

DESAFIO MAIDA Lanchonete

Você pode usar Postman ou Insomnia no meu caso usei Insomnia.

Desafio Maida Lanchonete

Projeto desenvolvido em Spring Boot

Java 17

Maven

Banco de dados PostgreSQL

Swagger 2.9.2

Docker

Para levantar o projeto

Deve ter o Java instalado na maquina, assim como Maven

Clone o projeto do repositório Git <https://github.com/krosscaal/maida-challenge-developing-snack-bar>

Na sua máquina usando o terminal entre na pasta do projeto criada pelo repositório deve ser parecida com /maida-challenge-developing-snack-bar

execute o comando do docker

```
sudo docker-compose -f docker-compose-dev.yml up -d
```

retire sudo se não estiver rodando em alguma distro linux

Assim será instalado o container do postgresSQL v14

Logo e rodar o projeto que o FlyWay irá rodar as migrations instalando as tabelas necesarias no banco

Faço notar que no pacote config especificamente da classe SwaggerConfig estão comentadas as anotações de configuração devido ao um bug que encontrei ao realiaar uma listagem de um endpoint listPedidosFinalizadosCliente, nos outros endpoint o swagger funciona bem.

Você pode testar usando o Postman ou Insomnia que é meu caso que todos os endpoint funcionam normalmente inclusive o citado.

A tabela products já tem alguns produtos persistidos.

Lista de endpoints e finalidade:

CLIENTE-CONTROLLER

Adicionar cliente

| | |
|------|---------------------------------------|
| POST | localhost:8080/snack-bar/costumer/new |
|------|---------------------------------------|

Atualizar cliente

| | |
|-----|---|
| PUT | localhost:8080/snack-bar/costumer/update/{id} |
|-----|---|

Busca cliente para visualização do cliente

| | |
|-----|--|
| GET | localhost:8080/snack-bar/costumer/v2/2 |
|-----|--|

Busca cliente para visualização do gestor

| | |
|-----|-------------------------------------|
| GET | localhost:8080/snack-bar/costumer/1 |
|-----|-------------------------------------|

Lista clientes para visualização do gestor

| | |
|-----|--|
| GET | localhost:8080/snack-bar/costumer/list |
|-----|--|

PRODUTO-CONTROLLER

Adicionar produto

| | |
|------|---------------------------------------|
| POST | localhost:8080/snack-bar/products/new |
|------|---------------------------------------|

Atualizar produto

| | |
|-----|---|
| PUT | localhost:8080/snack-bar/products/update/{id} |
|-----|---|

Lista produtos

| | |
|-----|--|
| GET | localhost:8080/snack-bar/products/list |
|-----|--|

Lista de produtos para visualização do cliente

| | |
|-----|---|
| GET | localhost:8080/snack-bar/products/list-v2 |
|-----|---|

Busca Produto por id

| | |
|-----|--|
| GET | localhost:8080/snack-bar/products/{id} |
|-----|--|

Busca Produto por id para visualização do cliente

| | |
|-----|---|
| GET | localhost:8080/snack-bar/products/v2/{id} |
|-----|---|

Remover produto

| | |
|-----|--|
| DEL | localhost:8080/snack-bar/products/{id} |
|-----|--|

GESTOR-CONTROLLER

Adicionar gestor

| | |
|------|------------------------------|
| POST | localhost:8080/snack-bar/new |
|------|------------------------------|

Atualizar gestor

| | |
|-----|---------------------------------|
| PUT | localhost:8080/snack-bar/update |
|-----|---------------------------------|

Busca gestor

| | |
|-----|----------------------------------|
| GET | localhost:8080/snack-bar/manager |
|-----|----------------------------------|

PEDIDO-CONTROLLER

Adicionar pedido

| | |
|------|-------------------------------------|
| POST | localhost:8080/snack-bar/orders/new |
|------|-------------------------------------|

Update status pedido pelo gestor

| | |
|-----|--|
| PUT | localhost:8080/snack-bar/orders/update-status/{id} |
|-----|--|

Update status pedido para cancelado pelo cliente

| | |
|-----|--|
| PUT | localhost:8080/snack-bar/orders/order-cancel/1 |
|-----|--|

