

Curso de Node.JS

Felipe MX

<https://www.youtube.com/watch?v=EGVPrfgl5jw&list=PLSx5HT-nMTzxfYej6YiT5WrWFXBa5CuT0&index=1>

Aula 01 - Conceitos básicos

Conceitos de Backend

API - (Application Programming Interface)

Uma definição fácil de entender seria: mecanismo para conectar sistema de softwares entre si. Ou seja, se compõe de um conjunto de instruções e regras que as aplicações podem seguir para se comunicarem.

Backend

É a parte de um sistema que se encarrega de toda a lógica da aplicação. Não é diretamente acessível pelo usuário. Entre algumas das operações que realiza uma delas é atender as requisições dos recursos, assim como acessar o servidor de base de dados.

Algumas linguagens mais empregadas para programação de backend são: PHP, Ruby, C#, Java, Python, JavaScript, entre outras.

API REST (REpresentational State Transfer)

(transferência de estado representacional)

É uma arquitetura que utiliza o protocolo http para realizar a execução de operações sobre os dados, opera sob a arquitetura cliente-servidor

Em outras palavras: permite o acesso e manipulação de recursos (dados, conjunto de dados) localizados em algum servidor web; as operações mais importantes em um sistema REST são: POST (criar), GET (ler/consultar), PUT (atualizar) e DELETE (apagar).

Para executar estas operações não se requer uma interface de usuário, a manipulação se faz através de uma URI.

URI (Uniform Resource Identifier)

Identificador de recursos uniforme, ou seja, o caminho ou rota onde se localiza o recurso no sistema REST.

O que faremos?

Desenvolveremos um backend básico para a manipulação de dados:

- Clientes
- Produtos
- Pedidos

O que utilizaremos?

NodeJS: ambiente de execução para JavaScript do lado do servidor, ou seja, com JavaScript programaremos o backend.

<https://nodejs.org>

npm: Sistema de gerenciamento de pacotes para NodeJS (é instalado ao se instalar o ambiente NodeJS)

MongoDB: motor/gerenciamento de base de dados não relacional, armazena os dados em documentos (coleções de objetos, documentos tipo JSON).

<https://www.mongodb.com>

Express: Servidor web ágil para NodeJS; através do qual montaremos nossa API REST.

Requisitos antes de começar

- NodeJS (descarregue e instale a versão atual)
- MongoDB (descarregue e instale a versão atual de MongoDB Community Server)
- Compass: aplicação cliente para MongoDB
- Postman API Client: aplicação para interagir com API REST (pode se utilizar uma aplicação similar para provar uma API REST)
- Um IDE ou editor de código de sua preferência

```
md felipemx  
cd felipemx
```

```
md store-api  
cd store-api
```

Inicializando o projeto Node.JS

npm init

```
C:\felipemx\store-api>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

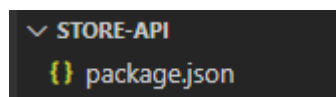
Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (store-api)
version: (1.0.0)
description: api rest
entry point: (index.js)
test command:
git repository:
keywords: api
author: Roberto Pinheiro
license: (ISC)
About to write to C:\felipemx\store-api\package.json:

{
  "name": "store-api",
  "version": "1.0.0",
  "description": "api rest",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "api"
  ],
  "author": "Roberto Pinheiro",
  "license": "ISC"
}

Is this OK? (yes) yes
```

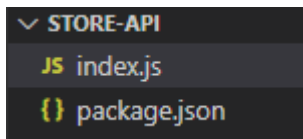
code .



package.json

```
{
  "name": "store-api",
  "version": "1.0.0",
  "description": "api rest",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "api"
  ],
  "author": "Roberto Pinheiro",
  "license": "ISC"
}
```

- Na pasta raiz do projeto adicione o arquivo `index.js`:

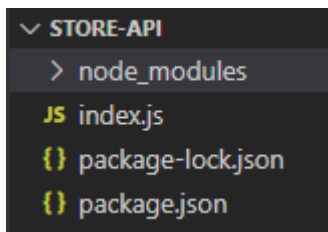


Instalando o Express

`npm install --save express`

```
PS C:\felipemx\store-api> npm install --save express
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN store-api@1.0.0 No repository field.

+ express@4.17.1
added 50 packages from 37 contributors and audited 50 packages in 14.08s
found 0 vulnerabilities
```



package.json

```
{
  "name": "store-api",
  "version": "1.0.0",
  "description": "api rest",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "api"
  ],
  "author": "Roberto Pinheiro",
  "license": "ISC",
  "dependencies": {
    "express": "^4.17.1"
  }
}
```

index.js

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

app.get('/', (req, res) => {
  res.send('Ola mundo NodeJS!');
});

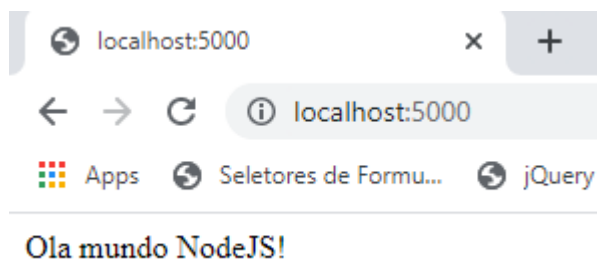
app.listen(5000, () => {
  console.log('Servidor web Express em execução!');
})
```

node index.js

```
PS C:\felipemx\store-api> node index.js
Servidor web Express em execução!
█
```

- No browser:

localhost:5000



Instalando nodemon

npm install -g nodemon

- Instala o nodemon de forma global.

```
PS C:\felipemx\store-api> npm install -g nodemon
C:\Users\beto1\AppData\Roaming\npm\nodemon -> C:\Users\beto1\AppData\Roaming\npm\node_modules\nodemon\bin\nodemon.js

> nodemon@2.0.7 postinstall C:\Users\beto1\AppData\Roaming\npm\node_modules\nodemon
> node bin/postinstall || exit 0

Love nodemon? You can now support the project via the open collective:
> https://opencollective.com/nodemon/donate

npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@~2.3.1 (node_modules\nodemon\node_modules\chokidar\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.3.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ nodemon@2.0.7
added 119 packages from 53 contributors in 38.246s
```

package.json

```
{
  "name": "store-api",
  "version": "1.0.0",
  "description": "api rest",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon index.js"
  },
  "keywords": [
    "api"
  ],
  "author": "Roberto Pinheiro",
  "license": "ISC",
  "dependencies": {
    "express": "^4.17.1"
  }
}
```

npm start

```
PS C:\felipemx\store-api> npm start

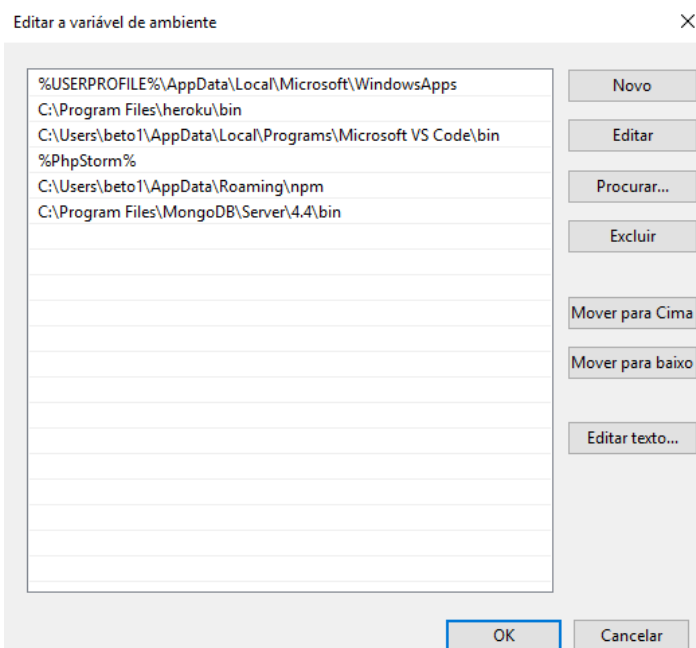
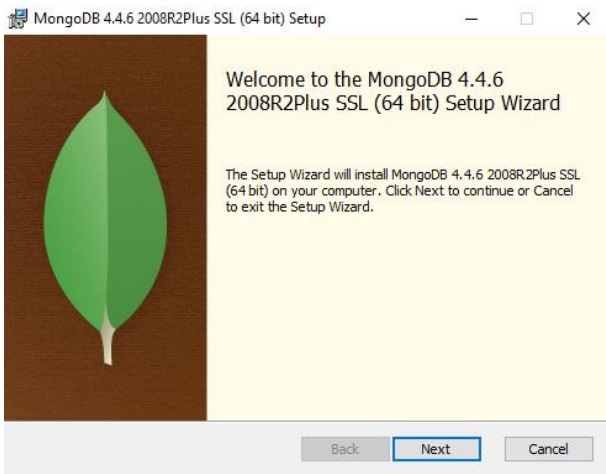
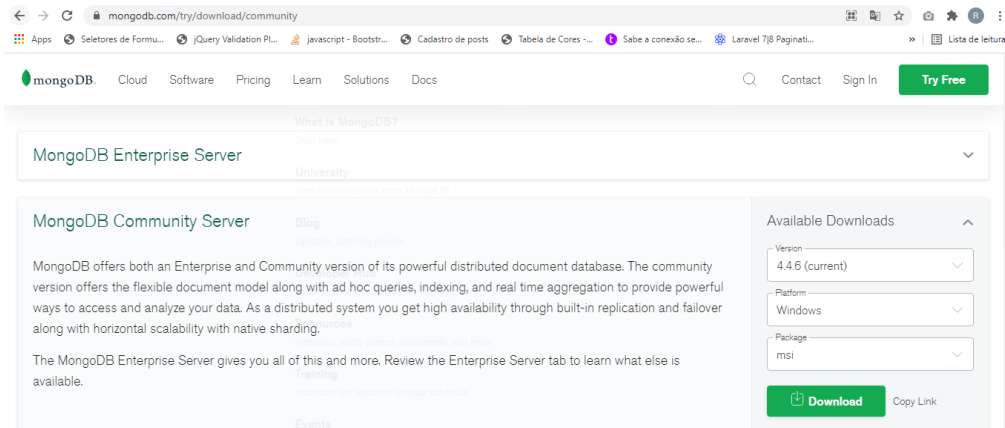
> store-api@1.0.0 start C:\felipemx\store-api
> nodemon index.js

[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Servidor web Express em execução!
```

Aula 02 - Iniciar MongoDB

- Baixe e instale o MongoDB

<https://www.mongodb.com/try/download/community>



- Dentro de **C:** crie a pasta **data** e dentro dela a subpasta **db**:

```
C:\>md data
C:\>cd data
C:\data>md db
C:\data>cd db
C:\data\db>
```

- No Linux - Ubuntu, dê permissão de escrita para **C:/data/db**:

`sudo chmod 777 /data/db`

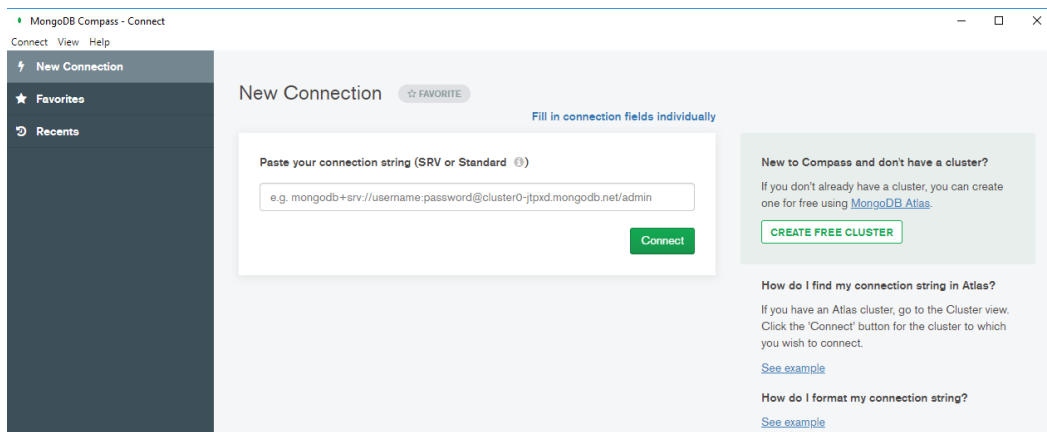
Executando o servidor do MongoDB

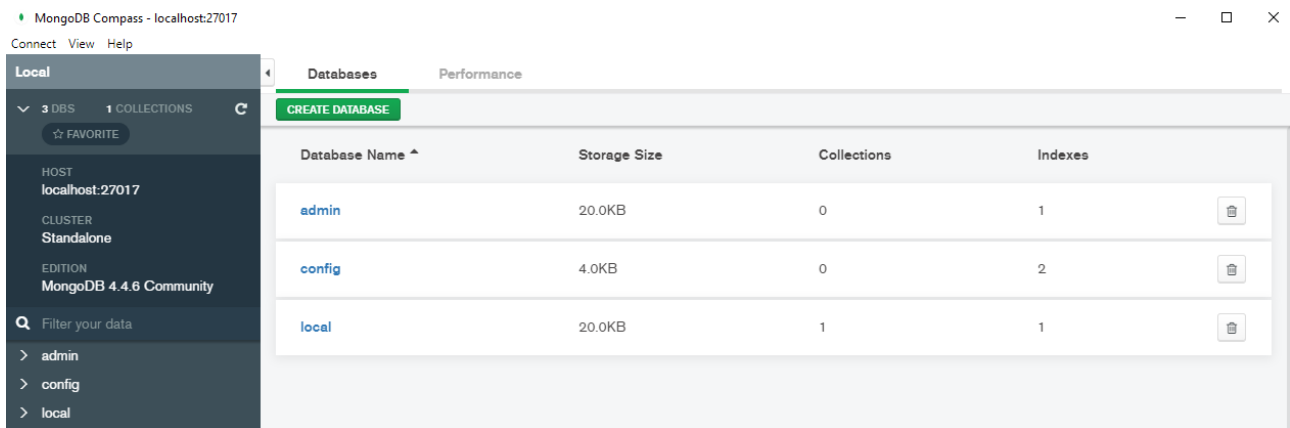
mongod

```
Microsoft Windows [vers o 10.0.16299.1087]
(c) 2017 Microsoft Corporation. Todos os direitos reservados.

C:\Users\beto1>mongod
{"t":{"$date":"2021-06-05T14:30:09.926-03:00"},"s":"I",  "c":"CONTROL",  "id":23285,   "ctx":"main","msg":"Automatically disabling TLS 1.0, to force-enable TLS 1.0 spec
ify --sslDisabledProtocols 'none'"}
{"t":{"$date":"2021-06-05T14:30:11.378-03:00"},"s":"W",  "c":"ASIO",      "id":22601,   "ctx":"main","msg":"No TransportLayer configured during NetworkInterface startup"
}
{"t":{"$date":"2021-06-05T14:30:11.380-03:00"},"s":"I",  "c":"NETWORK",  "id":4648602, "ctx":"main","msg":"Implicit TCP FastOpen in use."}
{"t":{"$date":"2021-06-05T14:30:11.382-03:00"},"s":"I",  "c":"STORAGE",  "id":4615611, "ctx":"initandlisten","msg":"MongoDB starting", "attr":{"pid":7668,"port":27017,"d
bPath":"C:/data/db/","architecture":"64-bit","host":"DESKTOP-NUIKR9E"}}
{"t":{"$date":"2021-06-05T14:30:11.383-03:00"},"s":"I",  "c":"CONTROL",  "id":23398,   "ctx":"initandlisten","msg":"Target operating system minimum version", "attr":{"ta
rgetMinOS":"Windows 7/Windows Server 2008 R2"}}
{"t":{"$date":"2021-06-05T14:30:11.383-03:00"},"s":"I",  "c":"CONTROL",  "id":23403,   "ctx":"initandlisten","msg":"Build Info", "attr":{"buildInfo":{"version":"4.4.6","
gitVersion":"7266213c2c3eab37d9358d5e7bad7f5c1d0d0d7","modules":[],"allocator":"tcmalloc","environment":{"distmod":"windows","distarch":"x86_64","target_arch":"x86_64"
}}}}
{"t":{"$date":"2021-06-05T14:30:11.383-03:00"},"s":"I",  "c":"CONTROL",  "id":51765,   "ctx":"initandlisten","msg":"Operating System", "attr":{"os":{"name":"Microsoft Wi
ndows 10","version":"10.0 (build 16299)"}}}
{"t":{"$date":"2021-06-05T14:30:11.383-03:00"},"s":"I",  "c":"CONTROL",  "id":21951,   "ctx":"initandlisten","msg":"Options set by command line", "attr":{"options":{}}}
{"t":{"$date":"2021-06-05T14:30:11.399-03:00"},"s":"I",  "c":"STORAGE",  "id":22315,   "ctx":"initandlisten","msg":"Opening WiredTiger", "attr":{"config":"create,cache_s
ize=1444M,session_max=39800,eviction=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),fi
le_manager=(close_idle_time=100000,close_scan_interval=10,close_handle_minimum=250),statistics_log=(wait=0),verbose=[recovery_progress,checkpoint_progress,compact_progr
ess]"}
{"t":{"$date":"2021-06-05T14:30:11.980-03:00"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx":"initandlisten","msg":"WiredTiger message", "attr":{"message":"[1622914211:9
80374][7668:140718874837840], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global recovery timestamp: (0, 0)"}
{"t":{"$date":"2021-06-05T14:30:11.981-03:00"},"s":"I",  "c":"STORAGE",  "id":22430,   "ctx":"initandlisten","msg":"WiredTiger message", "attr":{"message":"[1622914211:9
80374][7668:140718874837840], txn-recover: [WT_VERB_RECOVERY | WT_VERB_RECOVERY_PROGRESS] Set global oldest timestamp: (0, 0)"}
{"t":{"$date":"2021-06-05T14:30:12.133-03:00"},"s":"I",  "c":"STORAGE",  "id":4795906, "ctx":"initandlisten","msg":"WiredTiger opened", "attr":{"durationMillis":733}}
{"t":{"$date":"2021-06-05T14:30:12.138-03:00"},"s":"I",  "c":"RECOVERY", "id":23987,   "ctx":"initandlisten","msg":"WiredTiger recoveryTimestamp", "attr":{"recoveryTimes
tamp":{"$timestamp":{"t":"0","i":0}}}}
{"t":{"$date":"2021-06-05T14:30:12.425-03:00"},"s":"I",  "c":"STORAGE",  "id":4366408, "ctx":"initandlisten","msg":"No table logging settings modifications are required
for existing WiredTiger tables", "attr":{"loggingEnabled":true}}
{"t":{"$date":"2021-06-05T14:30:12.426-03:00"},"s":"I",  "c":"STORAGE",  "id":22262,   "ctx":"initandlisten","msg":"Timestamp monitor starting"}
{"t":{"$date":"2021-06-05T14:30:12.595-03:00"},"s":"W",  "c":"CONTROL",  "id":22120,   "ctx":"initandlisten","msg":"Access control is not enabled for the database. Read
and write access to data and configuration is unrestricted", "tags":["startupWarnings"]}
{"t":{"$date":"2021-06-05T14:30:12.595-03:00"},"s":"W",  "c":"CONTROL",  "id":22140,   "ctx":"initandlisten","msg":"This server is bound to localhost. Remote systems wi
ll be unable to connect to this server. Start the server with --bind_ip address to specify which IP addresses it should serve responses from, or with --bind_ip_all to
bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning", "tags":["startupWarnings"]}
{"t":{"$date":"2021-06-05T14:30:12.600-03:00"},"s":"I",  "c":"STORAGE",  "id":20320,   "ctx":"initandlisten","msg":"createCollection", "attr":{"namespace":"admin.system.
version","uuidDisposition":"provided","uuid":{"uuid":{"$uuid":"eb8c21a9-4456-45e2-8c16-48d054be7330"},"options":{"uuid":{"$uuid":"eb8c21a9-4456-45e2-8c16-48d054be7330"
}}}}}
{"t":{"$date":"2021-06-05T14:30:12.816-03:00"},"s":"I",  "c":"INDEX",    "id":20345,   "ctx":"initandlisten","msg":"Index build: done building", "attr":{"buildUUID":null
,"namespace":"admin.system.version","index":{"id":{"commitTimestamp":{"$timestamp":{"t":"0","i":0}}}}}
{"t":{"$date":"2021-06-05T14:30:12.823-03:00"},"s":"I",  "c":"COMMAND",  "id":20450,   "ctx":"initandlisten","msg":"Setting featureCompatibilityVersion", "attr":{"newVer
```

- Vamos utilizar Mongo Compass para conectar ao servidor.





Clique no botão "CREATE DATABASE":

Database Name: **store-api**

Collection Name: **customers**

Create Database

Database Name

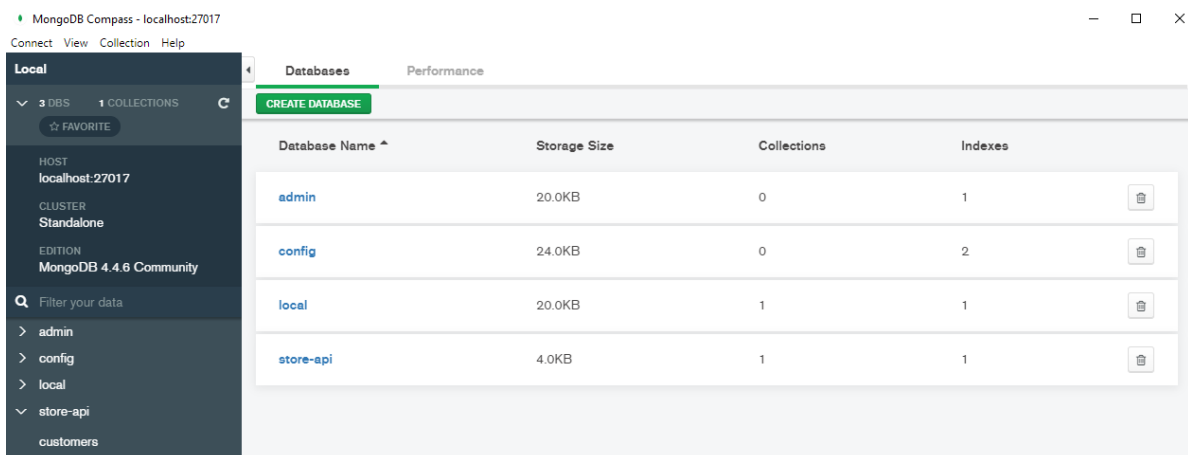
Collection Name

☐ Capped Collection ⓘ

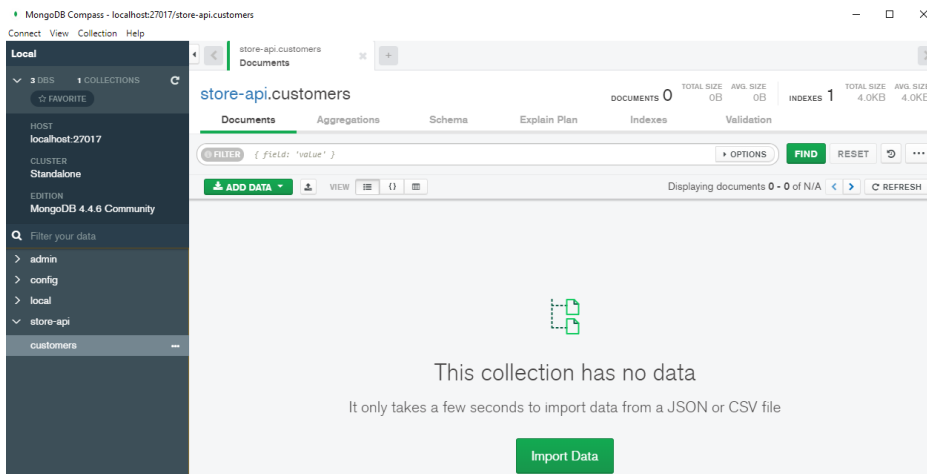
☐ Use Custom Collation ⓘ

Before MongoDB can save your new database, a collection name must also be specified at the time of creation. [More Information](#)

- Clique no botão "CREATE DATABASE"



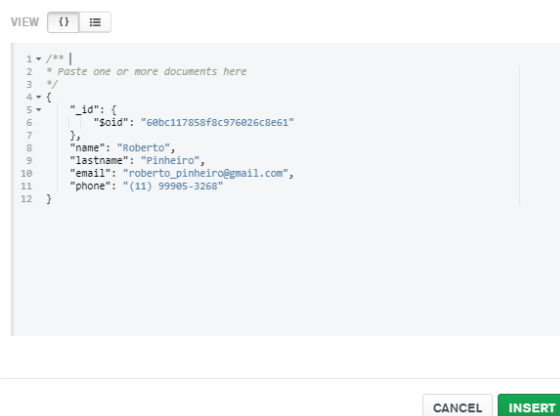
- Clique em "**customers**":



- Clique no botão "**ADD DATA -> Insert Document**":

