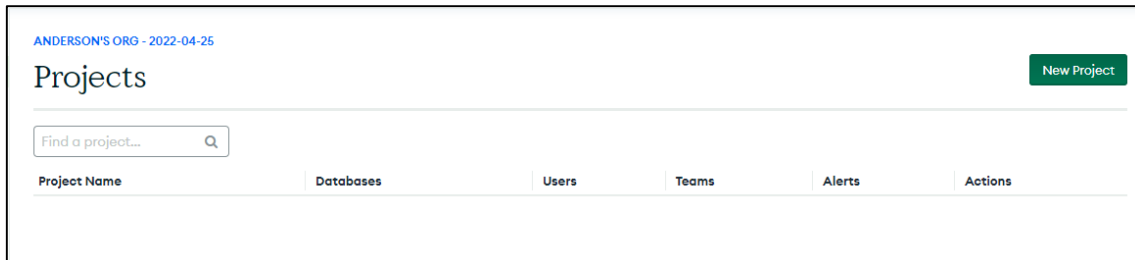
# DDM-II – Criação de API Rest com MongoDB

Prof. Anderson Vanin

## PARTE 02

12- Crie um projeto no Atlas DB e salve um cluster gratuito.

Crie uma conta gratuita em: <https://www.mongodb.com/atlas/database>



Agora crie um Banco de Dados

## Database Deployments



### Create a database

Choose your cloud provider, region, and specs.

**Build a Database**

Once your database is up and running, live migrate an existing MongoDB database into Atlas with our [Live Migration Service](#).

Renomeie o nome do Cluster e clique em **Create Cluster**.

Cluster Name

APICluster

**One time only:** once your cluster is created, you won't be able to change its name.

Cluster names can only contain ASCII letters, numbers, and hyphens.

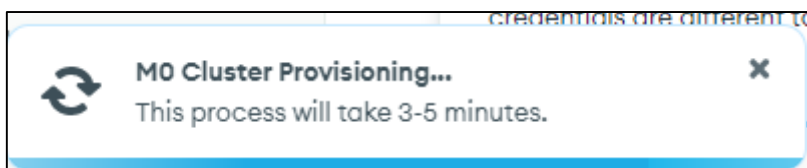
**FREE**

Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

[Back](#)


**Create Cluster**

Este processo demora alguns minutos para ser terminado.



Enquanto o cluster é criado, preencha um nome de usuário e senha para o banco.

**Username**

**Password** 




**Create User**

Altere o endereço de IP, para que sua aplicação possa ser acessada de qualquer IP

**Add entries to your IP Access List**

Only an IP address you add to your Access List will be able to connect to your project's clusters.

| IP Address | Description                                    |                  |                           |
|------------|--|------------------|---------------------------|
| 0.0.0.0/0  | <input type="text" value="Enter description"/> | <b>Add Entry</b> | Add My Current IP Address |

**Ip: 0.0.0.0/0 → permite acesso de qualquer lugar**

### Database Deployments

Find a database deployment...

**+ Create**

**APICluster**

Connect View Monitoring Browse Collections ...