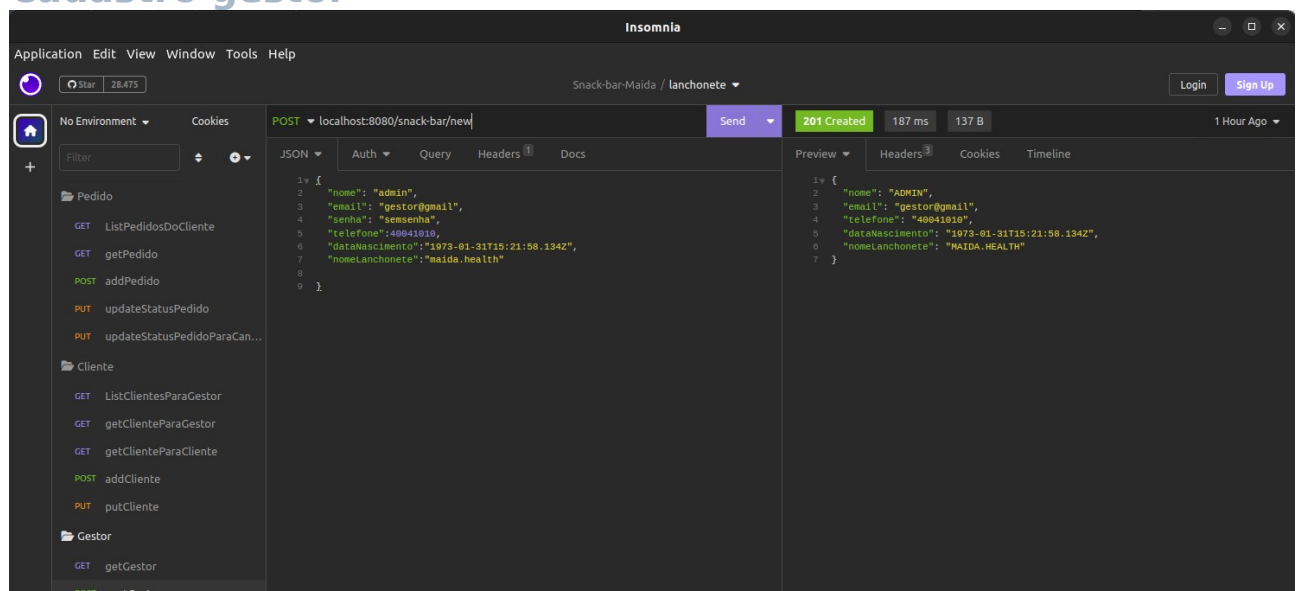
Busca pedido pelo id

| | |
|-----|---|
| GET | localhost:8080/snack-bar/orders/status/{id} |
|-----|---|

Lista de pedidos finalizados pelo cliente (com status recusado, cancelado, entregue)

| | |
|-----|---|
| GET | localhost:8080/snack-bar/orders/finished-orders/1 |
|-----|---|

Cadastro gestor



The screenshot shows the Insomnia API client interface. The main window displays a POST request to `localhost:8080/snack-bar/new` with a status of `201 Created`. The request body is a JSON object:

```
1 {
2   "nome": "admin",
3   "email": "gestor@gmail",
4   "senha": "sensenha",
5   "telefone": "40041910",
6   "dataNascimento": "1973-01-31T15:21:58.134Z",
7   "nomeLanchonete": "Maida.health"
8 }
9
```

The response body is a JSON object:

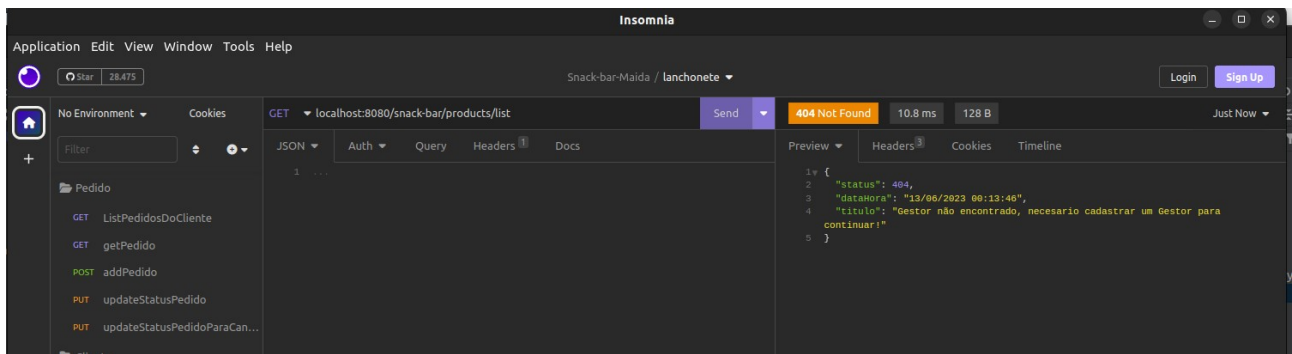
```
1 {
2   "nome": "ADMIN",
3   "email": "gestor@gmail",
4   "telefone": "40041910",
5   "dataNascimento": "1973-01-31T15:21:58.134Z",
6   "nomeLanchonete": "MAIDA.HEALTH"
7 }
```