```
"name": "Roberto",  
"lastname": "Pinheiro",  
"email": "roberto_pinheiro@gmail.com",  
"phone": "(11) 99905-3268"
```

Insert to Collection store-api.customers



- Clique no botão "**INSERT**"

```
_id: ObjectId("60bc117858f8c976026c8e61")  
name: "Roberto"  
lastname: "Pinheiro"  
email: "roberto_pinheiro@gmail.com"  
phone: "(11) 99905-3268"
```

- Insira um novo documento:

```
"name": "Felipe"  
"lastname": "Lima"  
"email": "felipe_lima@gmail.com"  
"phone": "(21) 98839-5723"
```

Insert to Collection store-api.customers

VIEW

```
1 ▾ /**
2  * Paste one or more documents here
3  */
4 ▾ {
5  ▾
6    "_id": {
7      "$oid": "60bc132058f8c976026c8e62"
8    },
9    "name": "Felipe",
10   "lastname": "Lima",
11   "email": "felipe_lima@gmail.com",
12   "phone": "(21) 98839-5723"
13 }
```

CANCEL

INSERT

```
_id: ObjectId("60bc117858f8c976026c8e61")
name: "Roberto"
lastname: "Pinheiro"
email: "roberto_pinheiro@gmail.com"
phone: "(11) 99905-3268"
```

```
_id: ObjectId("60bc132058f8c976026c8e62")
name: "Felipe"
lastname: "Lima"
email: "felipe_lima@gmail.com"
phone: "(21) 98839-5723"
```

Aula 03 - Conectar NodeJS a MongoDB Server

Instalando as dependências Mongoose Body-Parser e CORs

npm install --save mongoose body-parser cors

```
PS C:\felipemx\store-api> npm install --save mongoose body-parser cors
npm WARN store-api@1.0.0 No repository field.

+ body-parser@1.19.0
+ cors@2.8.5
+ mongoose@5.12.13
added 31 packages from 94 contributors, updated 1 package and audited 82 packages in 27.405s

2 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

index.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

const app = express();

// Configurando mongoose
mongoose.Promise = global.Promise;
mongoose.connect('mongodb://localhost/store-api', {
  useNewUrlParser: true
});

// Habilitar body-parser
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

// Habilitar cors
app.use(cors());

app.get('/', (req, res) =>{
  res.send('Ola mundo NodeJS!');
});

app.listen(5000, () => {
  console.log('Servidor web Express em execução!');
})
```

- Na pasta raiz do projeto, adicione uma pasta chamada "**models**" e dentro dela adicione um arquivo chamado "**Customer.js**".

models\Customer.js

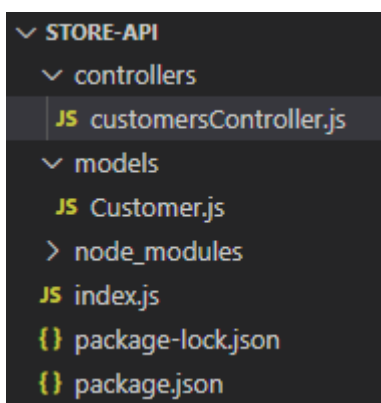
```
const mongoose = require('mongoose');

const Schema = mongoose.Schema;

const customersSchema = new Schema({
  name: {
    type: String,
    trim: true
  },
  lastname: {
    type: String,
    trim: true
  },
  email: {
    type: String,
    trim: true,
    unique: true,
    lowercase: true
  },
  phone: {
    type: String,
    trim: true,
  }
});

module.exports = mongoose.model('Customer', customersSchema);
```

- Na pasta raiz do projeto, adicione uma pasta chamada "**controllers**" e dentro dela adicione um arquivo chamado "**customersController.js**".



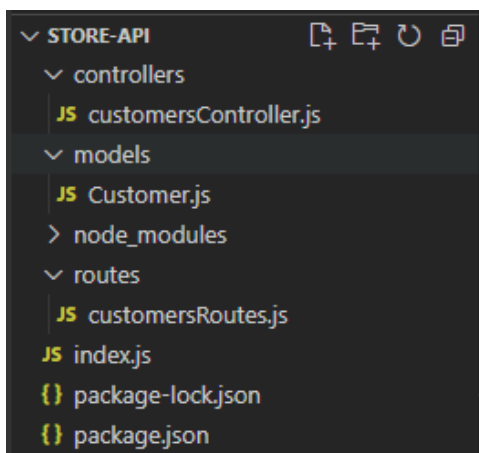
controllers\customersController.js

```
const Customer = require('../models/Customer');
```

```
// primeira ação: list
```

```
exports.index = async (req, res) => {  
  try {  
    const customers = await Customer.find({});  
    res.json(customers);  
  } catch (error) {  
    console.log(error);  
    res.send(error);  
    next();  
  }  
};
```

- Na pasta raiz do projeto, adicione uma pasta chamada "routes" e dentro dela adicione um arquivo chamado "customersRoutes.js".



routes\customersRoutes.js

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

```
const router = express.Router();
```

```
const customersController = require('../controllers/customersController');
```

```
module.exports = function() {  
  // get: /customers  
  router.get('/customers', customersController.index);  
  return router;  
}
```

index.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');
const customersRoutes = require('./routes/customersRoutes');

const app = express();

// Configurando mongoose
mongoose.Promise = global.Promise;
mongoose.connect('mongodb://localhost/store-api',
{
  useNewUrlParser: true
});

// Habilitar body-parser
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

// Habilitar cors
app.use(cors());

app.use('/', customersRoutes());

app.listen(5000, () => {
  console.log('Servidor web Express em execução!');
})
```

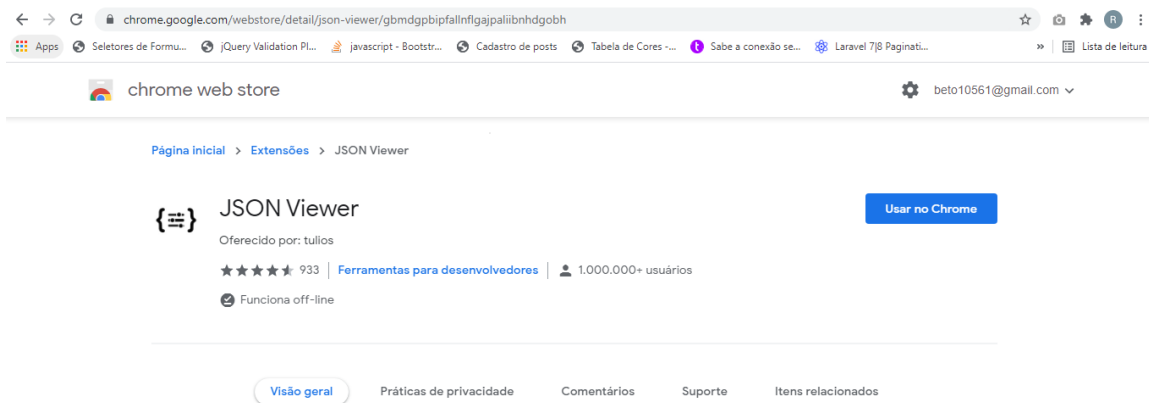
npm start

```
PS C:\felipemx\store-api> npm start

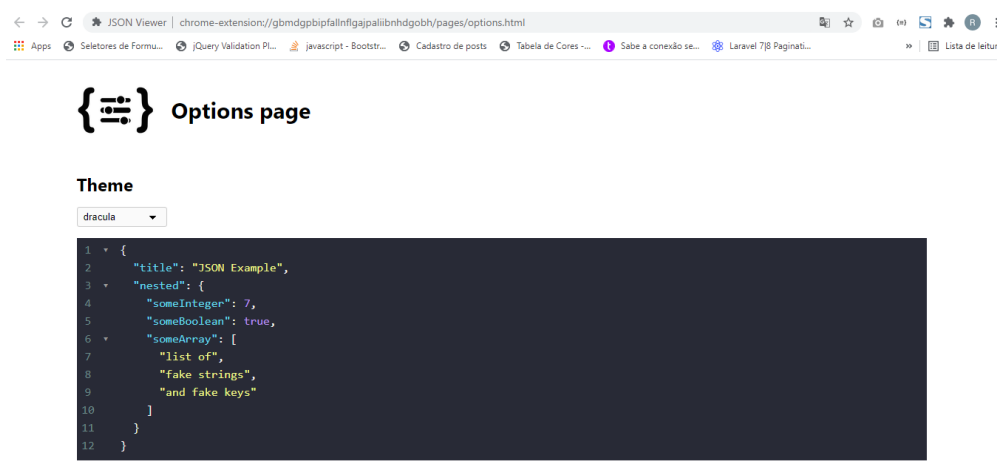
> store-api@1.0.0 start C:\felipemx\store-api
> nodemon index.js

[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
(node:7720) [MONGODB DRIVER] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
(Use `node --trace-warnings ...` to show where the warning was created)
Servidor web Express em execução!
(node:7720) DeprecationWarning: collection.ensureIndex is deprecated. Use createIndexes instead.
```

- Instale a extensão **JSON Viewer** no **Google Chrome**:

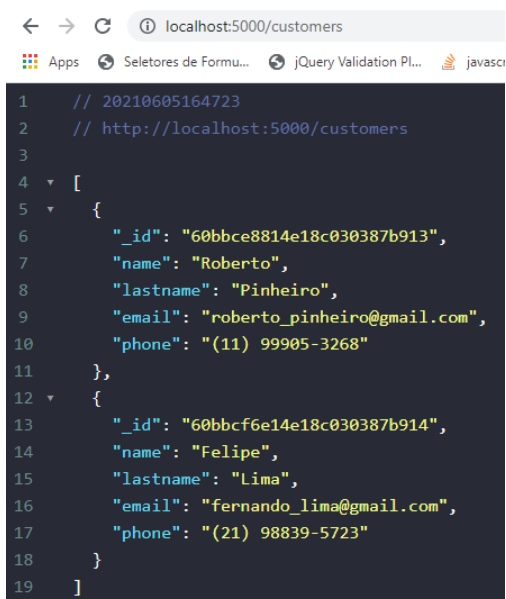


- Como tema, escolha: "dracula":



- No browser, entre com:

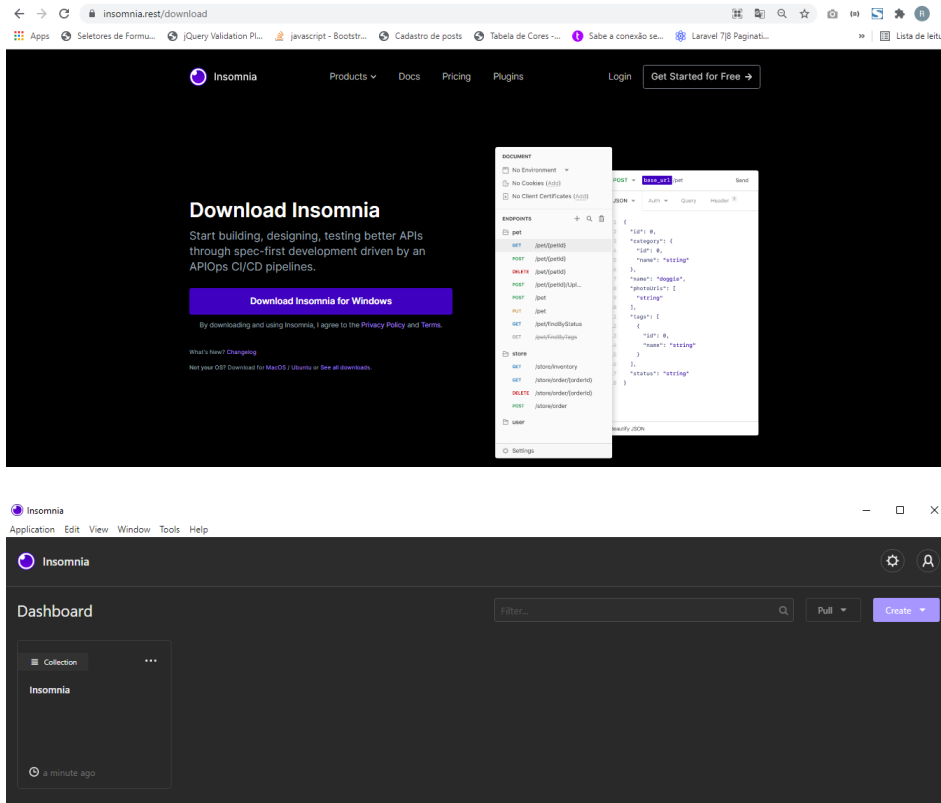
<http://localhost:5000/customers>



Aula 04 - Utilizando Postman para interagir com nossa API REST

- Faça o download do Insomnia e o instale no computador:

<https://insomnia.rest/download>



controllers\customersController.js

```
const Customer = require('../models/Customer');
```

```
exports.create = async (req, res) => {
  const customer = new Customer(req.body);

  try {
    await customer.save();
    res.status(201).json({
      message: "Cliente cadastrado com sucesso!"
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um cliente com o email: ${req.body.email}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
}
```

```
// listar clientes

exports.index = async (req, res) => {
  try {
    const customers = await Customer.find({});
    res.json(customers);
  } catch (error) {
    console.log(error);
    res.send(error);
    next();
  }
};
```

routes\customersRoutes.js.js

```
const express = require('express');
const router = express.Router();
const customersController = require('../controllers/customersController');

module.exports = function() {
  // post: /customers
  router.post('/customers', customersController.create);

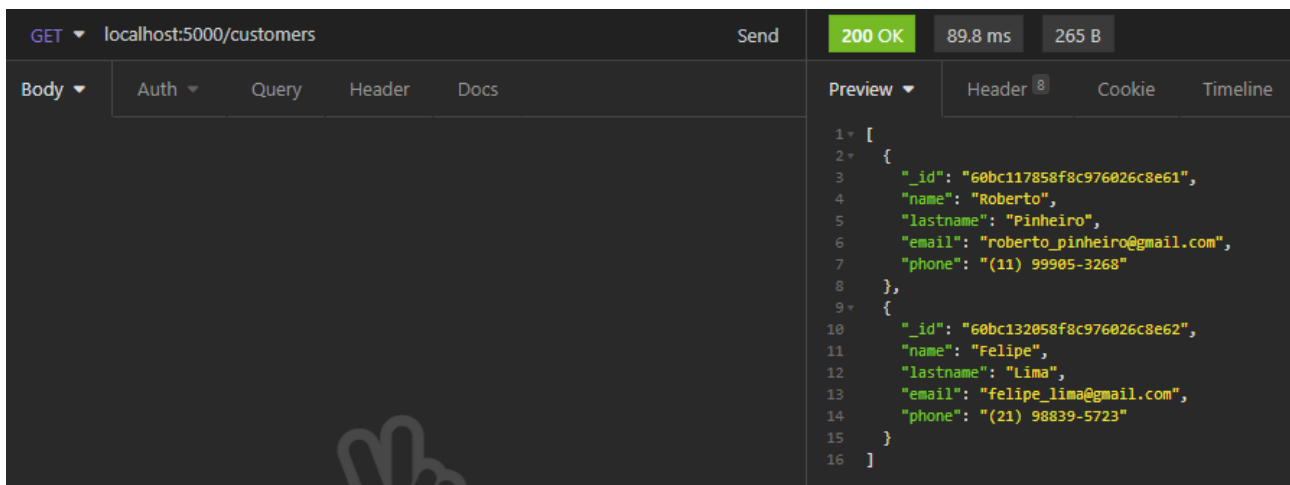
  // get: /customers
  router.get('/customers', customersController.index);

  return router;
}
```

- No Insomnia:

Listando clientes

GET - localhost:5000/customers



The screenshot shows a web browser interface with the address bar displaying 'localhost:5000/customers'. The status bar indicates a '200 OK' response with a time of '89.8 ms' and a size of '265 B'. The 'Body' tab is selected, showing a JSON array of two customer objects. The first object has an ID of '60bc117858f8c976026c8e61', name 'Roberto', lastname 'Pinheiro', email 'roberto_pinheiro@gmail.com', and phone '(11) 99905-3268'. The second object has an ID of '60bc132058f8c976026c8e62', name 'Felipe', lastname 'Lima', email 'felipe_lima@gmail.com', and phone '(21) 98839-5723'.

```
1+ [
2+ {
3+   "_id": "60bc117858f8c976026c8e61",
4+   "name": "Roberto",
5+   "lastname": "Pinheiro",
6+   "email": "roberto_pinheiro@gmail.com",
7+   "phone": "(11) 99905-3268"
8+ },
9+ {
10+  "_id": "60bc132058f8c976026c8e62",
11+  "name": "Felipe",
12+  "lastname": "Lima",
13+  "email": "felipe_lima@gmail.com",
14+  "phone": "(21) 98839-5723"
15+ }
16+ ]
```

Cadastrando cliente

POST - localhost:5000/customers


POST	localhost:5000/customers	Send	400 Bad Request	345 ms	75 B	Just Now		
JSON	Auth	Query	Header 1	Docs	Preview	Header 8	Cookie	Timeline
<pre>1 { 2 "name": "Roberto", 3 "lastname": "Pinheiro", 4 "email": "roberto_pinheiro@gmail.com", 5 "phone": "(11) 99905-3268" 6 }</pre>				<pre>1 { 2 "message": "Já existe um cliente com o email: roberto_pinheiro@gmail.com" 3 }</pre>				

POST - localhost:5000/customers

POST	localhost:5000/customers	Send	200 OK	458 ms	45 B			
JSON	Auth	Query	Header 1	Docs	Preview	Header 8	Cookie	Timeline
<pre>1 { 2 "name": "Juan", 3 "lastname": "Molina", 4 "email": "juan_molina@hotmail.com", 5 "phone": "(19) 99905-3268" 6 }</pre>				<pre>1 { 2 "message": "Cliente cadastrado com sucesso!" 3 }</pre>				

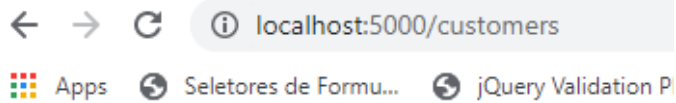
Listando clientes

GET - localhost:5000/customers

GET	localhost:5000/customers	Send	200 OK	12.8 ms	402 B			
Body	Auth	Query	Header	Docs	Preview	Header 8	Cookie	Timeline
 Select a body type from above				<pre>1 [2 { 3 "_id": "60bc117858f8c976026c8e61", 4 "name": "Roberto", 5 "lastname": "Pinheiro", 6 "email": "roberto_pinheiro@gmail.com", 7 "phone": "(11) 99905-3268" 8 }, 9 { 10 "_id": "60bc132058f8c976026c8e62", 11 "name": "Felipe", 12 "lastname": "Lima", 13 "email": "felipe_lima@gmail.com", 14 "phone": "(21) 98839-5723" 15 }, 16 { 17 "_id": "60bc1489fca2db1580e514d2", 18 "name": "Juan", 19 "lastname": "Molina", 20 "email": "juan_molina@hotmail.com", 21 "phone": "(19) 99905-3268", 22 "__v": 0 23 } 24]</pre>				

- No browser:

localhost:5000/customers



```
1 // 20210605212232
2 // http://localhost:5000/customers
3
4 [
5   {
6     "_id": "60bc117858f8c976026c8e61",
7     "name": "Roberto",
8     "lastname": "Pinheiro",
9     "email": "roberto_pinheiro@gmail.com",
10    "phone": "(11) 99905-3268"
11  },
12  {
13    "_id": "60bc132058f8c976026c8e62",
14    "name": "Felipe",
15    "lastname": "Lima",
16    "email": "felipe_lima@gmail.com",
17    "phone": "(21) 98839-5723"
18  },
19  {
20    "_id": "60bc1489fca2db1580e514d2",
21    "name": "Juan",
22    "lastname": "Molina",
23    "email": "juan_molina@hotmail.com",
24    "phone": "(19) 99905-3268",
25    "__v": 0
26  }
27 ]
```

Aula 05 - Consultar detalhes de um cliente por seu ID em nossa API REST

routes\customersRoutes.js

```
const express = require('express');
const router = express.Router();
const customersController = require('../controllers/customersController');

module.exports = function() {
  // post: /customers
  router.post('/customers', customersController.add);

  // get: /customers
  router.get('/customers', customersController.list);

  // get: /customers/:id
  router.get('/customers/:id', customersController.show);

  return router;
};
```

controllers\customersController.js

```
const Customer = require('../models/Customer');

// cadastrar cliente

exports.create = async (req, res) => {
  const customer = new Customer(req.body);

  try {
    await customer.save();
    res.status(201).json({
      message: "Cliente cadastrado com sucesso!"
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um cliente com o email: ${req.body.email}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
}

// listar clientes

exports.index = async (req, res) => {
  try {
    const customers = await Customer.find({});
    res.json(customers);
  } catch (error) {
    console.log(error);
  }
}
```

```

    res.send(error);
    next();
  }
};

// exibir detalhes de um cliente por id

exports.show = async (req, res, next) => {
  try {
    const customer = await Customer.findById(req.params.id);
    if(!customer){
      res.status(404).json({
        message: "O cliente não existe!"
      });
    }
    res.json(customer);
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

```

mongod

npm start

```

PS C:\felipemx\store-api> npm start

> store-api@1.0.0 start C:\felipemx\store-api
> nodemon index.js

[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
(node:4788) [MONGODB DRIVER] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
(Use `node --trace-warnings ...` to show where the warning was created)
Servidor web Express em execução!
(node:4788) DeprecationWarning: collection.ensureIndex is deprecated. Use createIndexes instead.

