

DOCUMENTAÇÃO DE USO

API FAST FOOD

Dev: Tiago Pereira

Data: 04/05/2022

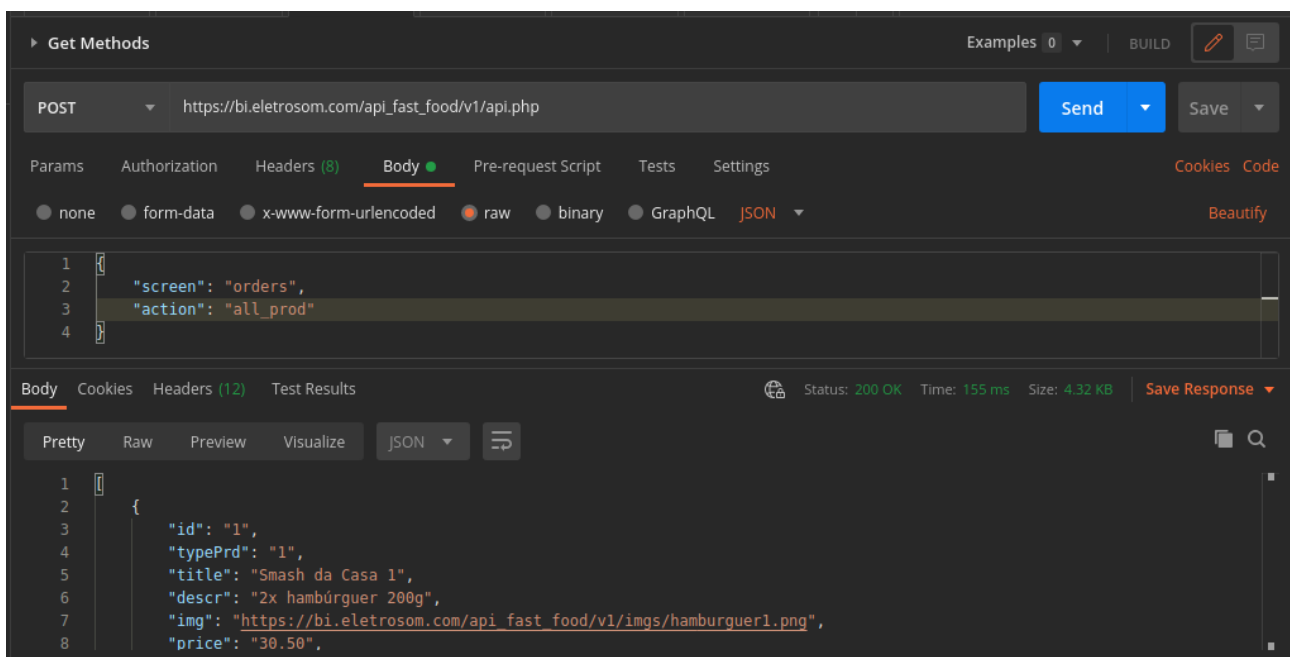
Para começar usar basta realizar as requisições no end point:

https://bi.eletrosom.com/api_fast_food/v1/api.php

exemplos:

Produtos / Pedidos

→ **all_prod:** (Todos os Produtos)



→ **all_payment_methods** (Todos os métodos de pagamento)

The screenshot shows a REST client interface with the following details:

- Request:** Method `GET`, URL `http://localhost:3000/api/orders/all_payment_methods`. Headers include `Accept: application/json` and `Content-Type: application/json`.
- Response:** Status `200 OK`, Time `130 ms`, Size `507 B`. The response body is a JSON array of two objects, each representing a payment method with an `id` and `typePay`.

```
1 {
2   "screen": "orders",
3   "action": "all_payment_methods"
4 }
```

```
1 {
2   {
3     "id": "1",
4     "typePay": "Débido"
5   },
6   {
7     "id": "2",
8     "typePay": "Crédito"
9   }
10 }
```

→ **all_smashes** (Todos os smashes)

The screenshot shows a REST client interface with the following details:

- Request:** Method `GET`, URL `http://localhost:3000/api/orders/all_smashes`. Headers include `Accept: application/json` and `Content-Type: application/json`.
- Response:** Status `200 OK`, Time `29 ms`, Size `2.28 KB`. The response body is a JSON array of one object representing a smash with various attributes including `id`, `typePrd`, `title`, `descr`, `img`, and `price`.

```
1 {
2   "screen": "orders",
3   "action": "all_smashes"
4 }
```

```
1 {
2   {
3     "id": "1",
4     "typePrd": "1",
5     "title": "Smash da Casa 1",
6     "descr": "2x hambúrguer 200g",
7     "img": "https://bi.eletrosom.com/api_fast_food/v1/imgs/hamburguer1.png",
8     "price": "30.50",
9   }
10 }
```

→ **all_drinks** (Todas as bebidas)

The screenshot shows a REST client interface with the following details:

- Request:** Method `GET`, URL `http://localhost:3000/api/orders/all_drinks`. Headers include `Accept: application/json` and `Content-Type: application/json`.
- Response:** Status `200 OK`, Time `28 ms`, Size `1 KB`. The response body is a JSON array of one object representing a drink with attributes including `id`, `typePrd`, `title`, `descr`, `img`, and `price`.

```
1 {
2   "screen": "orders",
3   "action": "all_drinks"
4 }
```

```
1 {
2   {
3     "id": "3",
4     "typePrd": "Bebida",
5     "title": "Coca-Cola 200 ml",
6     "descr": "Coca-Cola 200 ml",
7     "img": null,
8     "price": "7.50",
9   }
10 }
```

→ **all_combo** (Todos os combos)

The screenshot shows a REST client interface with the following details:

- Request:** Method `GET`, URL `http://localhost:3000/api/orders/all_combo`, Headers `Accept: application/json`.
- Response:** Status `200 OK`, Time `35 ms`, Size `1.04 KB`. The response body is a JSON array containing one object for a combo item.
- JSON Body:**

```
[{"id": "2", "typePrd": "Combo", "title": "Combo Smash 1", "descr": "2x Smash da Casa 1", "img": null, "price": "33.00"}]
```

→ **all_accompaniment** (Todos os acompanhamentos)

The screenshot shows a REST client interface with the following details:

- Request:** Method `GET`, URL `http://localhost:3000/api/orders/all_accompaniment`, Headers `Accept: application/json`.
- Response:** Status `200 OK`, Time `239 ms`, Size `788 B`. The response body is a JSON array containing one object for an accompaniment item.
- JSON Body:**

```
[{"id": "4", "typePrd": "Acompanhamento", "title": "Batata Frita", "descr": "Batata Frita 600g", "img": null, "price": "17.50"}]
```

→ **all_dessert** (Todos as sobremesas)

The screenshot shows a REST client interface with the following details:

- Request:** Method `GET`, URL `http://localhost:3000/api/orders/all_dessert`, Headers `Content-Type: application/json`, Body `{ "screen": "orders", "action": "all_dessert" }`.
- Response:** Status `200 OK`, Time `239 ms`, Size `788 B`. The response body is a JSON array with one object:

```
[{"id": "4", "typePrd": "Acompanhamento", "title": "Batata Frita", "descr": "Batata Frita 600g", "img": null, "price": "17.50"}]
```

→ **all_additional** (Todos os adicionais)

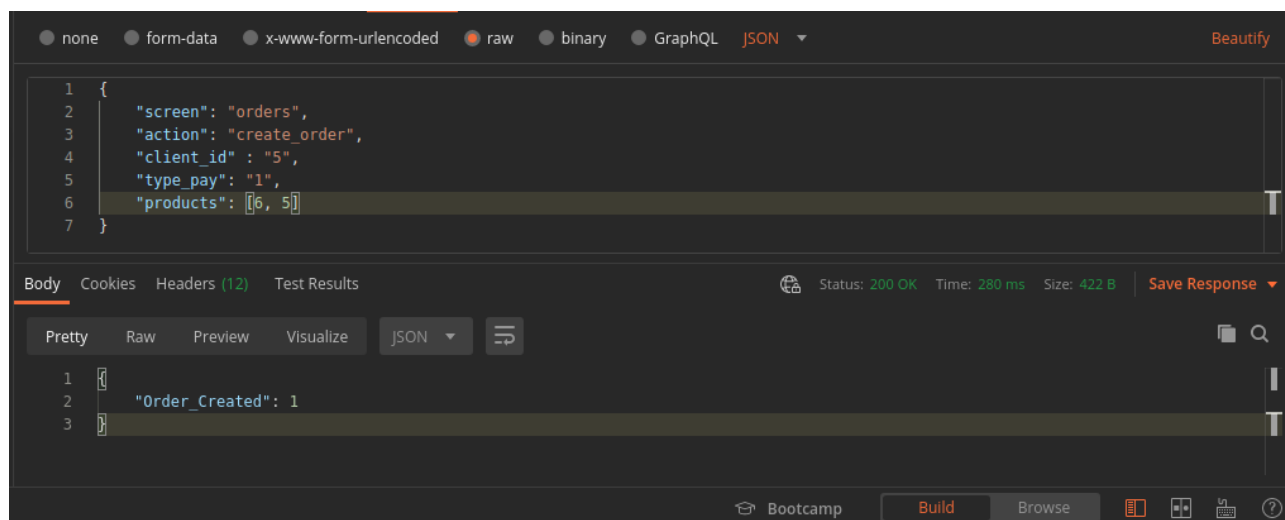
The screenshot shows a REST client interface with the following details:

- Request:** Method `GET`, URL `http://localhost:3000/api/orders/all_additional`, Headers `Content-Type: application/json`, Body `{ "screen": "orders", "action": "all_additional" }`.
- Response:** Status `200 OK`, Time `86 ms`, Size `1.03 KB`. The response body is a JSON array with one object:

```
[{"id": "1", "forTypePrd": "1", "title": "Bacon", "descr": "10g", "img": "https://bi.eletrosom.com/api_fast_food/v1/imgs/bacon.png", "price": "1.50"}]
```