The left sidebar shows a list of endpoints under the 'Pedido' folder:

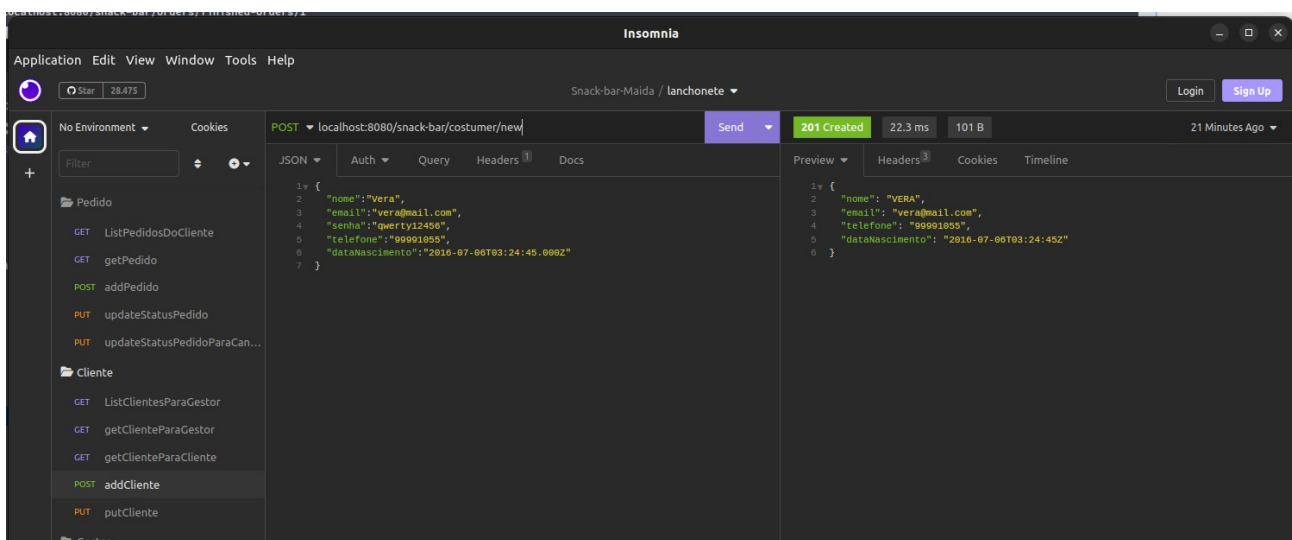
- GET ListPedidosDoCliente
- GET getPedido
- POST addPedido
- PUT updateStatusPedido
- PUT updateStatusPedidoParaCan...

The top right corner of the interface shows a 'Login' button and a 'Sign Up' button.