```

- No Insomnia:

localhost:5000/customers/60bc117858f8c976026c8e61

GET	localhost:5000/customers/60bc117858f8c976026c8e61	Send	200 OK	82.9 ms	136 B
Body	Auth	Query	Header	Docs	Preview
					<pre> 1+ { 2 "id": "60bc117858f8c976026c8e61", 3 "name": "Roberto", 4 "lastname": "Pinheiro", 5 "email": "roberto_pinheiro@gmail.com", 6 "phone": "(11) 99905-3268" 7 } </pre>

localhost:5000/customers/60bc117858f8c976026c8e60

GET

localhost:5000/customers/60bc117858f8c976026c8e60

Send

404 Not Found

9.39 ms

36 B

Body

Auth

Query

Header

Docs

Preview

Header

Cookie

Timeline

1

{

2

"message": "O cliente não existe!"

3

}

localhost:5000/customers/60bc1178

GET

localhost:5000/customers/60bc1178

Send

400 Bad Request

20.4 ms

47 B

Body

Auth

Query

Header

Docs

Preview

Header

Cookie

Timeline

1

{

2

"message": "Erro ao processar a requisição!"

3

}

Aula 06 - Atualizar um cliente em nossa API REST

routes\customersRoutes.js

```
const express = require('express');
const router = express.Router();
const customersController = require('../controllers/customersController');

module.exports = function() {
  // post: /customers
  router.post('/customers', customersController.create);

  // get: /customers
  router.get('/customers', customersController.index);

  // get: /customers/:id
  router.get('/customers/:id', customersController.show);

  // put: /customers/:id
  router.put('/customers/:id', customersController.update);

  return router;
};
```

controllers\customersController.js

```
const Customer = require('../models/Customer');

exports.create = async (req, res) => {
  const customer = new Customer(req.body);

  try {
    await customer.save();
    res.status(201).json({
      message: "Cliente cadastrado com sucesso!"
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um cliente com o email: ${req.body.email}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
}

// listar clientes
exports.index = async (req, res) => {
  try {
    const customers = await Customer.find({});
    res.json(customers);
  } catch (error) {
    console.log(error);
  }
}
```



```

    res.send(error);
    next();
  }
};

// exibir detalhes de um cliente por id
exports.show = async (req, res, next) => {
  try {
    const customer = await Customer.findById(req.params.id);
    if(!customer){
      res.status(404).json({
        message: "O cliente não existe!"
      });
    }
    res.json(customer);
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

```

// atualizar dados do cliente

```

exports.update = async (req, res, next) => {
  try {
    const customer = await Customer.findOneAndUpdate(
      { _id: req.params.id },
      req.body,
      { new: true }
    );

    res.json({
      message: 'Cliente atualizado com sucesso!'
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um cliente com o email: ${req.body.email}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
};

```

PUT localhost:5000/customers/60bc1489fca2db1580e514d2		Send	200 OK	696 ms	45 B
JSON	Auth	Query	Header 1	Docs	Preview
<pre> 1 { 2 "name": "Juan", 3 "lastname": "Molina López", 4 "email": "juan_molina@hotmail.com", 5 "phone": "(19) 99905-3268" 6 } </pre>			<pre> 1 { 2 "message": "Cliente atualizado com sucesso!" 3 } </pre>		

GET ▾

localhost:5000/customers

Send

200 OK

8.95 ms

409 B

Body ▾

Auth ▾

Query

Header


Docs

Preview ▾

Header 8

Cookie

Timeline



Select a body type from above

```
1  [
2  {
3    "_id": "60bc11785f8c976026c8e61",
4    "name": "Roberto",
5    "lastname": "Pinheiro",
6    "email": "roberto_pinheiro@gmail.com",
7    "phone": "(11) 99905-3268"
8  },
9  {
10   "_id": "60bc132058f8c976026c8e62",
11   "name": "Felipe",
12   "lastname": "Lima",
13   "email": "felipe_lima@gmail.com",
14   "phone": "(21) 98839-5723"
15  },
16  {
17   "_id": "60bc1489fca2db1580e514d2",
18   "name": "Juan",
19   "lastname": "Molina López",
20   "email": "juan_molina@hotmail.com",
21   "phone": "(19) 99905-3268",
22   "__v": 0
23  }
24 ]
```

Aula 07 - Eliminar um cliente em nossa API REST

routes\customersRoutes.js

```
const express = require('express');
const router = express.Router();
const customersController = require('../controllers/customersController');

module.exports = function() {
  // post: /customers
  router.post('/customers', customersController.create);

  // get: /customers
  router.get('/customers', customersController.index);

  // get: /customers/:id
  router.get('/customers/:id', customersController.show);

  // put: /customers/:id
  router.put('/customers/:id', customersController.update);

  // delete: /customers/:id
  router.delete('/customers/:id', customersController.delete);

  return router;
};
```

controllers\customersController.js

```
const Customer = require('../models/Customer');

exports.create = async (req, res) => {
  const customer = new Customer(req.body);

  try {
    await customer.save();
    res.status(201).json({
      message: "Cliente cadastrado com sucesso!"
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um cliente com o email: ${req.body.email}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
}

// listar clientes
exports.index = async (req, res) => {
  try {
    const customers = await Customer.find({});
  }
}
```

```

    res.json(customers);
  } catch (error) {
    console.log(error);
    res.send(error);
    next();
  }
};

// exibir detalhes de um cliente por id
exports.show = async (req, res, next) => {
  try {
    const customer = await Customer.findById(req.params.id);
    if(!customer){
      res.status(404).json({
        message: "O cliente não existe!"
      });
    }
    res.json(customer);
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

// atualizar dados do cliente
exports.update = async (req, res, next) => {
  try {
    const customer = await Customer.findOneAndUpdate(
      { _id: req.params.id },
      req.body,
      { new: true }
    );

    res.json({
      message: 'Cliente atualizado com sucesso!'
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um cliente com o email: ${req.body.email}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
};

// Excluir um cliente por id
exports.delete = async (req, res, next) => {
  try {
    await Customer.findOneAndDelete({
      _id: req.params.id
    });

    res.json({
      message: 'Cliente excluído com sucesso!'
    });
  }
};

```

```

});
} catch (error) {
  res.status(400).json({
    message: 'Erro ao processar a requisição!'
  });
}
};

```

- No Insomnia:

localhost:5000/customers/60bc1489fca2db1580e514d2

DELETE localhost:5000/customers/60bc1489fca2db1580e514d2		Send	200 OK	33.7 ms	43 B
Body	Auth	Query	Header	Docs	Preview
					Header 8 Cookie Timeline
					<pre> 1 { 2 "message": "Cliente excluido com sucesso!" 3 } </pre>

localhost:5000/customers

GET localhost:5000/customers		Send	200 OK	32 ms	265 B
Body	Auth	Query	Header	Docs	Preview
					Header 8 Cookie Timeline
					<pre> 1 [2 { 3 "_id": "60bc117858f8c976026c8e61", 4 "name": "Roberto", 5 "lastname": "Pinheiro", 6 "email": "roberto_pinheiro@gmail.com", 7 "phone": "(11) 99905-3268" 8 }, 9 { 10 "_id": "60bc132058f8c976026c8e62", 11 "name": "Felipe", 12 "lastname": "Lima", 13 "email": "felipe_lima@gmail.com", 14 "phone": "(21) 98839-5723" 15 } 16] </pre>

localhost:5000/customers/60bc1489fca2db1580e514d2

GET localhost:5000/customers/60bc1489fca2db1580e514d2		Send	404 Not Found	20.5 ms	36 B
Body	Auth	Query	Header	Docs	Preview
					Header 8 Cookie Timeline
					<pre> 1 { 2 "message": "O cliente não existe!" 3 } </pre>

Aula 08 - CRUD de Produtos

models\Product.js

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

const Schema = mongoose.Schema;

const productsSchema = new Schema({
  sku: {
    type: String,
    trim: true,
    unique: true,
    uppercase: true
  },
  name: {
    type: String,
    trim: true
  },
  description: {
    type: String,
    trim: true
  },
  price: {
    type: Number
  },
  stock: {
    type: Number,
    default: 0
  },
  available: {
    type: Boolean,
    default: true
  }
});

module.exports = mongoose.model('Product', productsSchema);
```

OBS.: Cada produto do estoque tem um SKU (Stock Keeping Unit) particular, que é um código formado por letras e números. Ao gerenciar o cadastro e retirada de produtos no seu estoque, ao invés de buscar pelo nome de cada produto, você procura pelo SKU.

controllers\productsController.js

```
const Product = require('../models/Product');

// cadastrar produto

exports.create = async (req, res) => {
  const product = new Product(req.body);

  try {
    await product.save();
    res.status(201).json({
      message: "Produto cadastrado com sucesso!"
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um produto com o sku: ${req.body.sku}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
};

// listar produtos

exports.index = async (req, res) => {
  try {
    const products = await Product.find({});
    res.json(products);
  } catch (error) {
    console.log(error);
    res.send(error);
    next();
  }
};

// exibir detalhes de um produto por id

exports.show = async (req, res, next) => {
  try {
    const product = await Product.findById(req.params.id);
    if(!product){
      res.status(404).json({
        message: "O produto não existe!"
      });
    }
    res.json(product);
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};
```

```
// atualizar dados de um produto
```

```
exports.update = async (req, res, next) => {
  try {
    const product = await Product.findOneAndUpdate(
      { _id: req.params.id },
      req.body,
      { new: true }
    );

    res.json({
      message: 'Produto atualizado com sucesso!'
    });
  } catch (error) {
    if (error.code === 11000) {
      res.status(400).json({
        message: `Já existe um produto com o sku: ${req.body.sku}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
};
```

```
// Excluir um produto por id
```

```
exports.delete = async (req, res, next) => {
  try {
    await Product.findOneAndDelete({
      _id: req.params.id
    });

    res.json({
      message: 'Produto excluído com sucesso!'
    });
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};
```


routes\productsRoutes.js

```
const express = require('express');
const router = express.Router();
const productsController = require('../controllers/productsController');

module.exports = function() {
  // post: /products
  router.post('/products', productsController.create);

  // get: /products
  router.get('/products', productsController.index);

  // get: /products/:id
  router.get('/products/:id', productsController.show);

  // put: /products/:id
  router.put('/products/:id', productsController.update);

  // delete: products/:id
  router.delete('/products/:id', productsController.delete);

  return router;
};
```

index.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

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

const app = express();

// Configurando mongoose
mongoose.Promise = global.Promise;
mongoose.connect('mongodb://localhost/store-api',
{
  useNewUrlParser: true
});

// Habilitar body-parser
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

// Habilitar cors
app.use(cors());

app.use('/', customersRoutes());
app.use('/', productsRoutes());

app.listen(5000, () => {
  console.log('Servidor web Express em execução!');
});
```

Cadastrando produtos

- No Insomnia:

POST - localhost:5000/products

```
{
  "sku": "A11",
  "name": "Ventilador",
  "description": "Ventilador de 3 velocidades",
  "price": 149.90,
  "stock": 25,
  "available": true
}
```

POST localhost:5000/products/	Send	201 Created	65.6 ms	45 B
JSON	Auth	Query	Header 1	Docs
<pre>1 { 2 "sku": "A11", 3 "name": "Ventilador", 4 "description": "Ventilador de 3 velocidades", 5 "price": 149.90, 6 "stock": 25, 7 "available": true 8 }</pre>				
Preview				
<pre>1 { 2 "message": "Produto cadastrado com sucesso!" 3 }</pre>				

POST - localhost:5000/products

```
{
  "sku": "B12",
  "name": "Liquidificador",
  "description": "Liquidificador com 5 velocidades",
  "price": 99.90,
  "stock": 45,
  "available": true
}
```

POST localhost:5000/products/	Send	201 Created	24.6 ms	45 B
JSON	Auth	Query	Header 1	Docs
<pre>1 { 2 "sku": "B12", 3 "name": "Liquidificador", 4 "description": "Liquidificador com 5 velocidades", 5 "price": 99.90, 6 "stock": 45, 7 "available": true 8 }</pre>				
Preview				
<pre>1 { 2 "message": "Produto cadastrado com sucesso!" 3 }</pre>				

POST localhost:5000/products/	Send	400 Bad Request	12.4 ms	50 B
JSON	Auth	Query	Header 1	Docs
<pre>1 { 2 "sku": "B12", 3 "name": "Liquidificador", 4 "description": "Liquidificador com 5 velocidades", 5 "price": 99.90, 6 "stock": 45, 7 "available": true 8 }</pre>				
Preview				
<pre>1 { 2 "message": "Já existe um produto com o sku: B12" 3 }</pre>				


POST - localhost:5000/products

```
{
  "sku": "E57",
  "name": "Rádio relógio",
  "description": "Rádio relógio digital",
  "price": 67.30,
  "stock": 10,
  "available": true
}
```

POST	localhost:5000/products/	Send	201 Created	86.8 ms	45 B	
JSON	Auth	Query	Header 1	Docs		
<pre>1 { 2 "sku": "E57", 3 "name": "Rádio relógio", 4 "description": "Rádio relógio digital", 5 "price": 67.30, 6 "stock": 10, 7 "available": true 8 }</pre>			Preview	Header 8	Cookie	Timeline
			<pre>1 { 2 "message": "Produto cadastrado com sucesso!" 3 }</pre>			

Listando os produtos

GET - localhost:5000/products

GET	localhost:5000/products	Send	200 OK	24.2 ms	505 B	
Body	Auth	Query	Header	Docs		
 <p>Select a body type from above</p>			Preview	Header 8	Cookie	Timeline
			<pre>1 [2 { 3 "stock": 25, 4 "available": true, 5 "_id": "60bc4cd3a512b00c14451ef6", 6 "sku": "A11", 7 "name": "Ventilador", 8 "description": "Ventilador de 3 velocidades com pedestal", 9 "price": 149.9, 10 "__v": 0 11 }, 12 { 13 "stock": 45, 14 "available": true, 15 "_id": "60bc4d9ea512b00c14451ef7", 16 "sku": "B12", 17 "name": "Liquidificador", 18 "description": "Liquidificador com 5 velocidades", 19 "price": 99.9, 20 "__v": 0 21 }, 22 { 23 "stock": 10, 24 "available": true, 25 "_id": "60bcc1f793f6110b50dd6d31", 26 "sku": "E57", 27 "name": "Rádio relógio", 28 "description": "Rádio relógio digital", 29 "price": 67.3, 30 "__v": 0 31 } 32]</pre>			

Atualizando produtos

PUT - localhost:5000/products/60bc4cd3a512b00c14451ef6

PUT	localhost:5000/products/60bc4cd3a512b00c14451ef6	Send	200 OK	1.27 s	45 B			
JSON	Auth	Query	Header 1	Docs	Preview	Header 8	Cookie	Timeline
<pre>1 { 2 "sku": "A11", 3 "name": "Ventilador", 4 "description": "Ventilador de 3 velocidades com pedestal", 5 "price": 149.90, 6 "stock": 25, 7 "available": true 8 }</pre>				<pre>1 { 2 "message": "Produto atualizado com sucesso!" 3 }</pre>				

GET - localhost:5000/products/60bc4cd3a512b00c14451ef6

GET	localhost:5000/products/60bc4cd3a512b00c14451ef6	Send	200 OK	218 ms	173 B			
Body	Auth	Query	Header	Docs	Preview	Header 8	Cookie	Timeline
				<pre>1 { 2 "stock": 25, 3 "available": true, 4 "_id": "60bc4cd3a512b00c14451ef6", 5 "sku": "A11", 6 "name": "Ventilador", 7 "description": "Ventilador de 3 velocidades com pedestal", 8 "price": 149.9, 9 "__v": 0 10 }</pre>				

Excluindo produto

DELETE - localhost:5000/products/60bcc1f793f6110b50dd6d31

DELETE	localhost:5000/products/60bcc1f793f6110b50dd6d31	Send	200 OK	32.6 ms	43 B			
Body	Auth	Query	Header	Docs	Preview	Header 8	Cookie	Timeline
				<pre>1 { 2 "message": "Produto excluido com sucesso!" 3 }</pre>				

Listando produtos

GET - localhost:5000/products

GET ▾

localhost:5000/products

Send

200 OK

15.1 ms

344 B

Body ▾

Auth ▾

Query

Header


Docs

Preview ▾

Header 8

Cookie

Timeline



Select a body type from above

```
1+ [
2+ {
3   "stock": 25,
4   "available": true,
5   "_id": "60bc4cd3a512b00c14451ef6",
6   "sku": "A11",
7   "name": "Ventilador",
8   "description": "Ventilador de 3 velocidades com pedestal",
9   "price": 149.9,
10  "__v": 0
11 },
12+ {
13   "stock": 45,
14   "available": true,
15   "_id": "60bc4d9ea512b00c14451ef7",
16   "sku": "B12",
17   "name": "Liquidificador",
18   "description": "Liquidificador com 5 velocidades",
19   "price": 99.9,
20   "__v": 0
21 }
22 ]
```

Aula 09 - Subir imagem de produto

Instalando multer e shortid

`npm install --save multer shortid`

multer será utilizado para processar e armazenar o artigo que recebemos na requisição e shortid utilizaremos para gerar id aleatório

```
PS C:\felipemx\store-api> npm install --save multer shortid
npm WARN store-api@1.0.0 No repository field.

+ shortid@2.2.16
+ multer@1.4.2
added 19 packages from 13 contributors and audited 100 packages in 15.245s

2 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

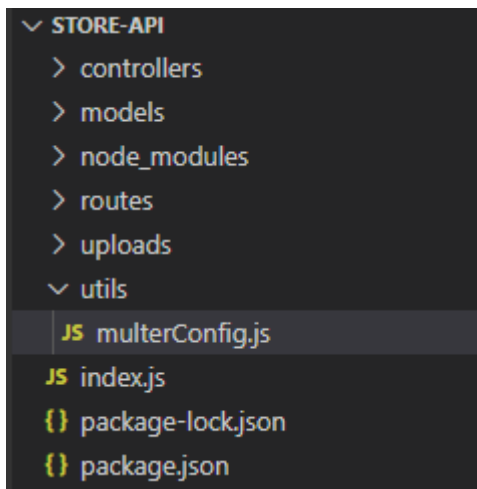
models\Product.js

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;
```

```
const productsSchema = new Schema({
  sku: {
    type: String,
    trim: true,
    unique: true,
    uppercase: true
  },
  name: {
    type: String,
    trim: true
  },
  description: {
    type: String,
    trim: true
  },
  image: {
    type: String
  },
  price: {
    type: Number
  },
  stock: {
    type: Number,
    default: 0
  },
  available: {
    type: Boolean,
    default: true
  }
});
```

```
module.exports = mongoose.model('Product', productsSchema);
```

Na pasta raiz do projeto adicione duas pastas: **uploads** e **utils** e dentro desta última pasta adicione um arquivo chamado **"multerConfig.js"**:



utils\multerConfig.js

```
// https://github.com/afelipelc/custom-snippets/blob/master/multerConfig.js
```

```
const multer = require('multer');
const shortid = require('shortid');

const multerConfig = {
  storage: fileStorage = multer.diskStorage({
    destination: (req, file, cb) => {
      cb(null, __dirname+'../../uploads/'); // los uploads se subirán en esta carpeta
    },
    filename: (req, file, cb) => {
      // obtener la extensión del archivo
      const extension = file.mimetype.split('/')[1];
      // generar ID para ponerlo como nombre de imagen
      cb(null, `${shortid.generate()}.${extension}`);
    }
  }),
  fileFilter(req, file, cb) {
    if ( file.mimetype === 'image/jpeg' || file.mimetype === 'image/png' ) { // solo aceptar imágenes
      cb(null, true);
    } else {
      cb(new Error('Formato de imagen inválido!'))
    }
  },
}

module.exports = multerConfig;
```

models\Product.js

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

const Schema = mongoose.Schema;

const productsSchema = new Schema({
  sku: {
    type: String,
    trim: true,
    unique: true,
    uppercase: true
  },
  name: {
    type: String,
    trim: true
  },
  description: {
    type: String,
    trim: true
  },
  image: {
    type: String
  },
  price: {
    type: Number
  },
  stock: {
    type: Number,
    default: 0
  },
  available: {
    type: Boolean,
    default: true
  }
});

module.exports = mongoose.model('Product', productsSchema);
```


controllers\productsController.js

```
const multer = require('multer');
const multerConfig = require('../utils/multerConfig');

const Product = require('../models/Product');

const upload = multer(multerConfig).single('image');

exports.fileUpload = (req, res, next) => {
  upload(req, res, function(error) {
    if(error){
      res.json({message: error});
    }
    return next();
  });
};

// cadastrar produto

exports.create = async (req, res) => {
  const product = new Product(req.body);

  try {
    if (req.file && req.file.filename) {
      product.image = req.file.filename;
    }
    await product.save();
    res.status(201).json({
      message: "Produto cadastrado com sucesso!"
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um produto com o sku: ${req.body.sku}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
};

// listar produtos

exports.index = async (req, res) => {
  try {
    const products = await Product.find({});
    res.json(products);
  } catch (error) {
    console.log(error);
    res.send(error);
    next();
  }
};
```

```
// exibir detalhes de um produto por id
```

```
exports.show = async (req, res, next) => {  
  try {  
    const product = await Product.findById(req.params.id);  
    if(!product){  
      res.status(404).json({  
        message: "O produto não existe!"  
      });  
    }  
    res.json(product);  
  } catch (error) {  
    res.status(400).json({  
      message: 'Erro ao processar a requisição!'  
    });  
  }  
};
```

```
// atualizar dados de um produto
```

```
exports.update = async (req, res, next) => {  
  try {  
    let newProduct = req.body;  
  
    if (req.file && req.file.filename) {  
      newProduct.image = req.file.filename;  
    } else {  
      const product = await Product.findById(req.params.id);  
      newProduct.image = product.image;  
    }  
  
    const productUpdated = await Product.findOneAndUpdate(  
      { _id: req.params.id },  
      newProduct,  
      { new: true }  
    );  
  
    res.json({  
      message: 'Produto atualizado com sucesso!'  
    });  
  } catch (error) {  
    if(error.code === 11000){  
      res.status(400).json({  
        message: `Já existe um produto com o sku: ${req.body.sku}`  
      });  
    } else {  
      res.status(400).json({  
        message: "Erro ao processar a requisição!"  
      });  
    }  
  }  
};
```

```
// Excluir um produto por id
```

```
exports.delete = async (req, res, next) => {  
  try {  
    await Product.findOneAndDelete({  
      _id: req.params.id
```

```

});

res.json({
  message: 'Produto excluído com sucesso!'
});
} catch (error) {
  res.status(400).json({
    message: 'Erro ao processar a requisição!'
  });
}
};

```

routes\productsRoutes.js