FREE SHARED

**R 0**

**W 0**

Last 26 seconds

100.0/s

**Connections 0**

Last 26 seconds

100.0

**In 0.0 B/s**

**Out 0.0 B/s**

Last 26 seconds

100.0 B/s

**Data Size 0.0 B**

Last 26 seconds

512.0 MB

**Enhance Your Experience**

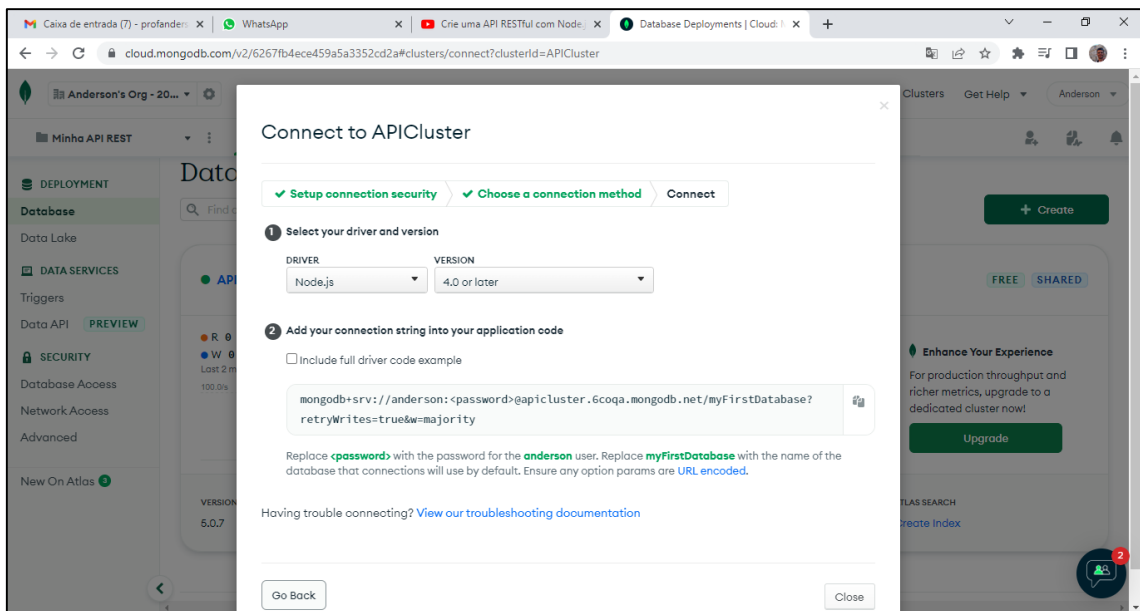
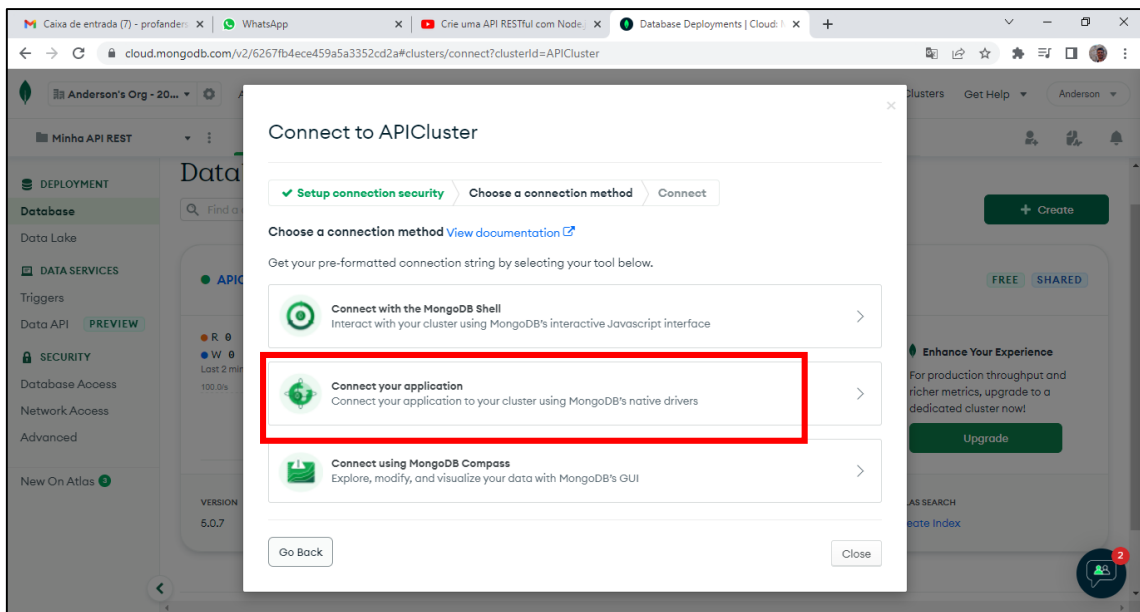
For production throughput and richer metrics, upgrade to a dedicated cluster now!

**Upgrade**

| VERSION | REGION                      | CLUSTER TIER         | TYPE                  | BACKUPS  | LINKED REALM APP | ATLAS SEARCH                 |
|---------|-----------------------------|----------------------|-----------------------|----------|------------------|------------------------------|
| 5.0.7   | AWS / Sao Paulo (sa-east-1) | M0 Sandbox (General) | Replica Set - 3 nodes | Inactive | None Linked      | <a href="#">Create Index</a> |

Uma vez que seu banco foi criado, clique em **CONNECT**, para obter o link de conexão que será utilizado pela nossa API.