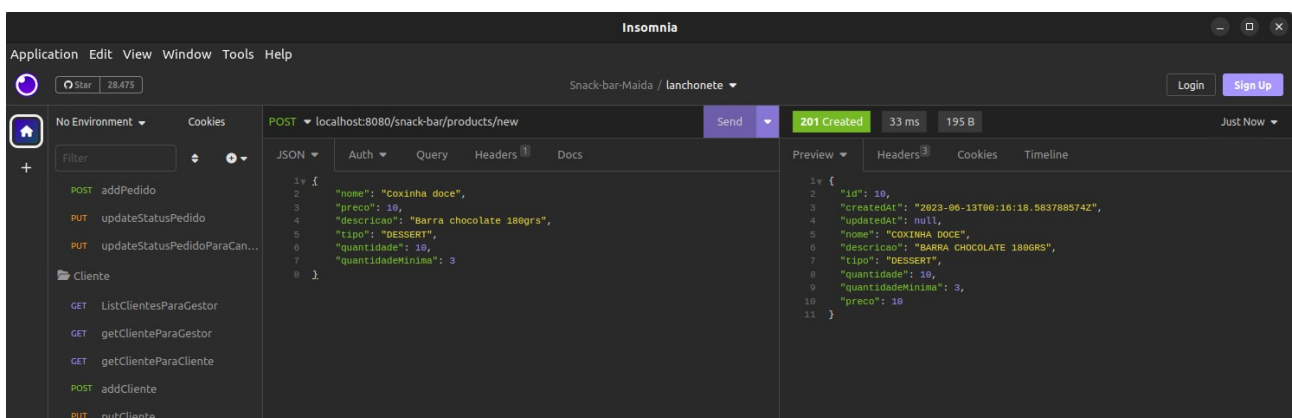
Se não tiver gestor cadastrado response 404