```

const express = require('express');
const router = express.Router();
const productsController = require('../controllers/productsController');

module.exports = function() {
  // post: /products
  router.post('/products', productsController.fileUpload, productsController.create);

  // get: /products
  router.get('/products', productsController.index);

  // get: /products/:id
  router.get('/products/:id', productsController.show);

  // put: /products/:id
  router.put('/products/:id', productsController.fileUpload, productsController.update);

  // delete: products/:id
  router.delete('/products/:id', productsController.delete);

  return router;
};

```

index.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

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

const app = express();

// Configurando mongoose
mongoose.Promise = global.Promise;
mongoose.connect('mongodb://localhost/store-api',
{
  useNewUrlParser: true
});

// Habilitar body-parser
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

// Habilitar cors
app.use(cors());

app.use('/', customersRoutes());
app.use('/', productsRoutes());
app.use(express.static('uploads'));

app.listen(5000, () => {
  console.log('Servidor web Express em execução!');
})
```

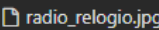
Cadastrando um novo produto

POST - localhost:5000/products/

POST localhost:5000/products/ Send

201 Created 686 ms 45 B

Multipart 7 Auth Query Header 1 Docs

sku	E67	✓	✕
name	Rádio relógio	✓	✕
description	Rádio relógio digital	✓	✕
image		✓	✕
price	78.50	✓	✕
stock	6	✓	✕
available	true	✓	✕

Preview Header 8 Cookie Timeline

```
1 {
2   "message": "Produto cadastrado com sucesso!"
3 }
```


Listando produtos

GET - localhost:5000/products

GET localhost:5000/products Send

200 OK 192 ms 529 B

Body Auth Query Header Docs



Select a body type from above

Preview Header 8 Cookie Timeline

```
1 [
2 {
3   "stock": 25,
4   "available": true,
5   "_id": "60bc4cd3a512b00c14451ef6",
6   "sku": "A11",
7   "name": "Ventilador",
8   "description": "Ventilador de 3 velocidades com pedestal",
9   "price": 149.9,
10  "__v": 0
11 },
12 {
13   "stock": 45,
14   "available": true,
15   "_id": "60bc4d9ea512b00c14451ef7",
16   "sku": "B12",
17   "name": "Liquidificador",
18   "description": "Liquidificador com 5 velocidades",
19   "price": 99.9,
20   "__v": 0
21 },
22 {
23   "stock": 6,
24   "available": true,
25   "_id": "60bcf97c1eb9f1063c4b429d",
26   "sku": "E67",
27   "name": "Rádio relógio",
28   "description": "Rádio relógio digital",
29   "price": 78.5,
30   "image": "-X04HB4pt.jpeg",
31   "__v": 0
32 }
33 ]
```

- No browser:

<http://localhost:5000/products>

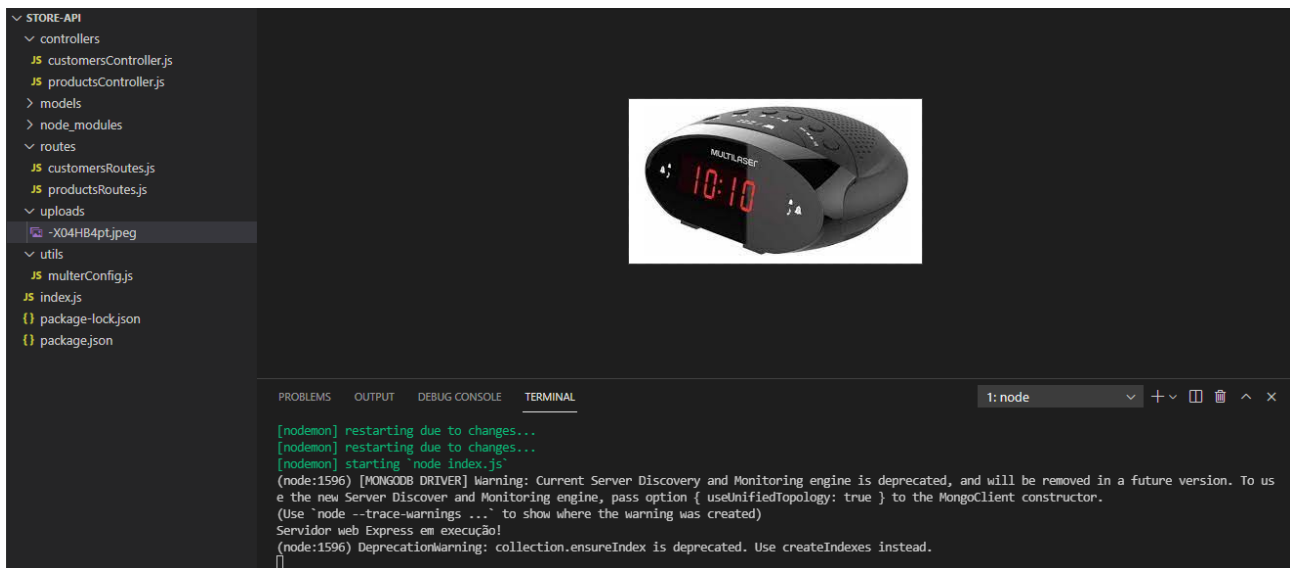
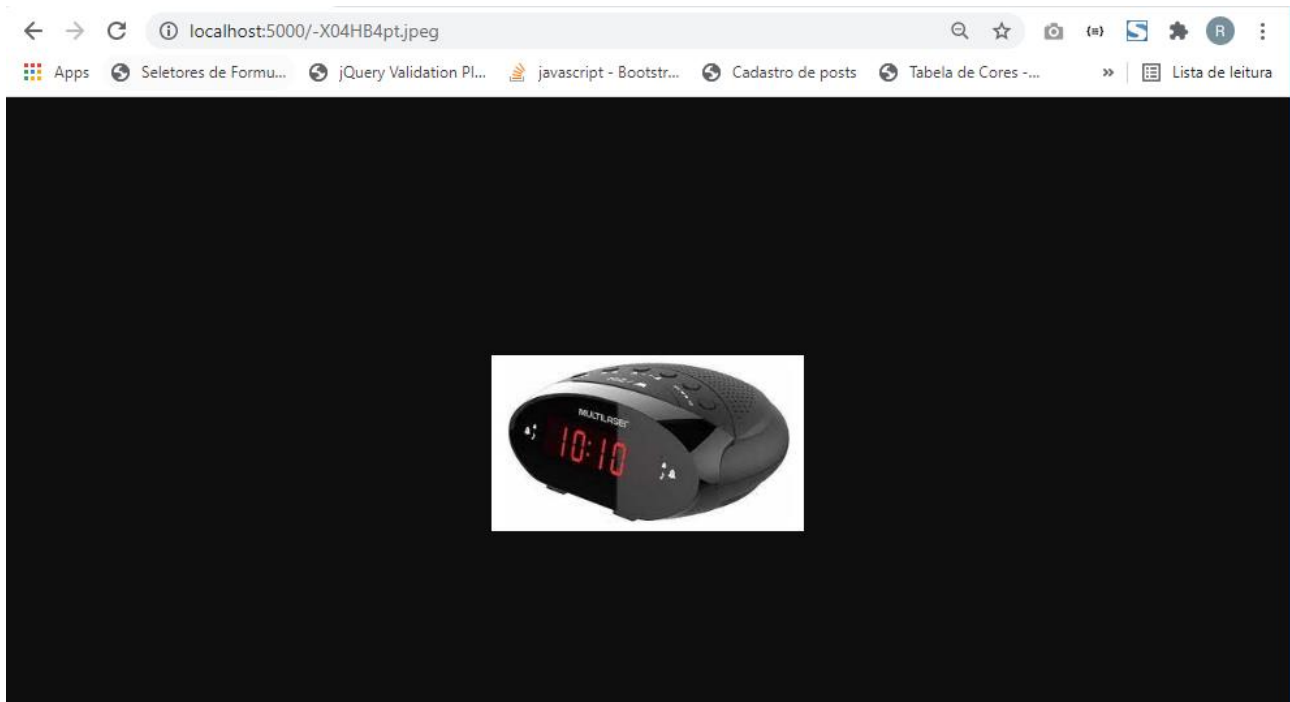


The screenshot shows a web browser window with the address bar displaying `localhost:5000/products`. Below the address bar, there are several tabs: "Apps", "Seletores de Formu...", "jQuery Validation Pl...", and "jav". The main content area of the browser displays a JSON array of three product objects. The JSON is formatted with syntax highlighting and line numbers on the left side, ranging from 1 to 36. The products are: a fan (Ventilador), a liquidifier (Liquidificador), and a digital radio clock (Rádio relógio digital).

```
1 // 20210606140222
2 // http://localhost:5000/products
3
4 [
5   {
6     "stock": 25,
7     "available": true,
8     "_id": "60bc4cd3a512b00c14451ef6",
9     "sku": "A11",
10    "name": "Ventilador",
11    "description": "Ventilador de 3 velocidades com pedestal",
12    "price": 149.9,
13    "__v": 0
14  },
15  {
16    "stock": 45,
17    "available": true,
18    "_id": "60bc4d9ea512b00c14451ef7",
19    "sku": "B12",
20    "name": "Liquidificador",
21    "description": "Liquidificador com 5 velocidades",
22    "price": 99.9,
23    "__v": 0
24  },
25  {
26    "stock": 6,
27    "available": true,
28    "_id": "60bcf97c1eb9f1063c4b429d",
29    "sku": "E67",
30    "name": "Rádio relógio",
31    "description": "Rádio relógio digital",
32    "price": 78.5,
33    "image": "-X04HB4pt.jpeg",
34    "__v": 0
35  }
36 ]
```

Imagem do produto

<http://localhost:5000/-X04HB4pt.jpeg>



Atualizando produtos

PUT - localhost:5000/products/60bc4cd3a512b00c14451ef6

PUTlocalhost:5000/products/60bc4cd3a512b00c14451ef6Send

Multipart7AuthQueryHeader1Docs

skuA11

nameVentilador

descriptionVentilador de 3 velocidades

image

ventilador.jpg

price149.90

stock25

availabletrue

200 OK1.07 s45 B

PreviewHeader8CookieTimeline

1 {

2 "message": "Produto atualizado com sucesso!"

3 }

PUT - localhost:5000/products/60bc4d9ea512b00c14451ef7

PUTlocalhost:5000/products/60bc4d9ea512b00c14451ef7Send

Multipart7AuthQueryHeader1Docs

skuB12

nameLiquidificador

descriptionLiquidificador com 5 velocidades

image

ventilador.jpg

price99.90

stock45

availabletrue

200 OK43.9 ms45 B

PreviewHeader8CookieTimeline

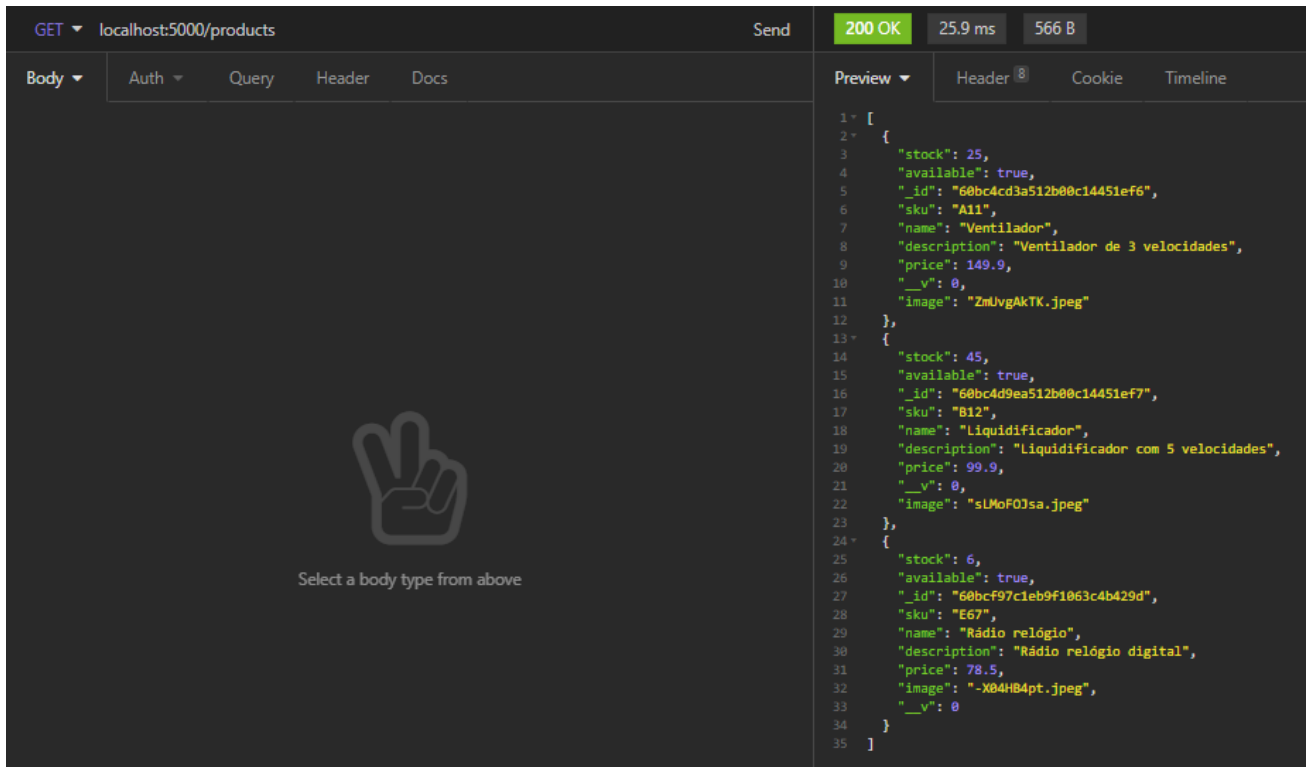
1 {

2 "message": "Produto atualizado com sucesso!"

3 }

Listando produtos

GET - localhost:5000/products



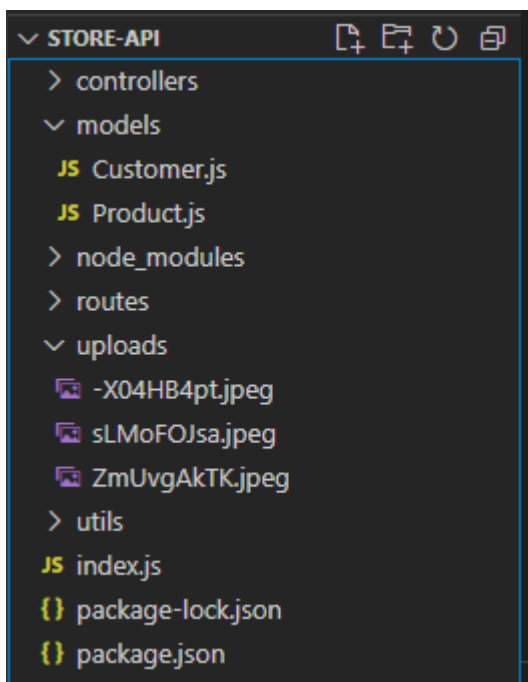
GET localhost:5000/products Send 200 OK 25.9 ms 566 B

Body Auth Query Header Docs

Preview Header 8 Cookie Timeline

Select a body type from above

```
1- [
2- {
3-   "stock": 25,
4-   "available": true,
5-   "_id": "60bc4cd3a512b00c14451ef6",
6-   "sku": "A11",
7-   "name": "Ventilador",
8-   "description": "Ventilador de 3 velocidades",
9-   "price": 149.9,
10-   "__v": 0,
11-   "image": "ZmUvgAkTK.jpeg"
12- },
13- {
14-   "stock": 45,
15-   "available": true,
16-   "_id": "60bc4d9ea512b00c14451ef7",
17-   "sku": "B12",
18-   "name": "Liquidificador",
19-   "description": "Liquidificador com 5 velocidades",
20-   "price": 99.9,
21-   "__v": 0,
22-   "image": "sLMoFOJsa.jpeg"
23- },
24- {
25-   "stock": 6,
26-   "available": true,
27-   "_id": "60bcf97c1eb9f1063c4b429d",
28-   "sku": "E67",
29-   "name": "Rádio relógio",
30-   "description": "Rádio relógio digital",
31-   "price": 78.5,
32-   "image": "-X04HB4pt.jpeg",
33-   "__v": 0
34- }
35- ]
```



STORE-API

- > controllers
- ▼ models
 - JS Customer.js
 - JS Product.js
- > node_modules
- > routes
- ▼ uploads
 - X04HB4pt.jpeg
 - sLMoFOJsa.jpeg
 - ZmUvgAkTK.jpeg
- > utils
- JS index.js
- { } package-lock.json
- { } package.json

- No browser:

<http://localhost:5000/products>

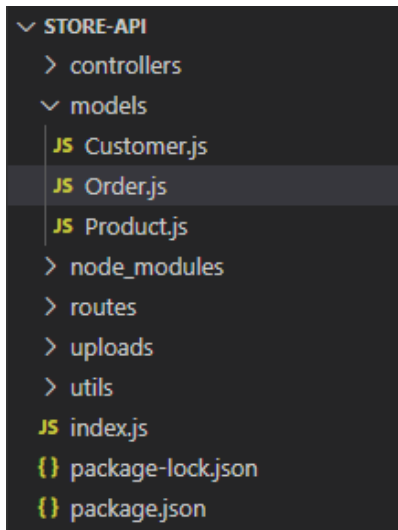


The screenshot shows a web browser window with the address bar displaying `localhost:5000/products`. Below the address bar, there are tabs for 'Apps', 'Seletores de Formu...', and 'jQuery Validation Pl...'. The main content area displays a REST client interface with a JSON array of three product objects. The JSON is formatted with syntax highlighting and line numbers from 1 to 38. The products are: a fan ('Ventilador'), a liquidifier ('Liquidificador'), and a digital radio ('Rádio relógio').

```
1 // 20210606141755
2 // http://localhost:5000/products
3
4 [
5   {
6     "stock": 25,
7     "available": true,
8     "_id": "60bc4cd3a512b00c14451ef6",
9     "sku": "A11",
10    "name": "Ventilador",
11    "description": "Ventilador de 3 velocidades",
12    "price": 149.9,
13    "__v": 0,
14    "image": "ZmUvgAkTK.jpeg"
15  },
16  {
17    "stock": 45,
18    "available": true,
19    "_id": "60bc4d9ea512b00c14451ef7",
20    "sku": "B12",
21    "name": "Liquidificador",
22    "description": "Liquidificador com 5 velocidades",
23    "price": 99.9,
24    "__v": 0,
25    "image": "sLMoFOJsa.jpeg"
26  },
27  {
28    "stock": 6,
29    "available": true,
30    "_id": "60bcf97c1eb9f1063c4b429d",
31    "sku": "E67",
32    "name": "Rádio relógio",
33    "description": "Rádio relógio digital",
34    "price": 78.5,
35    "image": "-X04HB4pt.jpeg",
36    "__v": 0
37  }
38 ]
```

Aula 10 - Registrar pedido e mostrar a listagem de pedidos

- Dentro da pasta "**models**" adicione um arquivo chamado "**Order.js**":



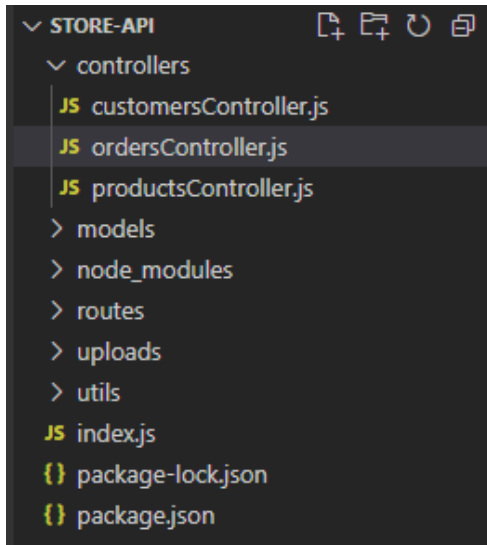
models\Order.js

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;

const ordersSchema = new Schema({
  created: {
    type: Date,
    default: Date.now
  },
  customer: {
    type: Schema.ObjectId,
    ref: 'Customer'
  },
  products: [{
    product: {
      type: Schema.ObjectId,
      ref: 'Product'
    },
    unitPrice: {
      type: Number
    },
    quantity: {
      type: Number
    },
    amount: {
      type: Number
    }
  }],
  totalAmount: {
    type: Number
  }
});

module.exports = mongoose.model('Order', ordersSchema);
```

- Dentro da pasta "**controllers**" adicione um arquivo chamado "**ordersController.js**":



controllers\ordersController.js

```
const Order = require('../models/Order');

// cadastrar pedido

exports.create = async (req, res, next) => {
  try {
    const order = new Order(req.body);
    await order.save();
    res.json(order);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};

// listar pedidos

exports.index = async (req, res, next) => {
  try {
    const orders = await Order.find({})
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });
    res.json(orders);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};
```

routes\ordersRoutes.js

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

const router = express.Router();

const ordersController = require('../controllers/ordersController');

module.exports = function() {
  // post: /orders
  router.post('/orders', ordersController.create);

  // get: /orders
  router.get('/orders', ordersController.index);

  return router;
};
```

index.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

const customersRoutes = require('./routes/customersRoutes');
const productsRoutes = require('./routes/productsRoutes');
const ordersRoutes = require('./routes/ordersRoutes');

const app = express();

// Configurando mongoose
mongoose.Promise = global.Promise;
mongoose.connect('mongodb://localhost/store-api',
{
  useNewUrlParser: true
});

// Habilitar body-parser
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

// Habilitar cors
app.use(cors());

app.use('/', customersRoutes());
app.use('/', productsRoutes());
app.use('/', ordersRoutes());
app.use(express.static('uploads'));

app.listen(5000, () => {
  console.log('Servidor web Express em execução!');
});
```

Cadastrando um pedido

POST - localhost:5000/orders

POSTlocalhost:5000/ordersSend200 OK298 ms385 B

JSONAuthQueryHeader1Docs

```
1 {
2   "customer": "60bc117858f8c976026c8e61",
3   "products": [
4     {
5       "product": "60bc4cd3a512b00c14451ef6",
6       "unitPrice": 149.90,
7       "quantity": 1,
8       "amount": 149.90
9     },
10    {
11      "product": "60bc4d9ea512b00c14451ef7",
12      "unitPrice": 99.90,
13      "quantity": 2,
14      "amount": 199.80
15    }
16  ],
17  "totalAmount": 349.70
18 }
```

PreviewHeader8CookieTimeline


```
1 {
2   "_id": "60bd22b64aac221b18baf9eb",
3   "customer": "60bc117858f8c976026c8e61",
4   "products": [
5     {
6       "_id": "60bd22b64aac221b18baf9ec",
7       "product": "60bc4cd3a512b00c14451ef6",
8       "unitPrice": 149.9,
9       "quantity": 1,
10      "amount": 149.9
11    },
12    {
13      "_id": "60bd22b64aac221b18baf9ed",
14      "product": "60bc4d9ea512b00c14451ef7",
15      "unitPrice": 99.9,
16      "quantity": 2,
17      "amount": 199.8
18    }
19  ],
20  "totalAmount": 349.7,
21  "created": "2021-06-06T19:32:06.132Z",
22  "__v": 0
23 }
```

Listando pedidos

GET - localhost:5000/orders

GETlocalhost:5000/ordersSend200 OK61.3 ms823 B

BodyAuthQueryHeaderDocs



Select a body type from above

PreviewHeader8CookieTimeline