Escolha conectar aplicação.



Copie o link

```
mongodb+srv://anderson:<password>@apicluster.6coqa.mongod
b.net/myFirstDatabase?retryWrites=true&w=majority
```

Substitua a senha e coloque um nome para seu banco de dados. Faça isso em seu código:

```

JS index.js > ...
1  // config inicial
2  const express = require('express')
3  const mongoose = require('mongoose')
4  const app = express()
5  // forma de ler JSON => utilizar middlewares

```

...

```

24 // entregar uma porta
25 const DB_USER = 'anderson'
26 const DB_PASSWORD = encodeURIComponent('y3sXPwf8Zjo5Xlyn')
27 mongoose
28 .connect(
29   `mongodb+srv://${DB_USER}:${DB_PASSWORD}@apiclust.6coqa.mongodb.net/bancoDaAPI?retryWrites=true&w=majority`
30 )
31 .then(()=>{
32   console.log('Conectamos ao DB!')
33   app.listen(3000)
34 })
35 .catch((err)=>{
36   console.log(err)
37 })
38

```

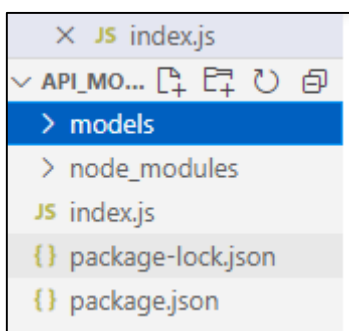
PROBLEMAS SAÍDA CONSOLE DE DEPURAÇÃO TERMINAL

```

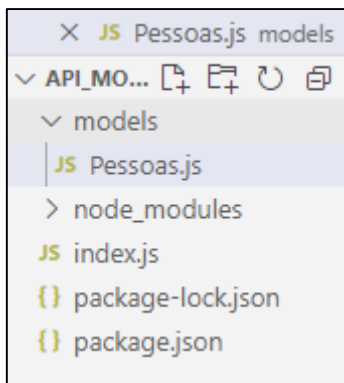
[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node ./index localhost 3000 index.js`
Conectamos ao DB!

```

Agora que estamos conectados, precisamos criar uma pasta chamada models.



Crie um arquivo chamado Pessoas.js (nota: utilizar letras maiúsculas pois se trata de uma classe)



```
JS index.js JS Pessoas.js X
models > JS Pessoas.js > ...
1  const mongoose = require('mongoose')
2
3  const Pessoas = mongoose.model('Pessoas',{
4      nome: String,
5      salario: Number,
6      aprovado: Boolean,
7  })
8
9  module.exports = Pessoas
```

Agora vamos criar uma rota para adicionar dados em nosso banco.