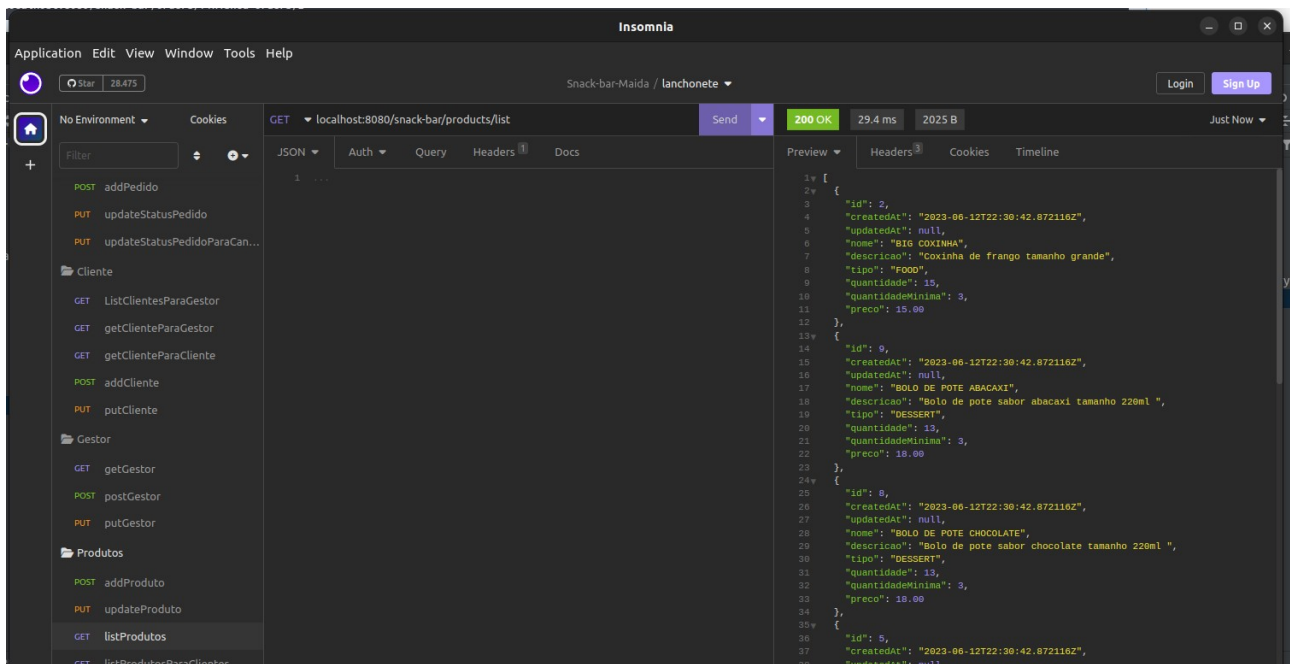
Cadastrar cliente



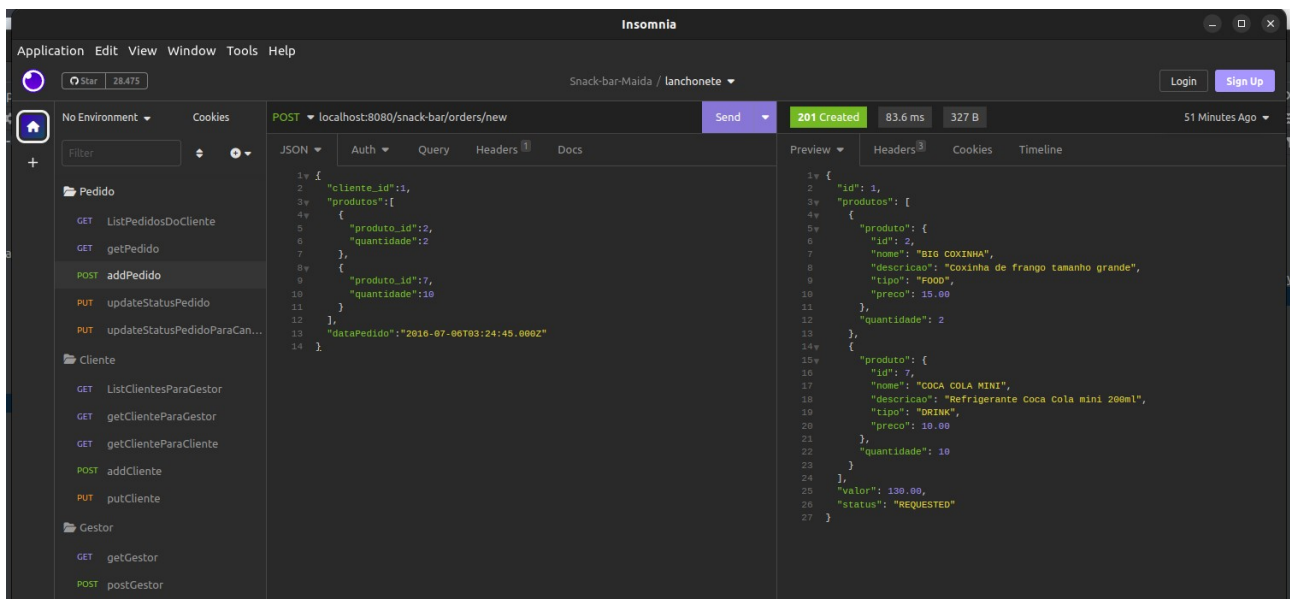
Cadastrar produto



Lista de produtos



Cadastrar pedido



lista para cliente de pedidos finalizados(com status delivered, canceled, recused)

The screenshot shows the Insomnia API client interface. The top bar includes the application name 'Insomnia' and standard window controls. Below the bar, the 'Application' menu is visible, and the current environment is 'No Environment' with 28,475 items. The main workspace is divided into three panes: a left sidebar with a collection of API endpoints, a central request editor, and a right pane for the response.

Left Sidebar (API Collections):

- Pedido**
 - GET ListPedidosDoCliente
 - GET getPedido
 - POST addPedido
 - PUT updateStatusPedido
 - PUT updateStatusPedidoParaCan...
- Cliente**
 - GET ListClientesParaGestor
 - GET getClienteParaGestor
 - GET getClienteParaCliente
 - POST addCliente
 - PUT putCliente
- Gestor**
 - GET getGestor
 - POST postGestor
 - PUT putGestor
- Produtos**
 - POST addProduto
 - PUT updateProduto
 - GET listProdutos

Central Request Editor:

The request is a GET method to the endpoint `localhost:8080/snack-bar/orders/finished-orders/1`. The status bar shows a **200 OK** response with a time of 21 ms and a body size of 327 B. The response is received 'Just Now'.

The 'Body' tab is selected, showing a large bug icon and the text: 'Enter a URL and send to get a response'. Below this, it says 'Select a body type from above to send data in the body of a request' and 'Introduction to Insomnia'.

Right Pane (Response):

The 'Preview' tab is selected, displaying the JSON response:

```
1 [
2   {
3     "id": 1,
4     "produtos": [
5       {
6         "produto": {
7           "id": 2,
8           "nome": "BIG COXINHA",
9           "descricao": "Coxinha de frango tamanho grande",
10          "tipo": "FOOD",
11          "preco": 15.00
12        },
13        "quantidade": 2
14      },
15      {
16        "produto": {
17          "id": 7,
18          "nome": "COCA COLA MINI",
19          "descricao": "Refrigerante Coca Cola mini 200ml",
20          "tipo": "DRINK",
21          "preco": 10.00
22        },
23        "quantidade": 10
24      }
25    ],
26    "valor": 130.00,
27    "status": "RECUSED"
28  }
29 ]
```