```
1 [
2   {
3     "_id": "60bd22b64aac221b18baf9eb",
4     "customer": {
5       "_id": "60bc117858f8c976026c8e61",
6       "name": "Roberto",
7       "lastname": "Pinheiro",
8       "email": "roberto.pinheiro@gmail.com",
9       "phone": "(11) 99905-3268"
10    },
11    "products": [
12      {
13        "_id": "60bd22b64aac221b18baf9ec",
14        "product": {
15          "stock": 25,
16          "available": true,
17          "_id": "60bc4cd3a512b00c14451ef6",
18          "sku": "A11",
19          "name": "Ventilador",
20          "description": "Ventilador de 3 velocidades",
21          "price": 149.9,
22          "__v": 0,
23          "image": "ZmUvgAkTK.jpeg"
24        },
25        "unitPrice": 149.9,
26        "quantity": 1,
27        "amount": 149.9
28      },
29      {
30        "_id": "60bd22b64aac221b18baf9ed",
31        "product": {
32          "stock": 45,
33          "available": true,
34          "_id": "60bc4d9ea512b00c14451ef7",
35          "sku": "B12",
36          "name": "Liquidificador",
37          "description": "Liquidificador com 5 velocidades",
38          "price": 99.9,
39          "__v": 0,
40          "image": "sLMofO3sa.jpeg"
41        },
42        "unitPrice": 99.9,
43        "quantity": 2,
44        "amount": 199.8
45      }
46    ],
47    "totalAmount": 349.7,
48    "created": "2021-06-06T19:32:06.132Z",
49    "__v": 0
50  }
51 ]
```

- No Mongo Compass:

```
_id: ObjectId("60bd22b64aac221b18baf9eb")
customer: ObjectId("60bc117858f8c976026c8e61")
products: Array
  0: Object
    _id: ObjectId("60bd22b64aac221b18baf9ec")
    product: ObjectId("60bc4cd3a512b00c14451ef6")
    unitPrice: 149.9
    quantity: 1
    amount: 149.9
  1: Object
    _id: ObjectId("60bd22b64aac221b18baf9ed")
    product: ObjectId("60bc4d9ea512b00c14451ef7")
    unitPrice: 99.9
    quantity: 2
    amount: 199.8
totalAmount: 349.7
created: 2021-06-06T19:32:06.132+00:00
__v: 0
```

- No browser:

<http://localhost:5000/orders>



```
4 * [
5 * {
6   "_id": "60bd22b64aac221b18baf9eb",
7   "customer": {
8     "_id": "60bc117858f8c976026c8e61",
9     "name": "Roberto",
10    "lastname": "Pinheiro",
11    "email": "roberto_pinheiro@gmail.com",
12    "phone": "(11) 99905-3268"
13  },
14  "products": [
15    {
16      "_id": "60bd22b64aac221b18baf9ec",
17      "product": {
18        "stock": 25,
19        "available": true,
20        "_id": "60bc4cd3a512b00c14451ef6",
21        "sku": "A11",
22        "name": "Ventilador",
23        "description": "Ventilador de 3 velocidades",
24        "price": 149.9,
25        "__v": 0,
26        "image": "ZmUvgAkTK.jpeg"
27      },
28      "unitPrice": 149.9,
29      "quantity": 1,
30      "amount": 149.9
31    },
32    {
33      "_id": "60bd22b64aac221b18baf9ed",
34      "product": {
35        "stock": 45,
36        "available": true,
37        "_id": "60bc4d9ea512b00c14451ef7",
38        "sku": "B12",
39        "name": "Liquidificador",
40        "description": "Liquidificador com 5 velocidades",
41        "price": 99.9,
42        "__v": 0,
43        "image": "sLMoF03sa.jpeg"
44      },
45      "unitPrice": 99.9,
46      "quantity": 2,
47      "amount": 199.8
48    }
49  ],
50  "totalAmount": 349.7,
51  "created": "2021-06-06T19:32:06.132Z",
52  "__v": 0
53 }
54 ]
```


Cadastrando um novo pedido

POST - localhost:5000/orders

POST localhost:5000/ordersSend

200 OK25.6 ms265 B

JSONAuthQueryHeader1Docs

PreviewHeader8CookieTimeline

```
1 {
2   "customer": "60bc132058f8c976026c8e62",
3   "products": [
4     {
5       "product": "60bcf97c1eb9f1063c4b429d",
6       "unitPrice": 78.5,
7       "quantity": 1,
8       "amount": 78.5
9     }
10  ],
11  "totalAmount": 78.5
12 }
```

```
1 {
2   "_id": "60bd26534aac221b18baf9ee",
3   "customer": "60bc132058f8c976026c8e62",
4   "products": [
5     {
6       "_id": "60bd26534aac221b18baf9ef",
7       "product": "60bcf97c1eb9f1063c4b429d",
8       "unitPrice": 78.5,
9       "quantity": 1,
10      "amount": 78.5
11    }
12  ],
13  "totalAmount": 78.5,
14  "created": "2021-06-06T19:47:31.482Z",
15  "__v": 0
16 }
```

Listando pedidos

- No browser:



The screenshot shows a web browser window with the address bar displaying 'localhost:5000/orders'. Below the address bar, there are tabs for 'Apps', 'Seletores de Formu...', and 'jQuery Validatio'. The main content area of the browser displays a JSON response, which is a list of two orders. The first order has a customer named Roberto Pinheiro and contains a fan (Ventilador). The second order contains a liquidizer (Liquidificador). The JSON is formatted with syntax highlighting and line numbers on the left.

```
4  [
5  {
6    "_id": "60bd22b64aac221b18baf9eb",
7    "customer": {
8      "_id": "60bc117858f8c976026c8e61",
9      "name": "Roberto",
10     "lastname": "Pinheiro",
11     "email": "roberto_pinheiro@gmail.com",
12     "phone": "(11) 99905-3268"
13   },
14   "products": [
15     {
16       "_id": "60bd22b64aac221b18baf9ec",
17       "product": {
18         "stock": 25,
19         "available": true,
20         "_id": "60bc4cd3a512b00c14451ef6",
21         "sku": "A11",
22         "name": "Ventilador",
23         "description": "Ventilador de 3 velocidades",
24         "price": 149.9,
25         "__v": 0,
26         "image": "ZmUvgAkTK.jpeg"
27       },
28       "unitPrice": 149.9,
29       "quantity": 1,
30       "amount": 149.9
31     },
32     {
33       "_id": "60bd22b64aac221b18baf9ed",
34       "product": {
35         "stock": 45,
36         "available": true,
37         "_id": "60bc4d9ea512b00c14451ef7",
38         "sku": "B12",
39         "name": "Liquidificador",
40         "description": "Liquidificador com 5 velocidades",
41         "price": 99.9,
42         "__v": 0,
43         "image": "sLMoF0Jsa.jpeg"
44       },
45       "unitPrice": 99.9,
46       "quantity": 2,
47       "amount": 199.8
48     }
49   ],
50   "totalAmount": 349.7,
51   "created": "2021-06-06T19:32:06.132Z",
52   "__v": 0
53 }
54 ]
```

```
54 ▾ {
55   "_id": "60bd26534aac221b18baf9ee",
56   ▾ "customer": {
57     "_id": "60bc132058f8c976026c8e62",
58     "name": "Felipe",
59     "lastname": "Lima",
60     "email": "felipe_lima@gmail.com",
61     "phone": "(21) 98839-5723"
62   },
63   ▾ "products": [
64     ▾ {
65       "_id": "60bd26534aac221b18baf9ef",
66       ▾ "product": {
67         "stock": 6,
68         "available": true,
69         "_id": "60bcf97c1eb9f1063c4b429d",
70         "sku": "E67",
71         "name": "Rádio relógio",
72         "description": "Rádio relógio digital",
73         "price": 78.5,
74         "image": "-X04H84pt.jpeg",
75         "__v": 0
76       },
77       "unitPrice": 78.5,
78       "quantity": 1,
79       "amount": 78.5
80     }
81   ],
82   "totalAmount": 78.5,
83   "created": "2021-06-06T19:47:31.482Z",
84   "__v": 0
85 }
86 ]
```

Aula 11 - Atualizar pedido

Exibir dados de um pedido específico

controllers\ordersController.js

```
const Order = require('../models/Order');

// cadastrar pedido
exports.create = async (req, res, next) => {
  try {
    const order = new Order(req.body);
    await order.save();
    res.json(order);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};

// listar pedidos

exports.index = async (req, res, next) => {
  try {
    const orders = await Order.find({})
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });
    res.json(orders);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};

// exibir detalhes de um pedido por id
exports.show = async (req, res, next) => {
  try {
    const order = await Order.findById(req.params.id)
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });

    if(!order){
      res.status(404).json({
        message: "O pedido não existe!"
      });
      next();
    }
    res.json(order);
  } catch (error) {
```

```

    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

```

routes\ordersRoutes.js

```

const express = require('express');
const router = express.Router();

const ordersController = require('../controllers/ordersController');

module.exports = function() {
  // post: /orders
  router.post('/orders', ordersController.create);


  // get: /orders
  router.get('/orders', ordersController.index);

  // get: /orders/:id
  router.get('/orders/:id', ordersController.show);

  return router;
};

```

GET - localhost:5000/orders/60bd26534aac221b18baf9ee

GET	localhost:5000/orders/60bd26534aac221b18baf9ee	Send	200 OK	107 ms	523 B
Body	Auth	Query	Header	Docs	Preview
 <p>Select a body type from above</p>			<pre> 1+ { 2 "_id": "60bd26534aac221b18baf9ee", 3+ "customer": { 4 "_id": "60bc132058f8c976026c8e62", 5 "name": "Felipe", 6 "lastname": "Lima", 7 "email": "felipe_lima@gmail.com", 8 "phone": "(21) 98839-5723" 9 }, 10+ "products": [11+ { 12 "_id": "60bd26534aac221b18baf9ef", 13+ "product": { 14 "stock": 6, 15 "available": true, 16 "_id": "60bcf97c1eb9f1063c4b429d", 17 "sku": "E67", 18 "name": "Rádio relógio", 19 "description": "Rádio relógio digital", 20 "price": 78.5, 21 "image": "-X04H84pt.jpeg", 22 "__v": 0 23 }, 24 "unitPrice": 78.5, 25 "quantity": 1, 26 "amount": 78.5 27 } 28], 29 "totalAmount": 78.5, 30 "created": "2021-06-06T19:47:31.482Z", 31 "__v": 0 32 } </pre>		

GET - localhost:5000/orders/60bd26534aac221b18baf9ef

GET ▾	localhost:5000/orders/60bd26534aac221b18baf9ef	Send	404 Not Found	9.64 ms	35 B
Body ▾	Auth ▾	Query	Header	Docs	Preview ▾
				Header 8	Cookie
				Timeline	
				1 { 2 "message": "O pedido não existe!" 3 }	

Atualizando um pedido

controllers\ordersController.js

```
const Order = require('../models/Order');
```

```
// cadastrar pedido
```

```
exports.create = async (req, res, next) => {  
  try {  
    const order = new Order(req.body);  
    await order.save();  
    res.json(order);  
  } catch (error) {  
    res.status(400).json({  
      message: "Erro ao processar a requisição!"  
    });  
  }  
};
```

```
// listar pedidos
```

```
exports.index = async (req, res, next) => {  
  try {  
    const orders = await Order.find({})  
      .populate('customer')  
      .populate({  
        path: 'products.product',  
        model: 'Product'  
      });  
    res.json(orders);  
  } catch (error) {  
    res.status(400).json({  
      message: "Erro ao processar a requisição!"  
    });  
  }  
};
```

```
// exibir detalhes de um pedido por id
```

```
exports.show = async (req, res, next) => {  
  try {  
    const order = await Order.findById(req.params.id)  
      .populate('customer')  
      .populate({  
        path: 'products.product',  
        model: 'Product'  
      });  
  }  
};
```

```

    if(!order){
      res.status(404).json({
        message: "O pedido não existe!"
      });
      next();
    }
    res.json(order);
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

// atualizar dados de um pedido
exports.update = async (req, res, next) => {
  try {
    const order = await Order.findOneAndUpdate(
      { _id: req.params.id },
      req.body,
      { new: true }
    )
    .populate('customer')
    .populate({
      path: 'products.product',
      model: 'Product'
    });
    res.json(order);
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
}

```

routes\ordersRoutes.js

```

const express = require('express');
const router = express.Router();

const ordersController = require('../controllers/ordersController');

module.exports = function() {
  // post: /orders
  router.post('/orders', ordersController.create);

  // get: /orders
  router.get('/orders', ordersController.index);

  // get: /orders/:id
  router.get('/orders/:id', ordersController.show);

  // put: /orders/:id
  router.put('/orders/:id', ordersController.update);

  return router;
};

```

PUT - localhost:5000/orders/60bd26534aac221b18baf9ee

PUTlocalhost:5000/orders/60bd26534aac221b18baf9eeSend

200 OK344 ms521 B

JSONAuthQueryHeader1Docs

```
1 {
2   "customer": {
3     "_id": "60bc132058f8c976026c8e62",
4     "name": "Felipe",
5     "lastname": "Lima",
6     "email": "felipe_lima@gmail.com",
7     "phone": "(21) 98839-5723"
8   },
9   "products": [
10    {
11      "_id": "60bd26534aac221b18baf9ef",
12      "product": {
13        "stock": 6,
14        "available": true,
15        "_id": "60bcf97c1eb9f1063c4b429d",
16        "sku": "E67",
17        "name": "Rádio relógio",
18        "description": "Rádio relógio digital",
19        "price": 78.5,
20        "image": "-X04HB4pt.jpeg",
21        "__v": 0
22      },
23      "unitPrice": 78.5,
24      "quantity": 2,
25      "amount": 157.00
26    }
27  ],
28  "totalAmount": 157.00
29 }
```

PreviewHeader8CookieTimeline


```
1 {
2   "_id": "60bd26534aac221b18baf9ee",
3   "customer": {
4     "_id": "60bc132058f8c976026c8e62",
5     "name": "Felipe",
6     "lastname": "Lima",
7     "email": "felipe_lima@gmail.com",
8     "phone": "(21) 98839-5723"
9   },
10  "products": [
11    {
12      "_id": "60bd26534aac221b18baf9ef",
13      "product": {
14        "stock": 6,
15        "available": true,
16        "_id": "60bcf97c1eb9f1063c4b429d",
17        "sku": "E67",
18        "name": "Rádio relógio",
19        "description": "Rádio relógio digital",
20        "price": 78.5,
21        "image": "-X04HB4pt.jpeg",
22        "__v": 0
23      },
24      "unitPrice": 78.5,
25      "quantity": 2,
26      "amount": 157
27    }
28  ],
29  "totalAmount": 157,
30  "created": "2021-06-06T19:47:31.482Z",
31  "__v": 0
32 }
```

GET - localhost:5000/orders/60bd26534aac221b18baf9ee

GETlocalhost:5000/orders/60bd26534aac221b18baf9eeSend

200 OK26.7 ms521 B

BodyAuthQueryHeaderDocs



Select a body type from above

PreviewHeader8CookieTimeline

```
1 {
2   "_id": "60bd26534aac221b18baf9ee",
3   "customer": {
4     "_id": "60bc132058f8c976026c8e62",
5     "name": "Felipe",
6     "lastname": "Lima",
7     "email": "felipe_lima@gmail.com",
8     "phone": "(21) 98839-5723"
9   },
10  "products": [
11    {
12      "_id": "60bd26534aac221b18baf9ef",
13      "product": {
14        "stock": 6,
15        "available": true,
16        "_id": "60bcf97c1eb9f1063c4b429d",
17        "sku": "E67",
18        "name": "Rádio relógio",
19        "description": "Rádio relógio digital",
20        "price": 78.5,
21        "image": "-X04HB4pt.jpeg",
22        "__v": 0
23      },
24      "unitPrice": 78.5,
25      "quantity": 2,
26      "amount": 157
27    }
28  ],
29  "totalAmount": 157,
30  "created": "2021-06-06T19:47:31.482Z",
31  "__v": 0
32 }
```


Aula 12 - Eliminar pedido

controllers\ordersController.js

```
const Order = require('../models/Order');

// cadastrar pedido
exports.create = async (req, res, next) => {
  try {
    const order = new Order(req.body);
    await order.save();
    res.json(order);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};

// listar pedidos
exports.index = async (req, res, next) => {
  try {
    const orders = await Order.find({})
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });
    res.json(orders);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};

// exibir detalhes de um pedido por id
exports.show = async (req, res, next) => {
  try {
    const order = await Order.findById(req.params.id)
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });
    if(!order){
      res.status(404).json({
        message: "O pedido não existe!"
      });
      next();
    }
    res.json(order);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};
```

```

    }
  };

  // atualizar dados de um pedido
  exports.update = async (req, res, next) => {
    try {
      const order = await Order.findOneAndUpdate(
        { _id: req.params.id },
        req.body,
        { new: true }
      )
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });
      res.json(order);
    } catch (error) {
      res.status(400).json({
        message: 'Erro ao processar a requisição!'
      });
    }
  }
}

```

```

// Excluir um pedido por id
exports.delete = async (req, res, next) => {
  try {
    await Order.findOneAndDelete({
      _id: req.params.id
    });

    res.json({
      message: 'Pedido excluído com sucesso!'
    });
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

```

routes\ordersRoutes.js

```
const express = require('express');
const router = express.Router();
const ordersController = require('../controllers/ordersController');

module.exports = function() {
  // post: /orders
  router.post('/orders', ordersController.create);

  // get: /orders
  router.get('/orders', ordersController.index);

  // get: /orders/:id
  router.get('/orders/:id', ordersController.show);

  // put: /orders/:id
  router.put('/orders/:id', ordersController.update);

  // delete: /orders/:id
  router.delete('/orders/:id', ordersController.delete);

  return router;
};
```


Excluindo um pedido

DELETE - localhost:5000/orders/60bd26534aac221b18baf9ee

DELETE	localhost:5000/orders/60bd26534aac221b18baf9ee	Send	200 OK	66.5 ms	42 B	
Body	Auth	Query	Header	Docs	Preview	
				Header 8	Cookie	Timeline
				<pre>1 { 2 "message": "Pedido excluido com sucesso!" 3 }</pre>		

Listando pedidos

GET - localhost:5000/orders

GET	localhost:5000/orders	Send	200 OK	61.3 ms	823 B	
Body	Auth	Query	Header	Docs	Preview	
				Header 8	Cookie	Timeline
 Select a body type from above				<pre>1 { 2 "_id": "60bd22b64aac221b18baf9eb", 3 "customer": { 4 "_id": "60bc117858f8c976026c8e61", 5 "name": "Roberto", 6 "lastname": "Pinheiro", 7 "email": "roberto.pinheiro@gmail.com", 8 "phone": "(11) 99905-3268" 9 }, 10 "products": [11 { 12 "_id": "60bd22b64aac221b18baf9ec", 13 "product": { 14 "stock": 25, 15 "available": true, 16 "_id": "60bc4cd3a512b00c14451ef6", 17 "sku": "A11", 18 "name": "Ventilador", 19 "description": "Ventilador de 3 velocidades", 20 "price": 149.9, 21 "__v": 0, 22 "image": "ZmUvgAkTK.jpeg" 23 }, 24 "unitPrice": 149.9, 25 "quantity": 1, 26 "amount": 149.9 27 }, 28 { 29 "_id": "60bd22b64aac221b18baf9ed", 30 "product": { 31 "stock": 45, 32 "available": true, 33 "_id": "60bc4d9ea512b00c14451ef7", 34 "sku": "B12", 35 "name": "Liquidificador", 36 "description": "Liquidificador com 5 velocidades", 37 "price": 99.9, 38 "__v": 0, 39 "image": "sLMofO3sa.jpeg" 40 }, 41 "unitPrice": 99.9, 42 "quantity": 2, 43 "amount": 199.8 44 } 45], 46 "totalAmount": 349.7, 47 "created": "2021-06-06T19:32:06.132Z", 48 "__v": 0 49 } 50 }</pre>		

Aula 13 - Obter pedidos por cliente

controllers\ordersController.js

```
const Order = require('../models/Order');

// cadastrar pedido
exports.create = async (req, res, next) => {
  try {
    const order = new Order(req.body);
    await order.save();
    res.json(order);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};

// listar pedidos
exports.index = async (req, res, next) => {
  try {
    const orders = await Order.find({})
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });
    res.json(orders);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};

// exibir detalhes de um pedido por id
exports.show = async (req, res, next) => {
  try {
    const order = await Order.findById(req.params.id)
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });
    if(!order){
      res.status(404).json({
        message: "O pedido não existe!"
      });
      next();
    }
    res.json(order);
  } catch (error) {
    res.status(400).json({
      message: "Erro ao processar a requisição!"
    });
  }
};
```

```

    }
  };

  // atualizar dados de um pedido
  exports.update = async (req, res, next) => {
    try {
      const order = await Order.findOneAndUpdate(
        { _id: req.params.id },
        req.body,
        { new: true }
      )
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });
      res.json(order);
    } catch (error) {
      res.status(400).json({
        message: 'Erro ao processar a requisição!'
      });
    }
  }
}

```

```

// Excluir um pedido por id
exports.delete = async (req, res, next) => {
  try {
    await Order.findOneAndDelete({
      _id: req.params.id
    });

    res.json({
      message: 'Pedido excluído com sucesso!'
    });
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

```

```

// exibir pedidos por cliente
exports.byCustomer = async (req, res, next) => {
  try {
    const orders = await Order.find({ customer: req.params.id })
      .populate('customer')
      .populate({
        path: 'products.product',
        model: 'Product'
      });
    res.json(orders);
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

```

routes\ordersRoutes.js

```
const express = require('express');
const router = express.Router();
const ordersController = require('../controllers/ordersController');

module.exports = function() {
  // post: /orders
  router.post('/orders', ordersController.create);

  // get: /orders
  router.get('/orders', ordersController.index);

  // get: /orders/:id
  router.get('/orders/:id', ordersController.show);

  // get: /orders/customer/:id
  router.get('/orders/customer/:id', ordersController.byCustomer);

  // put: /orders/:id
  router.put('/orders/:id', ordersController.update);