Faça a importação do módulo que acabamos de criar no arquivo index.js

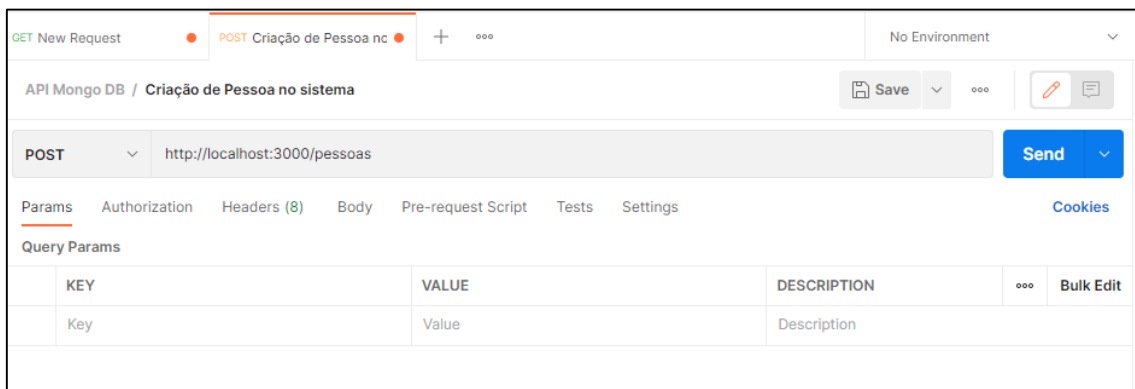
```
JS index.js > [P] Pessoas
1  // config inicial
2  const express = require('express')
3  const mongoose = require('mongoose')
4  const app = express()
5
6  const Pessoas = require('./models/Pessoas')
7
```

```

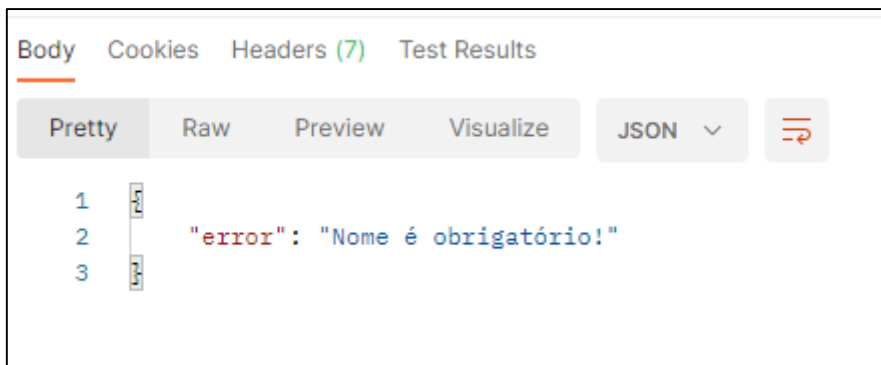
14 app.use(express.json())
15
16 // rotas da API
17 app.post('/pessoas', async(req,res)=>{
18
19     const {nome,salario,aprovado} = req.body
20     //{nome:"anderson",salario:5000,aprovado:false}
21
22     if(!nome){
23         res.status(422).json({error: 'Nome é obrigatório!'})
24     }
25
26     const pessoas = {
27         nome,
28         salario,
29         aprovado
30     }
31
32     try {
33         await Pessoas.create(pessoas)
34         res.status(201).json({message: 'Pessoa inserida com sucesso!'})
35     } catch (error) {
36         res.status(500).json({error:error})
37     }
38
39 })
40 // rota inicial / endpoint

```

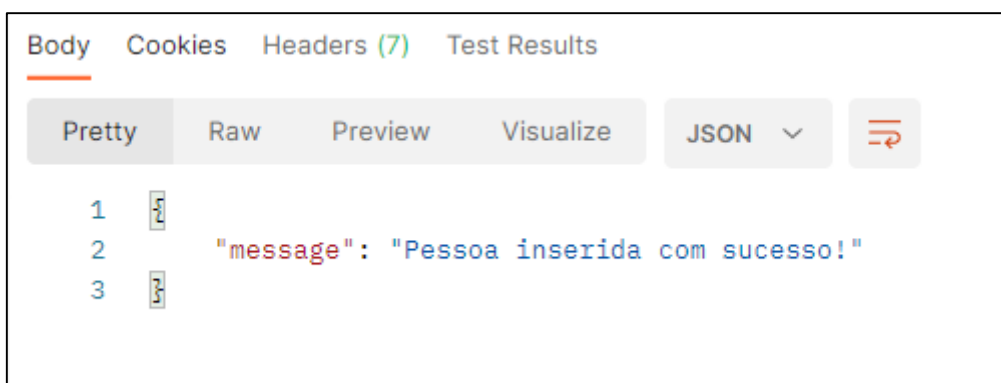
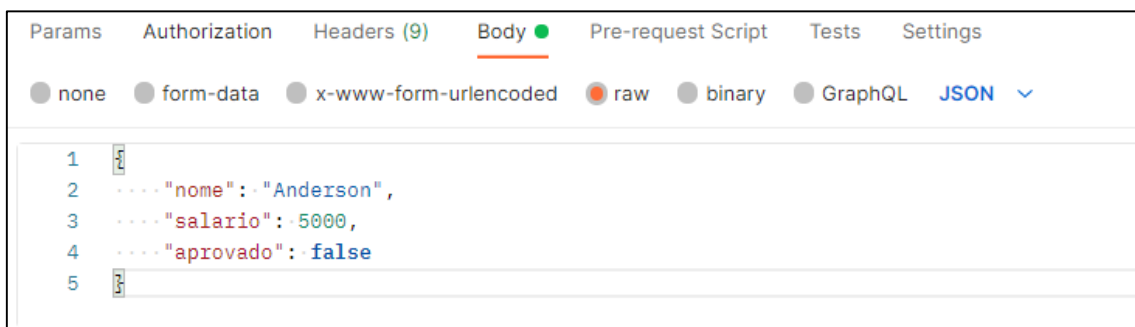
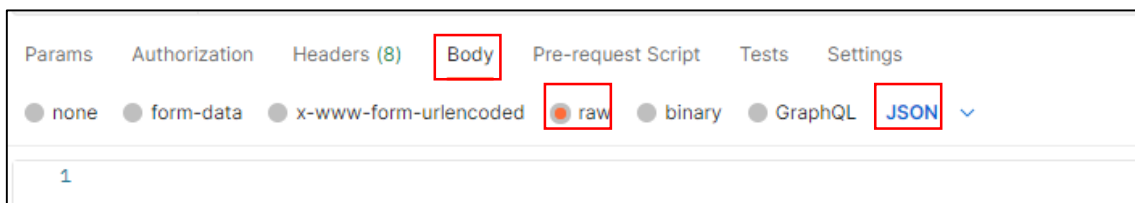
Agora vamos testar esta rota através do POSTMAN