  // delete: /orders/:id
  router.delete('/orders/:id', ordersController.delete);

  return router;
};
```

- Cadastre mais dois pedidos:

POST - localhost:5000/orders

POST	localhost:5000/orders	Send	200 OK	331 ms	265 B			
JSON	Auth	Query	Header 1	Docs	Preview	Header 8	Cookie	Timeline
<pre>1 { 2 "customer": "60bc132058f8c976026c8e62", 3 "products": [4 { 5 "product": "60bcf97c1eb9f1063c4b429d", 6 "unitPrice": 78.5, 7 "quantity": 1, 8 "amount": 78.5 9 } 10], 11 "totalAmount": 78.5 12 }</pre>			<pre>1 { 2 "_id": "60bd5b4a1396e037288e9bdb", 3 "customer": "60bc132058f8c976026c8e62", 4 "products": [5 { 6 "_id": "60bd5b4a1396e037288e9bdc", 7 "product": "60bcf97c1eb9f1063c4b429d", 8 "unitPrice": 78.5, 9 "quantity": 1, 10 "amount": 78.5 11 } 12], 13 "totalAmount": 78.5, 14 "created": "2021-06-06T23:33:30.802Z", 15 "__v": 0 16 }</pre>					

POST - localhost:5000/orders

POST	localhost:5000/orders	Send	200 OK	59.5 ms	265 B			
JSON	Auth	Query	Header 1	Docs	Preview	Header 8	Cookie	Timeline
<pre>1 { 2 "customer": "60bc132058f8c976026c8e62", 3 "products": [4 { 5 "product": "60bc4d9ea512b00c14451ef7", 6 "unitPrice": 99.90, 7 "quantity": 1, 8 "amount": 99.90 9 } 10], 11 "totalAmount": 99.90 12 }</pre>			<pre>1 { 2 "_id": "60bd5ca41396e037288e9bdd", 3 "customer": "60bc132058f8c976026c8e62", 4 "products": [5 { 6 "_id": "60bd5ca41396e037288e9bde", 7 "product": "60bc4d9ea512b00c14451ef7", 8 "unitPrice": 99.9, 9 "quantity": 1, 10 "amount": 99.9 11 } 12], 13 "totalAmount": 99.9, 14 "created": "2021-06-06T23:39:16.973Z", 15 "__v": 0 16 }</pre>					

Listando pedidos por um cliente específico

GET - localhost:5000/orders/customer/60bc132058f8c976026c8e62

GET ▾ localhost:5000/orders/customer/60bc132058f8c976026c8e62

```
[
  {
    "_id": "60bd5b4a1396e037288e9bdb",
    "customer": {
      "_id": "60bc132058f8c976026c8e62",
      "name": "Felipe",
      "lastname": "Lima",
      "email": "felipe_lima@gmail.com",
      "phone": "(21) 98839-5723"
    },
    "products": [
      {
        "_id": "60bd5b4a1396e037288e9bdc",
        "product": {
          "stock": 6,
          "available": true,
          "_id": "60bcf97c1eb9f1063c4b429d",
          "sku": "E67",
          "name": "Rádio relógio",
          "description": "Rádio relógio digital",
          "price": 78.5,
          "image": "-X04HB4pt.jpeg",
          "__v": 0
        },
        "unitPrice": 78.5,
        "quantity": 1,
        "amount": 78.5
      }
    ],
    "totalAmount": 78.5,
    "created": "2021-06-06T23:33:30.802Z",
    "__v": 0
  },
  {
    "_id": "60bd5ca41396e037288e9bdd",
    "customer": {
      "_id": "60bc132058f8c976026c8e62",
      "name": "Felipe",
      "lastname": "Lima",
      "email": "felipe_lima@gmail.com",
      "phone": "(21) 98839-5723"
    },
    "products": [
      {
        "_id": "60bd5ca41396e037288e9bde",
        "product": {
          "stock": 45,
          "available": true,
          "_id": "60bc4d9ea512b00c14451ef7",
          "sku": "B12",
          "name": "Liquidificador",
          "description": "Liquidificador com 5 velocidades",
          "price": 99.9,
          "__v": 0,
          "image": "sLMoFOJsa.jpeg"
        },
        "unitPrice": 99.9,
        "quantity": 1,
        "amount": 99.9
      }
    ],
    "totalAmount": 99.9,
    "created": "2021-06-06T23:39:16.973Z",
    "__v": 0
  }
]
```

Aula 14 - Buscar produtos por nome

controllers\productsController.js

```
const multer = require('multer');
const multerConfig = require('../utils/multerConfig');

const Product = require('../models/Product');

const upload = multer(multerConfig).single('image');

exports.fileUpload = (req, res, next) => {
  upload(req, res, function(error) {
    if(error){
      res.json({message: error});
    }
    return next();
  });
};

// cadastrar produto
exports.create = async (req, res) => {
  const product = new Product(req.body);

  try {
    if (req.file && req.file.filename) {
      product.image = req.file.filename;
    }
    await product.save();
    res.status(201).json({
      message: "Produto cadastrado com sucesso!"
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um produto com o sku: ${req.body.sku}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
};

// listar produtos
exports.index = async (req, res) => {
  try {
    const products = await Product.find({});
    res.json(products);
  } catch (error) {
    console.log(error);
    res.send(error);
    next();
  }
};
```

```

// exibir detalhes de um produto por id
exports.show = async (req, res, next) => {
  try {
    const product = await Product.findById(req.params.id);
    if(!product){
      res.status(404).json({
        message: "O produto não existe!"
      });
    }
    res.json(product);
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

// atualizar dados de um produto
exports.update = async (req, res, next) => {
  try {
    let newProduct = req.body;

    if (req.file && req.file.filename) {
      newProduct.image = req.file.filename;
    } else {
      const product = await Product.findById(req.params.id);
      newProduct.image = product.image;
    }

    const productUpdated = await Product.findOneAndUpdate(
      { _id: req.params.id },
      newProduct,
      { new: true }
    );

    res.json({
      message: 'Produto atualizado com sucesso!'
    });
  } catch (error) {
    if(error.code === 11000){
      res.status(400).json({
        message: `Já existe um produto com o sku: ${req.body.sku}`
      });
    } else {
      res.status(400).json({
        message: "Erro ao processar a requisição!"
      });
    }
  }
};

// Excluir um produto por id
exports.delete = async (req, res, next) => {
  try {
    await Product.findOneAndDelete({
      _id: req.params.id
    });
  }
};

```

```

    res.json({
      message: 'Produto excluído com sucesso!'
    });
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
};

exports.search = async (req, res, next) => {
  try {
    const products = await Product.find({
      name: new RegExp(req.params.query, 'i')
    });
    res.json(products);
  } catch (error) {
    res.status(400).json({
      message: 'Erro ao processar a requisição!'
    });
  }
}

```

routes\productsRoutes.js

```

const express = require('express');

const router = express.Router();

const productsController = require('../controllers/productsController');

module.exports = function() {
  // post: /products
  router.post('/products', productsController.fileUpload, productsController.create);

  // get: /products
  router.get('/products', productsController.index);

  // get: /products/:id
  router.get('/products/:id', productsController.show);

  // put: /products/:id
  router.put('/products/:id', productsController.fileUpload, productsController.update);

  // delete: products/:id
  router.delete('/products/:id', productsController.delete);

  // get: /products/search/:query
  router.get('/products/search/:query', productsController.search);

  return router;
};

```

- Cadastre dois novos produtos

```
{
  "sku": "A12",
  "name": "Ventilador",
  "description": "Ventilador de parede",
  "price": 139.50,
  "stock": 12,
  "available": true
}
```

POST - localhost:5000/products/

POST localhost:5000/products/ Send

201 Created 155 ms 45 B

Multipart 7 Auth Query Header 1 Docs

sku	A12	▼	✓	🗑
name	Ventilador	▼	✓	🗑
description	Ventilador de parede	▼	✓	🗑
image	ventilador_de_parede.jpg	▼	✓	🗑
price	139.50	▼	✓	🗑
stock	12	▼	✓	🗑
available	true	▼	✓	🗑

Preview Header 8 Cookie Timeline

```
1 {
2   "message": "Produto cadastrado com sucesso!"
3 }
```

```
{
  "sku": "A13",
  "name": "Ventilador",
  "description": "Ventilador de teto",
  "price": 109.50,
  "stock": 10,
  "available": true
}
```

POST - localhost:5000/products/

POST localhost:5000/products/ Send

201 Created 72.4 ms 45 B

Multipart 7 Auth Query Header 1 Docs

sku	A13	▼	✓	🗑
name	Ventilador	▼	✓	🗑
description	Ventilador de teto	▼	✓	🗑
image	ventilador_de_teto.jpg	▼	✓	🗑
price	109.50	▼	✓	🗑
stock	10	▼	✓	🗑
available	true	▼	✓	🗑

Preview Header 8 Cookie Timeline

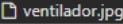
```
1 {
2   "message": "Produto cadastrado com sucesso!"
3 }
```

Atualizando dados de um ventilador

PUT - localhost:5000/products/60bc4cd3a512b00c14451ef6

PUT localhost:5000/products/60bc4cd3a512b00c14451ef6 Send 200 OK 195 ms 45 B

Multipart 7 Auth Query Header 1 Docs

sku	A11	▼	✓	🗑
name	Ventilador	▼	✓	🗑
description	Ventilador com pedestal de 3 velocidades	▼	✓	🗑
image		▼	✓	🗑
price	149.90	▼	✓	🗑
stock	25	▼	✓	🗑
available	true	▼	✓	🗑

Preview 8 Header 8 Cookie Timeline

```
1 {
2   "message": "Produto atualizado com sucesso!"
3 }
```

Listando ventiladores cadastrados

GET - localhost:5000/products/search/ventilador

GET localhost:5000/products/search/ventilador Send 200 OK 35.9 ms 556 B

Body Auth Query Header Docs

Select a body type from above

Preview 8 Header 8 Cookie Timeline

```
1 [
2   {
3     "stock": 25,
4     "available": true,
5     "_id": "60bc4cd3a512b00c14451ef6",
6     "sku": "A11",
7     "name": "Ventilador",
8     "description": "Ventilador com pedestal de 3 velocidades",
9     "price": 149.9,
10    "__v": 0,
11    "image": "hz5gKTfE0.jpeg"
12  },
13  {
14    "stock": 12,
15    "available": true,
16    "_id": "60bd637a7b6eaf0c040d68e0",
17    "sku": "A12",
18    "name": "Ventilador",
19    "description": "Ventilador de parede",
20    "price": 139.5,
21    "image": "EYhwgyDwM.jpeg",
22    "__v": 0
23  },
24  {
25    "stock": 10,
26    "available": true,
27    "_id": "60bd64457b6eaf0c040d68e1",
28    "sku": "A13",
29    "name": "Ventilador",
30    "description": "Ventilador de teto",
31    "price": 109.5,
32    "image": "EN8o6zoHP.jpeg",
33    "__v": 0
34  }
35 ]
```

localhost:5000/products/search/rádio

GET

localhost:5000/products/search/rádio

Send

200 OK

120 ms

186 B

Body

Auth

Query

Header

Docs

Preview

Header

Cookie

Timeline

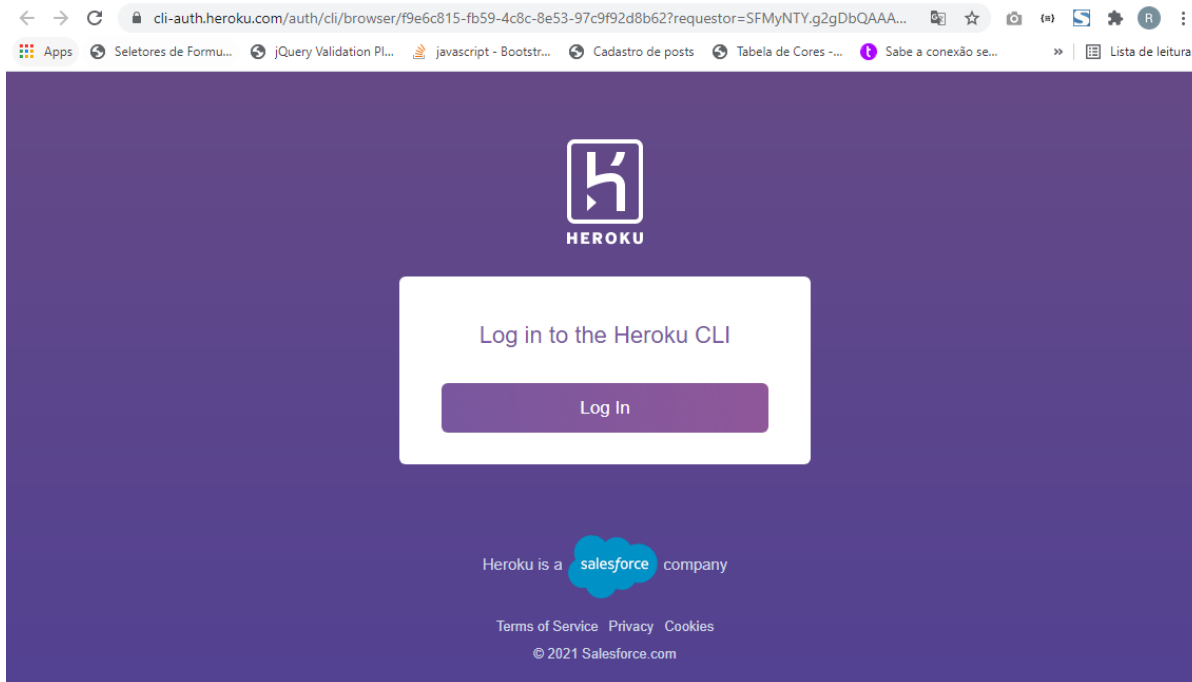
```
1 [
2   {
3     "stock": 6,
4     "available": true,
5     "_id": "60bcf97c1eb9f1063c4b429d",
6     "sku": "E67",
7     "name": "Rádio relógio",
8     "description": "Rádio relógio digital",
9     "price": 78.5,
10    "image": "-X04HB4pt.jpeg",
11    "__v": 0
12  }
13 ]
```

Aula 15 - Deploy da aplicação em Heroku e MongoDB Atlas

- Instale Heroku CLI em seu computador.

- No terminal, entre com o comando:

`heroku login`



`heroku create api-store-felipemx`

```
beto1@DESKTOP-NUIKR9E MINGW64 /c/felipemx/store-api
$ heroku create api-store-felipemx
» Warning: heroku update available from 7.47.6 to 7.54.0.
Creating ● api-store-felipemx... done
https://api-store-felipemx.herokuapp.com/ | https://git.heroku.com/api-store-felipemx.git
```

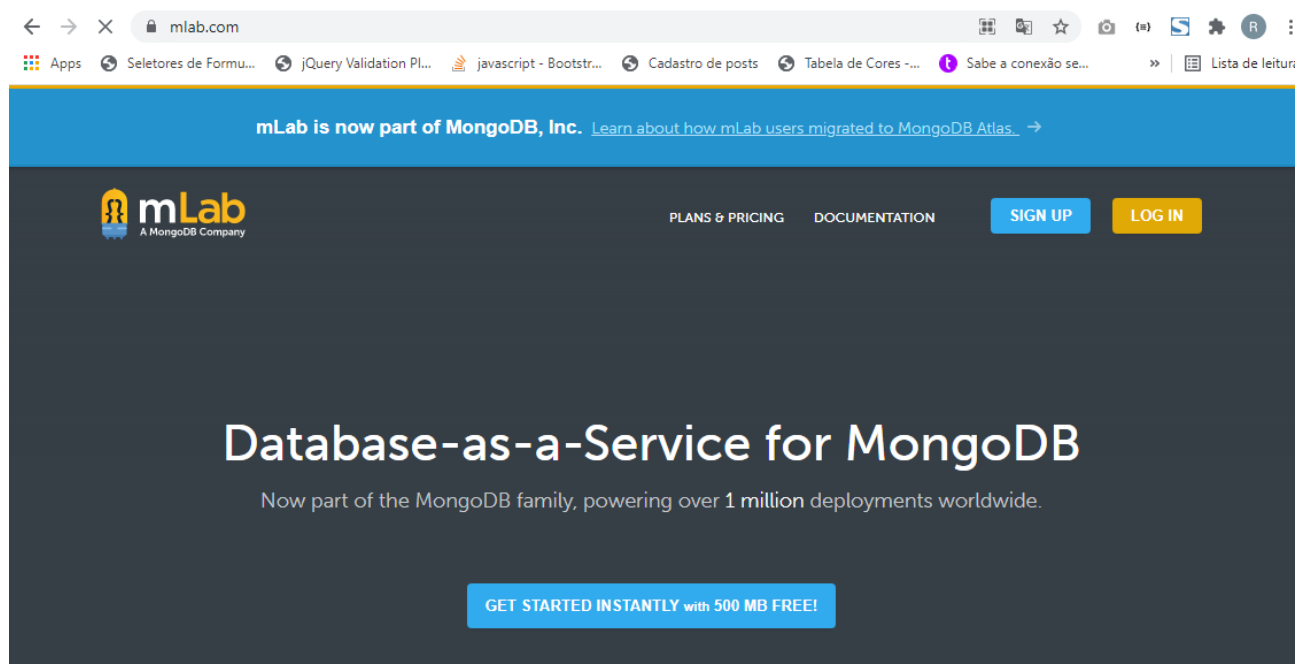

Configurações

package.json

```
{
  "name": "store-api",
  "version": "1.0.0",
  "description": "api rest",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "dev": "nodemon index.js",
    "start": "node index.js"
  },
  "keywords": [
    "api"
  ],
  "author": "Roberto Pinheiro",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.19.0",
    "cors": "^2.8.5",
    "express": "^4.17.1",
    "mongoose": "^5.12.13",
    "multer": "^1.4.2",
    "shortid": "^2.2.16"
  }
}
```

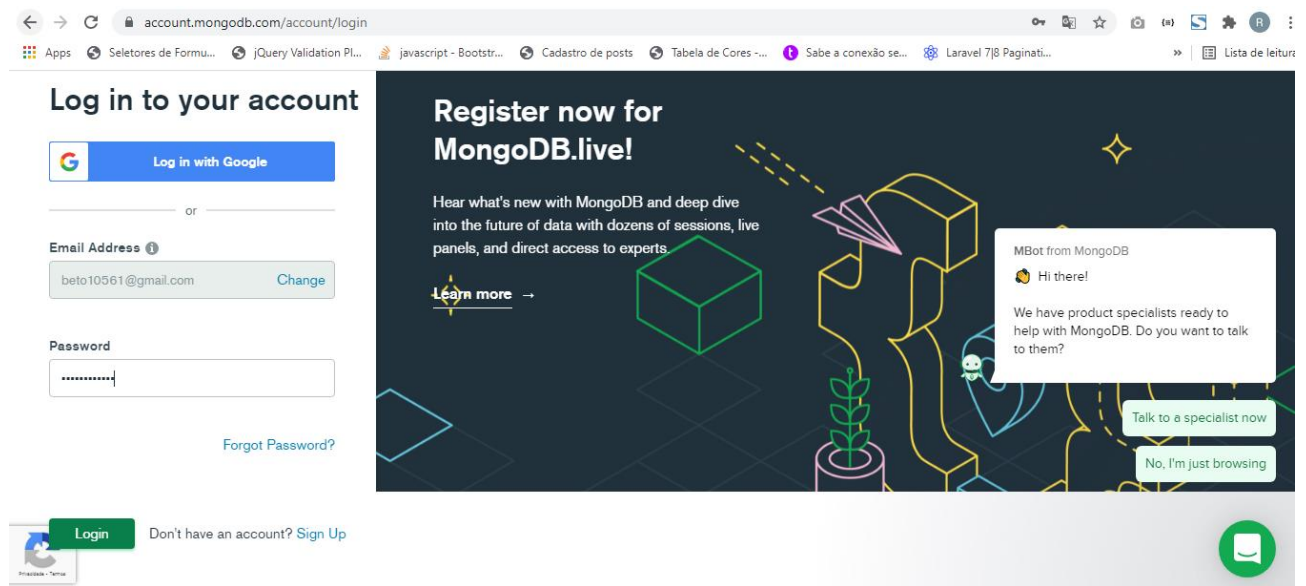
- Acesse o site do mlab:

<https://mlab.com/>

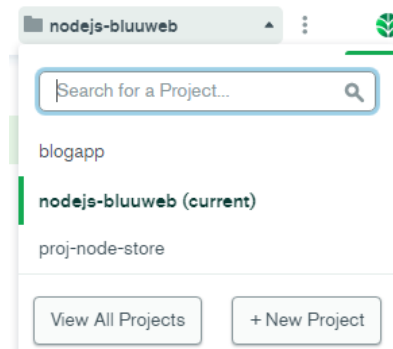


- Crie uma conta e faça o login.

<https://account.mongodb.com/account/login>



- Crie um novo projeto chamado "api-store-felipemx"



ORION3 > [PROJECTS](#)

Create a Project

Name Your Project Add Members

Next

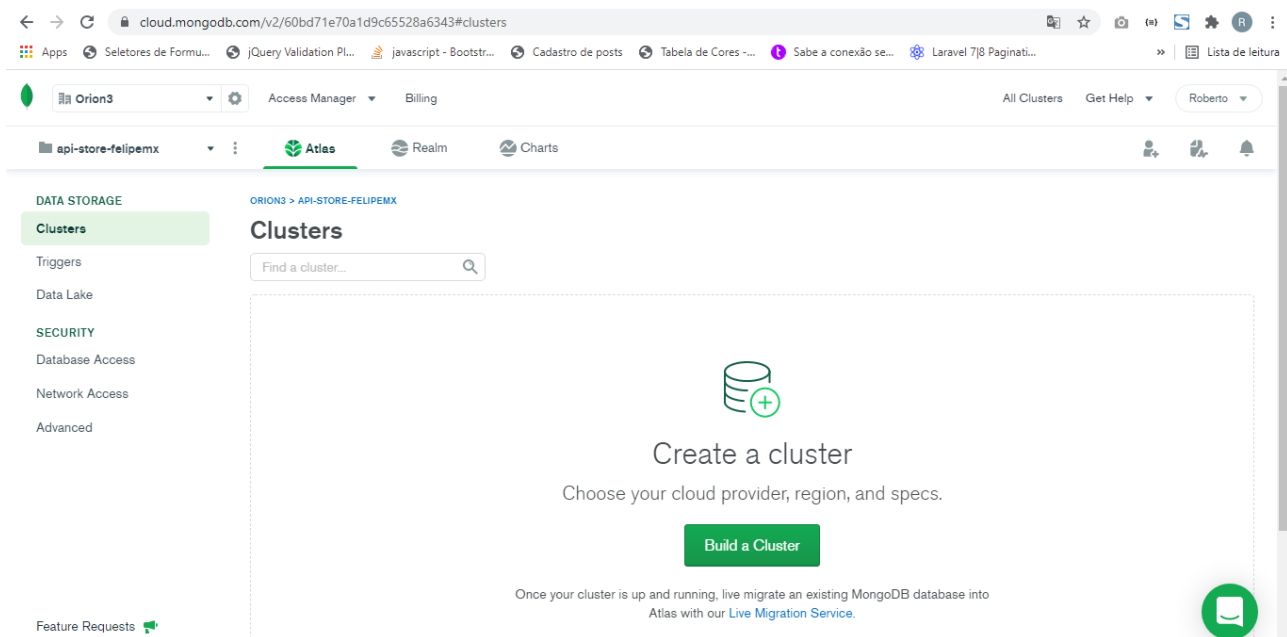
Name Your Project

Project names have to be unique within the organization (and other restrictions).

api-store-felipemx

Cancel

Next



- Crie um cluster (FREE)

CLUSTERS > CREATE A SHARED CLUSTER

Create a Shared Cluster

Welcome to MongoDB Atlas! We've recommended some of our most popular options, but feel free to customize your cluster to your needs. For more information, check our [documentation](#).

Cloud Provider & Region AWS, N. Virginia (us-east-1) ▾

★ Recommended region ⓘ

NORTH AMERICA	EUROPE	ASIA
Oregon (us-west-2) ★	Ireland (eu-west-1) ★	Singapore (ap-southeast-1) ★
N. Virginia (us-east-1) ★	Frankfurt (eu-central-1) ★	Mumbai (ap-south-1)
	AUSTRALIA	
	Sydney (ap-southeast-2) ★	

Cluster Tier M0 Sandbox (Shared RAM, 512 MB Storage) >
Encrypted

Additional Settings MongoDB 4.4, No Backup >

Cluster Name Cluster1 ▾

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

Cluster1

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](#)

- Aguarde a criação do cluster (de 1 a 3 minutos).

The screenshot shows the MongoDB Atlas interface. At the top, there's a 'Clusters' section with a search bar and a 'Create a New Cluster' button. Below this, a modal window displays the details of a cluster named 'Cluster1' in the 'Sandbox' tier. The cluster is in the process of being created, with a message: 'Your cluster is being created.. New clusters take between 1-3 minutes to provision.' The modal also shows the cluster tier (M0 Sandbox (General)), region (AWS / N. Virginia (us-east-1)), type (Replica Set - 3 nodes), and linked realm app (None Linked). The background shows the main dashboard with various navigation options and a sidebar.

- Clique no botão "**COLLECTIONS**":

The screenshot shows the MongoDB Atlas interface for a specific cluster named 'Cluster1'. The 'Collections' tab is selected, showing a list of collections (currently empty). The page includes a sidebar with navigation options like 'Overview', 'Real Time', 'Metrics', 'Collections', 'Search', 'Profiler', 'Performance Advisor', 'Online Archive', and 'Command Line Tools'. A 'Visualize Your Data' button is visible. Below the collections list, there's a section titled 'Explore Your Data' with instructions on how to find, index, aggregate, and search data. It includes buttons for 'Load a Sample Dataset' and 'Add My Own Data', and a link to 'Learn more in Docs and Tutorials'.

- Clique no botão "Add My Own Data":

×

Create Database

DATABASE NAME ?

ecommerce

COLLECTION NAME ?

customers

☐ Capped Collection

Before MongoDB can save your new database, a collection name must be specified at the time of creation.

Cancel

Create

- Clique no botão "Create":

×

Create Database

DATABASE NAME ?

ecommerce

COLLECTION NAME ?

customers

☐ Capped Collection

Before MongoDB can save your new database, a collection name must be specified at the time of creation.

Cancel

Create

cloud.mongodb.com/v2/60bd71e70a1d9c65528a6343#metrics/replicaSet/60be168d3426ce3fac760c44/explorer/ecommerce/customers/find

Orion3 Access Manager Billing All Clusters Get Help Roberto

api-store-felipemx Atlas Realm Charts

DATA STORAGE Clusters Triggers Data Lake SECURITY Database Access Network Access Advanced

ORION2 > API-STORE-FELIPEMX > CLUSTERS

Cluster1 VERSION 4.4.6 REGION AWS N. Virginia (us-east-1)

Overview Real Time Metrics Collections Search Profiler Performance Advisor Online Archive Command Line Tools

DATABASES: 1 COLLECTIONS: 1 VISUALIZE YOUR DATA REFRESH

+ Create Database NAMESPACES

ecommerce customers

ecommerce.customers COLLECTION SIZE: 0B TOTAL DOCUMENTS: 0 INDEXES TOTAL SIZE: 4KB Find Indexes Schema Anti-Patterns Aggregation Search Indexes

INSERT DOCUMENT

FILTER {"filter": "example"} Find Reset

QUERY RESULTS 0

Feature Requests

- Clique no botão "INSERT DOCUMENT"

Insert to Collection

VIEW {} ≡

```
1 _id : ObjectId("60be1af228a842df75573ea6 ")
2 name : "Roberto "
3 lastname : "Pinheiro "
4 email : "roberto_pinheiro "
5 phone : "(11) 99905-3268 "
```

ObjectId
String
String
String
String

Cancel

Insert

Cluster1 VERSION 4.4.6 REGION AWS N. Virginia (us-east-1)

Overview Real Time Metrics Collections Search Profiler Performance Advisor Online Archive Command Line Tools

DATABASES: 1 COLLECTIONS: 1 VISUALIZE YOUR DATA REFRESH

+ Create Database NAMESPACES

ecommerce customers

ecommerce.customers COLLECTION SIZE: 0B TOTAL DOCUMENTS: 0 INDEXES TOTAL SIZE: 4KB Find Indexes Schema Anti-Patterns Aggregation Search Indexes

INSERT DOCUMENT

FILTER {"filter": "example"} Find Reset

QUERY RESULTS 1-1 OF 1

```
_id: ObjectId("60be1af228a842df75573ea6")
name: "Roberto"
lastname: "Pinheiro"
email: "roberto_pinheiro"
phone: "(11) 99905-3268"
```

- Em "Cluster1" clique no botão "CONNECT":

×

Connect to Cluster1

Setup connection security

Choose a connection method

Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

You can't connect yet. Set up your user security permission below.

1 Add a connection IP address

✓ An IP address has been added to the IP Access List. Add another address in the [IP Access List tab](#).

2 Create a Database User

This first user will have [atlasAdmin](#) permissions for this project.

Keep your credentials handy, you'll need them for the next step.

Username

betopinheiro

Password

Autogenerate Secure Password

SHOW

Create Database User

Close

Choose a connection method

- Clique no botão "Create Database User":

Connect to Cluster1

Setup connection security

Choose a connection method

Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

You're ready to connect. Choose how you want to connect in the next step.

1 Add a connection IP address

✓ An IP address has been added to the IP Access List. Add another address in the [IP Access List tab](#).

2 Create a Database User

✓ A MongoDB user has been added to this project. Not yours? Create one in the [MongoDB Users tab](#).

You'll need your MongoDB user's credentials in the next step.

Close

Choose a connection method

- Clique no botão "Choose a connection method":

Connect to Cluster1

✓ Setup connection security > Choose a connection method > Connect

Choose a connection method [View documentation](#)

Get your pre-formatted connection string by selecting your tool below.



Connect with the `mongo` shell

Interact with your cluster using MongoDB's interactive Javascript interface



Connect your application

Connect your application to your cluster using MongoDB's native drivers



Connect using MongoDB Compass

Explore, modify, and visualize your data with MongoDB's GUI



Go Back

Close

- Selecione a opção "Connect your application":

Connect to Cluster1

✓ Setup connection security > ✓ Choose a connection method > Connect

1 Select your driver and version

DRIVER

Node.js

VERSION

3.6 or later

2 Add your connection string into your application code

☐ Include full driver code example

```
mongodb+srv://betopinheiro:<password>@cluster1.8eduz.mongodb.net/myFirstDatabase?
retryWrites=true&w=majority
```



Replace `<password>` with the password for the `betopinheiro` user. Replace `myFirstDatabase` with the name of the database that connections will use by default. Ensure any option params are [URL encoded](#).

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back

Close

String de conexão:

```
mongodb+srv://betopinheiro:<password>@cluster1.8eduz.mongodb.net/myFirstDatabase?retryWrites=true&w=majority
```

Na string de conexão acima, substitua **<password>** pela **senha** e **myFirstDatabase** por **ecommerce**.

```
mongodb+srv://betopinheiro:abc123456@cluster1.8eduz.mongodb.net/ecommerce?retryWrites=true&w=majority
```

Trabalhando em modo development (local - database: store-api)

index.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

const customersRoutes = require('./routes/customersRoutes');
const productsRoutes = require('./routes/productsRoutes');
const ordersRoutes = require('./routes/ordersRoutes');

const app = express();

// Configurando mongoose
mongoose.Promise = global.Promise;

mongoose.connect('mongodb://localhost/store-api',
{
  useNewUrlParser: true
})
.then(db => console.log('Base de dados conectada!'))
.catch(error => console.log(error));

// Habilitar body-parser
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

// Habilitar cors
app.use(cors());

app.use('/', customersRoutes());
app.use('/', productsRoutes());
app.use('/', ordersRoutes());
app.use(express.static('uploads'));

// server port

app.listen(process.env.PORT || 5000, () => {
  console.log('Servidor web Express em execução!');
})
```

GET - localhost:5000/customers

GET localhost:5000/customers Send 200 OK 100 ms 265 B

Body Auth Query Header Docs Preview Header 8 Cookie Timeline

```
1 [
2 {
3   "_id": "60bc117858f8c976026c8e61",
4   "name": "Roberto",
5   "lastname": "Pinheiro",
6   "email": "roberto_pinheiro@gmail.com",
7   "phone": "(11) 99905-3268"
8 },
9 {
10  "_id": "60bc132058f8c976026c8e62",
11  "name": "Felipe",
12  "lastname": "Lima",
13  "email": "felipe_lima@gmail.com",
14  "phone": "(21) 98839-5723"
15 }
16 ]
```

GET - localhost:5000/products

GET localhost:5000/products Send

```
[
{
  "stock": 25,
  "available": true,
  "_id": "60bc4cd3a512b00c14451ef6",
  "sku": "A11",
  "name": "Ventilador",
  "description": "Ventilador com pedestal de 3 velocidades",
  "price": 149.9,
  "__v": 0,
  "image": "hz5gKTfE0.jpeg"
},
{
  "stock": 45,
  "available": true,
  "_id": "60bc4d9ea512b00c14451ef7",
  "sku": "B12",
  "name": "Liquidificador",
  "description": "Liquidificador com 5 velocidades",
  "price": 99.9,
  "__v": 0,
  "image": "sLMoFOJsa.jpeg"
},
{
  "stock": 6,
  "available": true,
  "_id": "60bcf97c1eb9f1063c4b429d",
  "sku": "E67",
  "name": "Rádio relógio",
  "description": "Rádio relógio digital",
  "price": 78.5,
  "image": "-X04HB4pt.jpeg",
  "__v": 0
},
]
```

```
{
  "stock": 12,
  "available": true,
  "_id": "60bd637a7b6eaf0c040d68e0",
  "sku": "A12",
  "name": "Ventilador",
  "description": "Ventilador de parede",
  "price": 139.5,
  "image": "EYhwgyDwM.jpeg",
  "__v": 0
},
{
  "stock": 10,
  "available": true,
  "_id": "60bd64457b6eaf0c040d68e1",
  "sku": "A13",
  "name": "Ventilador",
  "description": "Ventilador de teto",
  "price": 109.5,
  "image": "EN8o6zoHP.jpeg",
  "__v": 0
}
]
```

GET - localhost:5000/orders

GET localhost:5000/orders

Send

```
[
  {
    "_id": "60bd22b64aac221b18baf9eb",
    "customer": {
      "_id": "60bc117858f8c976026c8e61",
      "name": "Roberto",
      "lastname": "Pinheiro",
      "email": "roberto_pinheiro@gmail.com",
      "phone": "(11) 99905-3268"
    },
    "products": [
      {
        "_id": "60bd22b64aac221b18baf9ec",
        "product": {
          "stock": 25,
          "available": true,
          "_id": "60bc4cd3a512b00c14451ef6",
          "sku": "A11",
          "name": "Ventilador",
          "description": "Ventilador com pedestal de 3 velocidades",
          "price": 149.9,
          "__v": 0,
          "image": "hz5gKTFE0.jpeg"
        },
        "unitPrice": 149.9,
        "quantity": 1,
        "amount": 149.9
      },
      {
        "_id": "60bd22b64aac221b18baf9ed",
        "product": {
          "stock": 45,
          "available": true,
          "_id": "60bc4d9ea512b00c14451ef7",
          "sku": "B12",
          "name": "Liquidificador",
          "description": "Liquidificador com 5 velocidades",
          "price": 99.9,
          "__v": 0,
          "image": "slMoFO3sa.jpeg"
        },
        "unitPrice": 99.9,
        "quantity": 2,
        "amount": 199.8
      }
    ],
    "totalAmount": 349.7,
    "created": "2021-06-06T19:32:06.132Z",
    "__v": 0
  },
  {
    "_id": "60bd5b4a1396e037288e9bdb",
    "customer": {
      "_id": "60bc132058f8c976026c8e62",
      "name": "Felipe",
      "lastname": "Lima",
      "email": "felipe_lima@gmail.com",
      "phone": "(21) 98839-5723"
    },
    "products": [
      {
        "_id": "60bd5b4a1396e037288e9bdc",
        "product": {
          "stock": 6,
          "available": true,
          "_id": "60bcf97c1eb9f1063c4b429d",
          "sku": "E67",
          "name": "Rádio relógio",
          "description": "Rádio relógio digital",
          "price": 78.5,
          "image": "-X04H84pt.jpeg",
          "__v": 0
        },
        "unitPrice": 78.5,
        "quantity": 1,
        "amount": 78.5
      }
    ],
    "totalAmount": 78.5,
    "created": "2021-06-06T23:33:30.802Z",
    "__v": 0
  }
]
```

```
{
  "_id": "60bd5ca41396e037288e9bdd",
  "customer": {
    "_id": "60bc132058f8c976026c8e62",
    "name": "Felipe",
    "lastname": "Lima",
    "email": "felipe_lima@gmail.com",
    "phone": "(21) 98839-5723"
  },
  "products": [
    {
      "_id": "60bd5ca41396e037288e9bde",
      "product": {
        "stock": 45,
        "available": true,
        "_id": "60bc4d9ea512b00c14451ef7",
        "sku": "B12",
        "name": "Liquidificador",
        "description": "Liquidificador com 5 velocidades",
        "price": 99.9,
        "__v": 0,
        "image": "slMoF0Jsa.jpeg"
      },
      "unitPrice": 99.9,
      "quantity": 1,
      "amount": 99.9
    }
  ],
  "totalAmount": 99.9,
  "created": "2021-06-06T23:39:16.973Z",
  "__v": 0
}
```

Local

4 DBS

4 COLLECTIONS

HOST

localhost:27017

CLUSTER

Standalone

EDITION

MongoDB 4.4.6 Community

Filter your data

admin

config

local

store-api

customers

orders

products

Collections

CREATE COLLECTION

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
customers	2	123.0 B	246.0 B	2	72.0 KB	
orders	3	232.7 B	698.0 B	1	36.0 KB	
products	5	175.4 B	877.0 B	2	72.0 KB	

Trabalhando em modo development (local - database: ecommerce)

Alterações no arquivo index.js

index.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

const customersRoutes = require('./routes/customersRoutes');
const productsRoutes = require('./routes/productsRoutes');
const ordersRoutes = require('./routes/ordersRoutes');

const app = express();

mongoose.connect("mongodb+srv://betopinheiro:abc123456@cluster1.8eduz.mongodb.net/ecommerce?retryWrites=true&w=majority",
{
  useNewUrlParser: true
})
.then(db => console.log('Base de dados conectada!'))
.catch(error => console.log(error));

// Habilitar body-parser
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

// Habilitar cors
app.use(cors());

app.use('/', customersRoutes());
app.use('/', productsRoutes());
app.use('/', ordersRoutes());
app.use(express.static('uploads'));

// server port

app.listen(process.env.PORT || 5000, () => {
  console.log('Servidor web Express em execução!');
})
```

- Rode o servidor do Mongo:

mongod

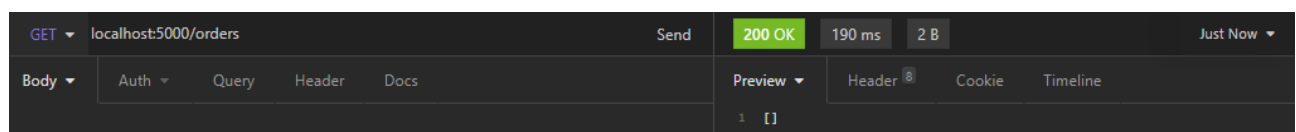
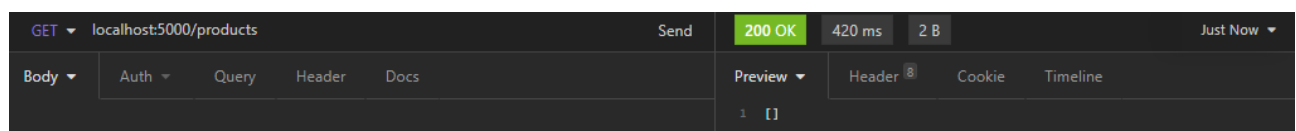
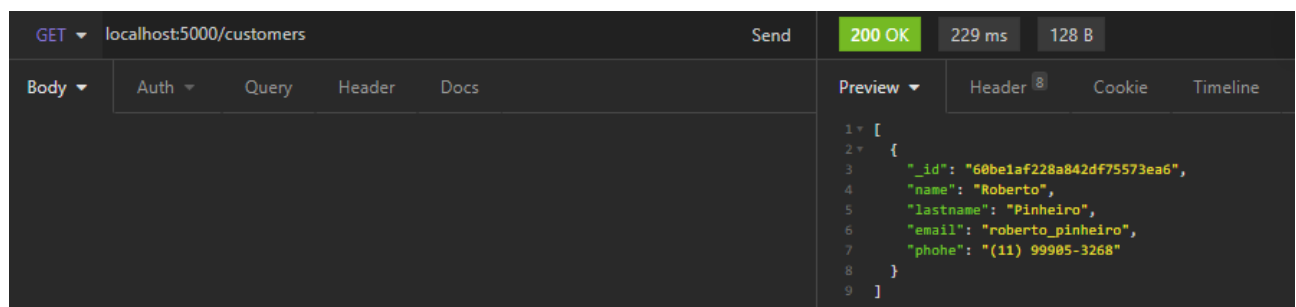
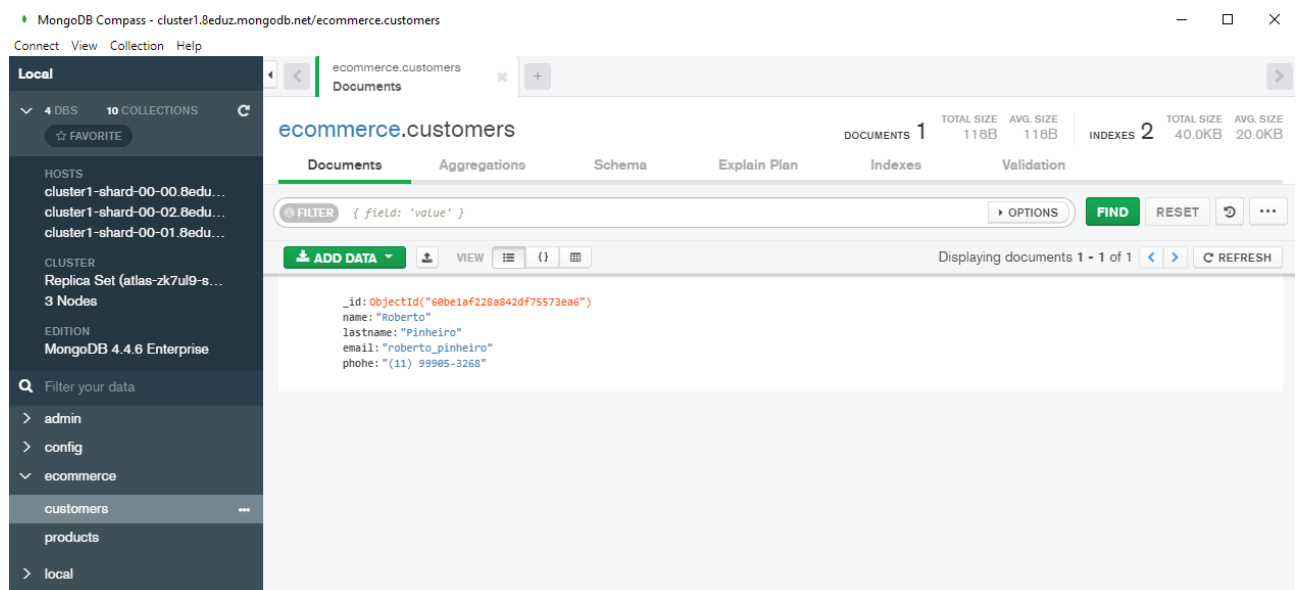
- Rode o servidor da aplicação:

`npm run dev`

```
[nodemon] restarting due to changes...
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
Servidor web Express em execução!
(node:9808) [MONGODB DRIVER] Warning: Current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version. To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
(node:9808) DeprecationWarning: collection.ensureIndex is deprecated. Use createIndexes instead.
Base de dados conectada!
```

- Crie uma nova conexão no MongoDB Compass, com a string de conexão:

`mongodb+srv://betopinheiro:abc123456@cluster1.8eduz.mongodb.net/ecommerce?retryWrites=true&w=majority`



Cadastrando clientes

```
{
  "name": "Roberto",
  "lastname": "Pinheiro",
  "email": "roberto_pinheiro@gmail.com",
  "phone": "(11) 99905-3268"
}
```

A screenshot of a REST client interface. The top bar shows a POST request to localhost:5000/customers. The status bar indicates a 400 Bad Request with a response time of 218 ms and a body size of 75 B. The JSON tab on the left shows the request body: { "name": "Roberto", "lastname": "Pinheiro", "email": "roberto_pinheiro@gmail.com", "phone": "(11) 99905-3268" }. The Preview tab on the right shows the response body: { "message": "Já existe um cliente com o email: roberto_pinheiro@gmail.com" }.

```
{
  "name": "Felipe",
  "lastname": "Lima",
  "email": "felipe_lima@gmail.com",
  "phone": "(21) 98839-5723"
}
```

A screenshot of a REST client interface. The top bar shows a POST request to localhost:5000/customers. The status bar indicates a 201 Created response with a response time of 196 ms and a body size of 45 B. The JSON tab on the left shows the request body: { "name": "Felipe", "lastname": "Lima", "email": "felipe_lima@gmail.com", "phone": "(21) 98839-5723" }. The Preview tab on the right shows the response body: { "message": "Cliente cadastrado com sucesso!" }.