Se você não enviar nenhuma informação será informada a mensagem:

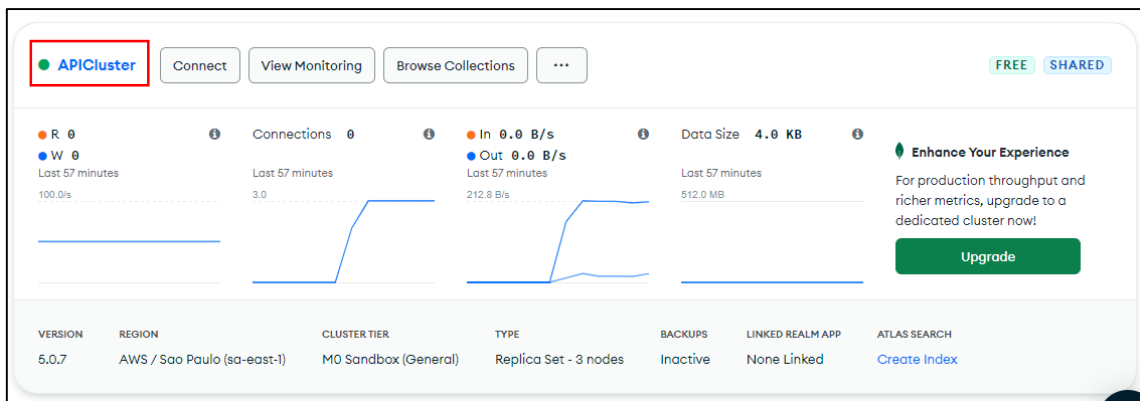


Para enviar dados faça o seguinte:



No MongoDB clique sobre o nome de seu cluster





Agora clique em **Collections**

The screenshot shows the MongoDB Atlas database interface. On the left, there's a sidebar with a '+ Create Database' button and a search bar. Below the search bar, there's a list of namespaces: 'bancoDaAPI' and 'pessoas'. The main area displays the 'bancoDaAPI.pessoas' collection. It shows 'STORAGE SIZE: 36KB', 'TOTAL DOCUMENTS: 2', and 'INDEXES TOTAL SIZE: 36KB'. There are tabs for 'Find', 'Indexes', 'Schema Anti-Patterns', 'Aggregation', and 'Search Indexes'. A 'FILTER' bar contains the query '{ field: 'value' }'. Below the filter, there's a 'QUERY RESULTS 1-1 OF 1' section showing a single document:

```
{
  "_id": ObjectId("62680a001214fb2f61d841d4"),
  "nome": "Anderson",
  "salario": 5000,
  "aprovado": false,
  "__v": 0
}
```

Conseguimos criar nossa primeira rota para ser acessada via API e inserir dados em um Banco de Dados na nuvem.

Na terceira parte vamos continuar com a criação das outras rotas!