```
{
  "name": "Juan",
  "lastname": "Molina López",
  "email": "juan_molina@hotmail.com",
  "phone": "(19) 99905-3268"
}
```

A screenshot of a REST client interface. The top bar shows a POST request to localhost:5000/customers. The status bar indicates a 201 Created response with a response time of 166 ms and a body size of 45 B. The JSON tab on the left shows the request body: { "name": "Juan", "lastname": "Molina López", "email": "juan_molina@hotmail.com", "phone": "(19) 99905-3268" }. The Preview tab on the right shows the response body: { "message": "Cliente cadastrado com sucesso!" }.

GET localhost:5000/customers Send 200 OK 266 ms 425 B

Body Auth Query Header Docs

Select a body type from above

Preview Header 8 Cookie Timeline

```

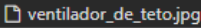
1 [
2   {
3     "_id": "60be2a75bfda642650d01770",
4     "name": "Roberto",
5     "lastname": "Pinheiro",
6     "email": "roberto_pinheiro@gmail.com",
7     "phone": "(11) 99905-3268",
8     "__v": 0
9   },
10  {
11    "_id": "60be2b46bfda642650d01772",
12    "name": "Felipe",
13    "lastname": "Lima",
14    "email": "felipe_lima@gmail.com",
15    "phone": "(21) 98839-5723",
16    "__v": 0
17  },
18  {
19    "_id": "60be2b75bfda642650d01773",
20    "name": "Juan",
21    "lastname": "Molina López",
22    "email": "juan_molina@hotmail.com",
23    "phone": "(19) 99905-3268",
24    "__v": 0
25  }
26 ]

```

Cadastrando produtos

POST localhost:5000/products/ Send 201 Created 382 ms 45 B

Multipart 8 Auth Query Header 1 Docs

sku	A13	
name	Ventilador	
description	Ventilador de teto	
image		
price	109.50	
stock	10	
available	true	

Preview Header 8 Cookie Timeline

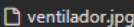
```

1 {
2   "message": "Produto cadastrado com sucesso!"
3 }

```

POST localhost:5000/products/ Send 201 Created 230 ms 45 B

Multipart 8 Auth Query Header 1 Docs

sku	A11	
name	Ventilador	
description	Ventilador com pedestal	
image		
price	149.50	
stock	5	
available	true	

Preview Header 8 Cookie Timeline

```

1 {
2   "message": "Produto cadastrado com sucesso!"
3 }

```

GET localhost:5000/products Send 200 OK 181 ms 359 B

Body Auth Query Header Docs Preview Header 8 Cookie Timeline

Select a body type from above

```
1 [
2   {
3     "stock": 10,
4     "available": true,
5     "_id": "60be418d6a20112a18941766",
6     "sku": "A13",
7     "name": "Ventilador",
8     "description": "Ventilador de teto",
9     "price": 109.5,
10    "image": "Zj2RxZLRM.jpeg",
11    "__v": 0
12  },
13  {
14    "stock": 5,
15    "available": true,
16    "_id": "60be42046a20112a18941767",
17    "sku": "A11",
18    "name": "Ventilador",
19    "description": "Ventilador com pedestal",
20    "price": 149.5,
21    "image": "XrSqlUucl.jpeg",
22    "__v": 0
23  }
24 ]
```

localhost:5000/customers

Apps Seletores de Formu... jQuery Validation

```
1 // 20210607132301
2 // http://localhost:5000/customers
3
4 [
5   {
6     "_id": "60be2a75bfda642650d01770",
7     "name": "Roberto",
8     "lastname": "Pinheiro",
9     "email": "roberto_pinheiro@gmail.com",
10    "phone": "(11) 99905-3268",
11    "__v": 0
12  },
13  {
14    "_id": "60be2b46bfda642650d01772",
15    "name": "Felipe",
16    "lastname": "Lima",
17    "email": "felipe_lima@gmail.com",
18    "phone": "(21) 98839-5723",
19    "__v": 0
20  },
21  {
22    "_id": "60be2b75bfda642650d01773",
23    "name": "Juan",
24    "lastname": "Molina López",
25    "email": "juan_molina@hotmail.com",
26    "phone": "(19) 99905-3268",
27    "__v": 0
28  }
29 ]
```



```
1 // 20210607132423
2 // http://localhost:5000/products
3
4 [
5   {
6     "stock": 10,
7     "available": true,
8     "_id": "60be418d6a20112a18941766",
9     "sku": "A13",
10    "name": "Ventilador",
11    "description": "Ventilador de teto",
12    "price": 109.5,
13    "image": "Zj2RxZLRM.jpeg",
14    "__v": 0
15  },
16  {
17    "stock": 5,
18    "available": true,
19    "_id": "60be42046a20112a18941767",
20    "sku": "A11",
21    "name": "Ventilador",
22    "description": "Ventilador com pedestal",
23    "price": 149.5,
24    "image": "XrSqlUuc1.jpeg",
25    "__v": 0
26  }
27 ]
```

Instalando o pacote dotenv

npm install --save dotenv

Deploy

- Adicione a pasta raiz do projeto um arquivo chamado **".gitignore"**:

index.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

// Rotas

const customersRoutes = require('./routes/customersRoutes');
const productsRoutes = require('./routes/productsRoutes');
const ordersRoutes = require('./routes/ordersRoutes');

const app = express();

app.get('/', (req, res) => {
  res.send('Página inicial');
});

// Conexão com o mongoDB

// Configurando mongoose
mongoose.Promise = global.Promise;

require('dotenv').config();

// Conexão a base de dados
const uri = process.env.MONGODB_URI || 'mongodb+srv://$${process.env.USER}:$${process.env.PASSWORD}@cluster1.8eduz.mongodb.net/$${process.env.DBNAME}?retryWrites=true&w=majority';

mongoose.connect(uri, {useNewUrlParser: true, useUnifiedTopology: true})
.then(() => {
  console.log('Base de dados conectada!');
}).catch((error) => {
  console.log(error);
});

// Habilitar body-parser
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));

// Habilitar cors
app.use(cors());

app.use('/', customersRoutes());
app.use('/', productsRoutes());
app.use('/', ordersRoutes());
app.use(express.static('uploads'));
```

```
// server port
```

```
const port = process.env.PORT || 5000;
```

```
app.listen(port, () => {  
  console.log(`Server running at port ` + port);  
});
```

.env

```
USER=betopinheiro  
PASSWORD=abc123456  
DBNAME=ecommerce  
URL=https://api-store-felipemx.herokuapp.com/
```









.gitignore

```
/node_modules  
.env
```

Variáveis de ambiente no Heroku

Config Vars

Hide Config Vars

DBNAME	ecommerce	 
PASSWORD	abc123456	 
URL	https://api-store-felipemx.herokuapp.com/	 
USER	betopinheiro	 

Install the Heroku CLI

Download and install the [Heroku CLI](#).

If you haven't already, log in to your Heroku account and follow the prompts to create a new SSH public key.

```
$ heroku login
```

Create a new Git repository

Initialize a git repository in a new or existing directory

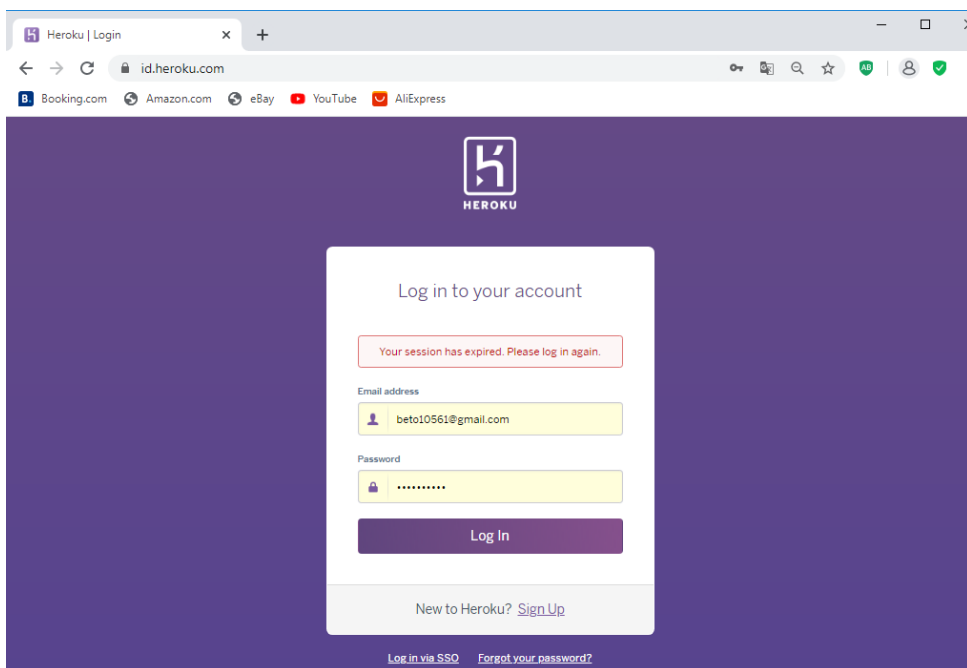
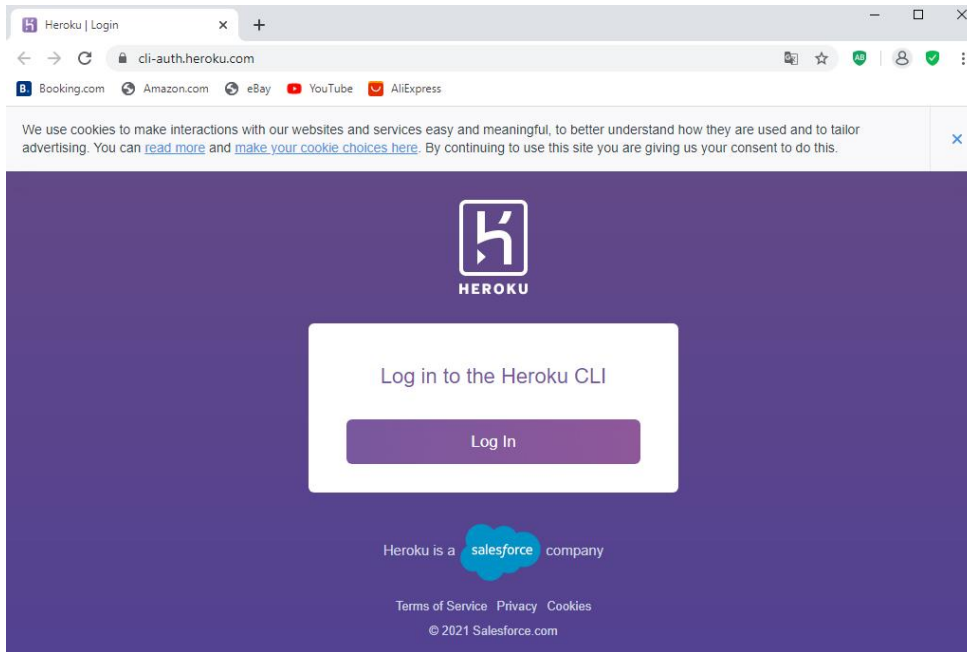
```
$ cd my-project/  
$ git init  
$ heroku git:remote -a api-store-felipemx
```

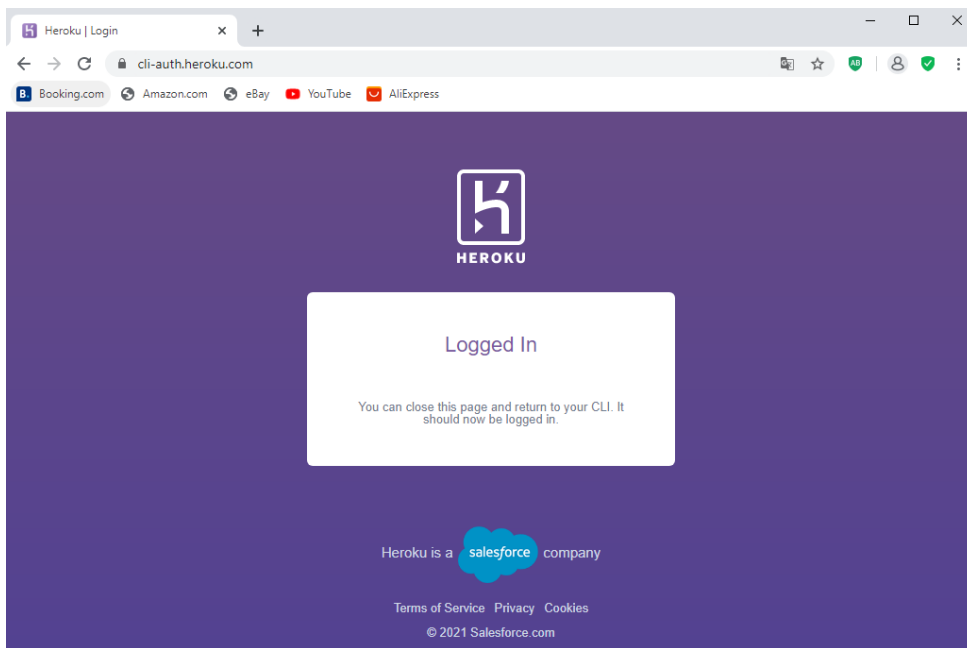
Deploy your application

Commit your code to the repository and deploy it to Heroku using Git.

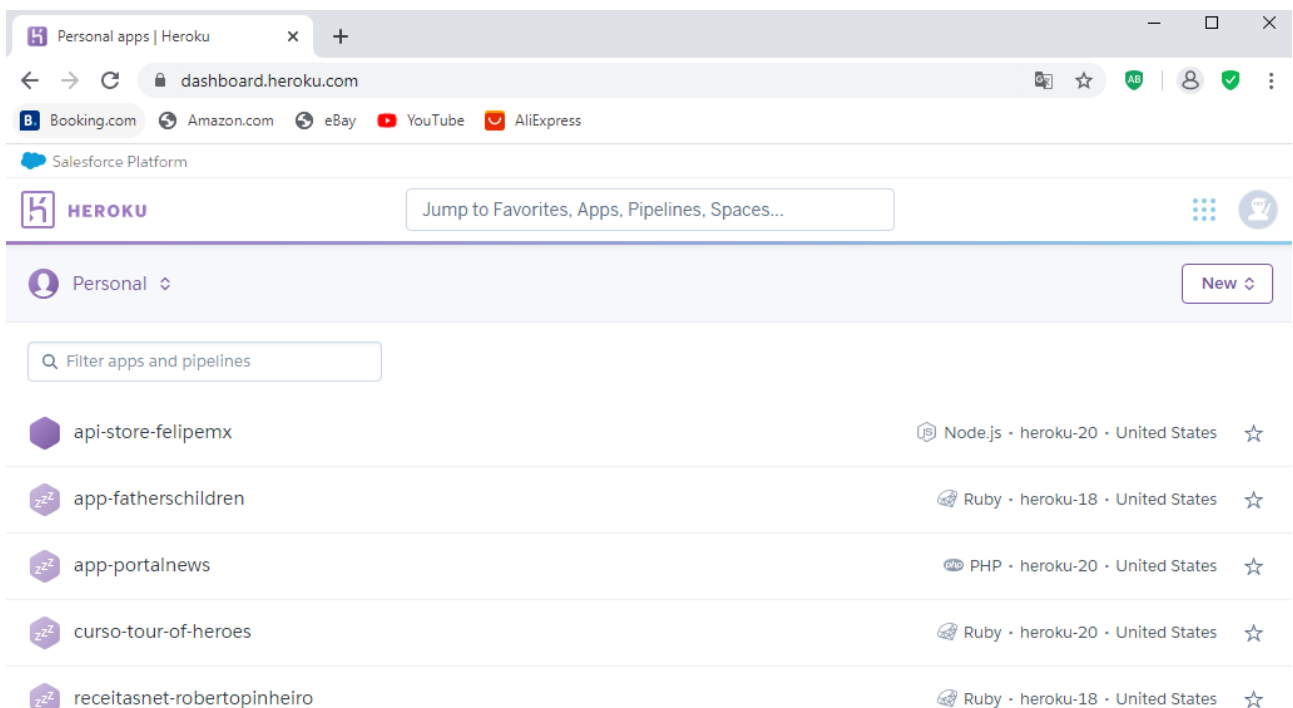
```
$ git add .  
$ git commit -am "make it better"  
$ git push heroku master
```

heroku login





```
beto1@DESKTOP-NUIKR9E MINGW64 /c/felipemx/store-api
$ heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/a2bfb706-d496-4c13-91b5-e85e0c4207b7?requestor=SFMyNTY.g2gDbQAAA8xODkuMTIwLjE0Ny4yMDEuBgCnL-TmeQFiAAFRgA.sGVNOCvPFLSo6ypJgokhNgJz28pcXE53oCnv6djNV90
Logging in... done
Logged in as beto10561@gmail.com
```



Install the Heroku CLI

Download and install the [Heroku CLI](#).

If you haven't already, log in to your Heroku account and follow the prompts to create a new SSH public key.

```
$ heroku login
```

Create a new Git repository

Initialize a git repository in a new or existing directory

```
$ cd my-project/  
$ git init  
$ heroku git:remote -a api-store-felipemx
```

Deploy your application

Commit your code to the repository and deploy it to Heroku using Git.

```
$ git add .  
$ git commit -am "make it better"  
$ git push heroku master
```

git init

```
beto1@DESKTOP-NUIKR9E MINGW64 /c/felipemx/store-api  
$ git init  
Initialized empty Git repository in C:/felipemx/store-api/.git/
```

heroku git:remote -a api-store-felipemx

```
beto1@DESKTOP-NUIKR9E MINGW64 /c/felipemx/store-api (master)  
$ heroku git:remote -a api-store-felipemx  
set git remote heroku to https://git.heroku.com/api-store-felipemx.git
```

git status

```
beto1@DESKTOP-NUIKR9E MINGW64 /c/felipemx/store-api (master)  
$ git status  
On branch master  
  
No commits yet  
  
Untracked files:  
  (use "git add <file>..." to include in what will be committed)  
    .gitignore  
    controllers/  
    index.js  
    models/  
    package-lock.json  
    package.json  
    routes/  
    utils/  
  
nothing added to commit but untracked files present (use "git add" to track)
```


git add .

```
beto1@DESKTOP-NUIKR9E MINGW64 /c/felipemx/store-api (master)
$ git add .
warning: LF will be replaced by CRLF in package-lock.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in package.json.
The file will have its original line endings in your working directory
```

git status

```
beto1@DESKTOP-NUIKR9E MINGW64 /c/felipemx/store-api (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   .gitignore
    new file:   controllers/customersController.js
    new file:   controllers/ordersController.js
    new file:   controllers/productsController.js
    new file:   index.js
    new file:   models/Custom.js
    new file:   models/Order.js
    new file:   models/Product.js
    new file:   package-lock.json
    new file:   package.json
    new file:   routes/customersRoutes.js
    new file:   routes/ordersRoutes.js
    new file:   routes/productsRoutes.js
    new file:   utils/multerConfig.js
```

git commit -m "First commit"

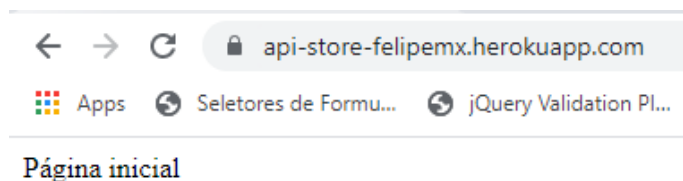
```
beto1@DESKTOP-NUIKR9E MINGW64 /c/felipemx/store-api (master)
$ git commit -m "First commit"
[master (root-commit) f3adef1] First commit
14 files changed, 1364 insertions(+)
create mode 100644 .gitignore
create mode 100644 controllers/customersController.js
create mode 100644 controllers/ordersController.js
create mode 100644 controllers/productsController.js
create mode 100644 index.js
create mode 100644 models/Custom.js
create mode 100644 models/Order.js
create mode 100644 models/Product.js
create mode 100644 package-lock.json
create mode 100644 package.json
create mode 100644 routes/customersRoutes.js
create mode 100644 routes/ordersRoutes.js
create mode 100644 routes/productsRoutes.js
create mode 100644 utils/multerConfig.js
```

git push heroku master

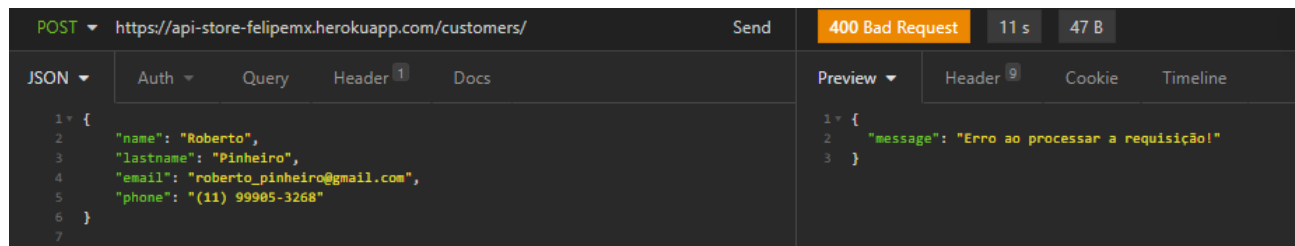
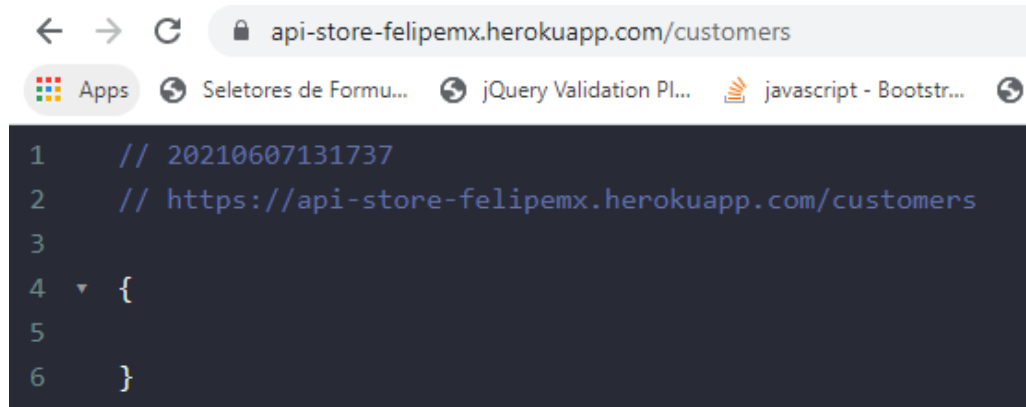
```
betoi@DESKTOP-NUIKR9E MINGW64 /c/felipemx/store-api (master)
$ git push heroku master
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 2 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (20/20), 13.52 KiB | 865.00 KiB/s, done.
Total 20 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Building on the Heroku-20 stack
remote: -----> Determining which buildpack to use for this app
remote: -----> Node.js app detected
remote:
remote: -----> Creating runtime environment
remote:
remote:       NPM_CONFIG_LOGLEVEL=error
remote:       NODE_VERBOSE=false
remote:       NODE_ENV=production
remote:       NODE_MODULES_CACHE=true
```

```
remote: -----> Pruning devDependencies
remote:       audited 101 packages in 1.12s
remote:
remote:       2 packages are looking for funding
remote:       run `npm fund` for details
remote:
remote:       found 0 vulnerabilities
remote:
remote: -----> Build succeeded!
remote: -----> Discovering process types
remote:       Procfile declares types   -> (none)
remote:       Default types for buildpack -> web
remote:
remote: -----> Compressing...
remote:       Done: 34.8M
remote: -----> Launching...
remote:       Released v3
remote:       https://api-store-felipemx.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/api-store-felipemx.git
 * [new branch]      master -> master
```

<https://api-store-felipemx.herokuapp.com/>



https://api-store-felipemx.herokuapp.com/customers



??????????