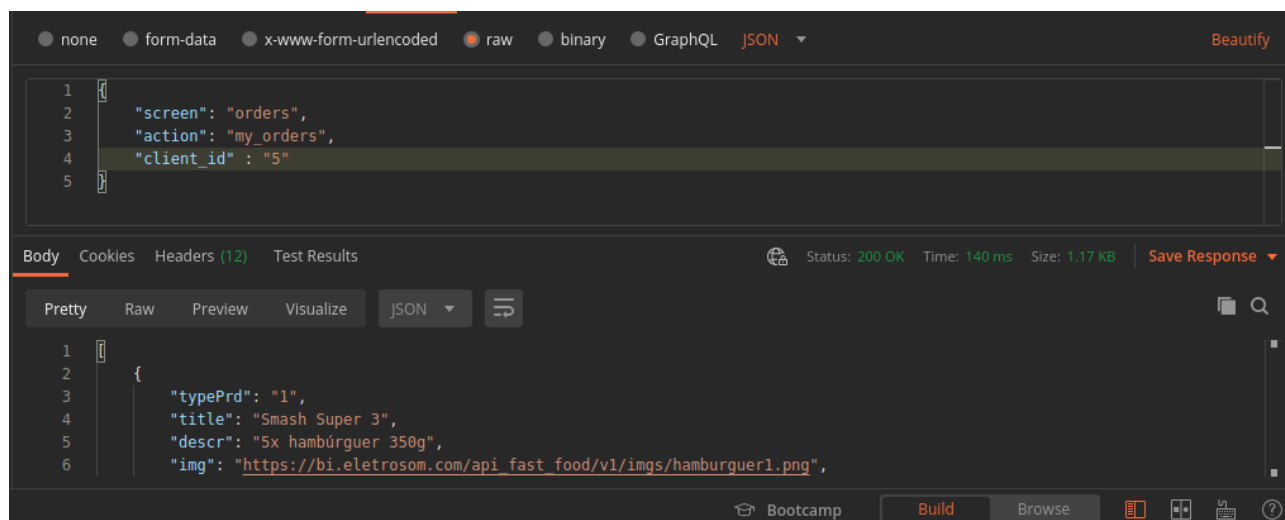
→ `create_order` (Criar um pedido)

Para a criação do pedido, é necessário passar como parâmetro o id do cliente (`client_id`), o tipo de pagamento (`type_pay`), e os produtos (`products`) em formato de array.



→ `my_orders` (Todos os pedidos do cliente)

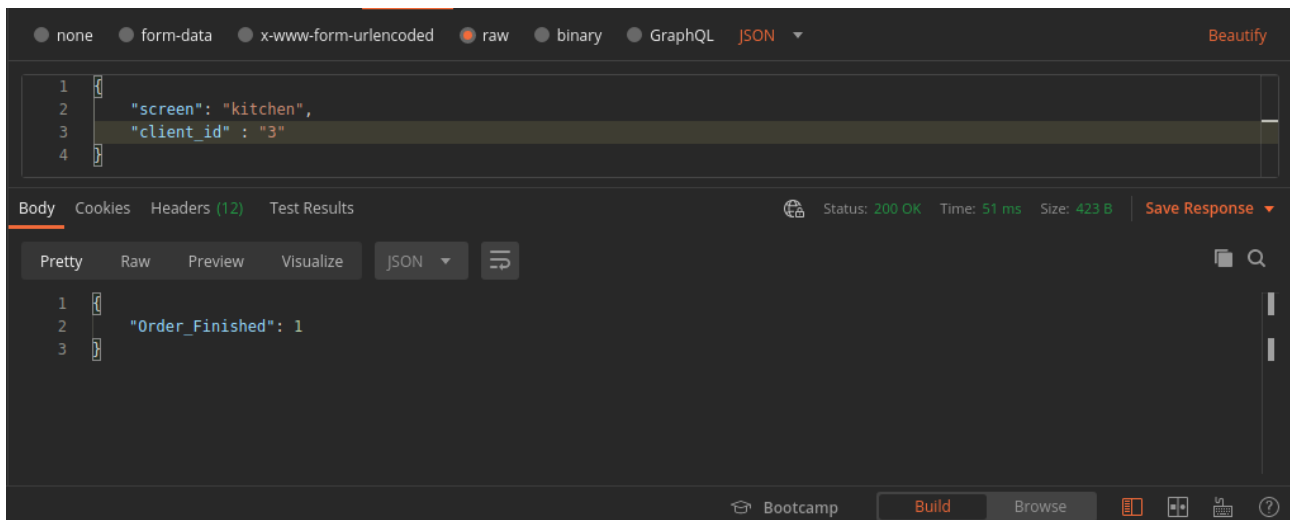
Para buscar todos os pedidos do cliente é necessário passar como parâmetro o id do cliente (`client_id`).



Cozinha

A cozinha é responsável por alterar o estado do pedido de ativo para finalizado, para realizar esta alteração é necessário passar como parâmetro o id do cliente (`client_id`).

→ **kitchen** (Alterar pedido para finalizado)



Retirada

A Retirada é responsável por exibir todos os nomes dos clientes e seus respectivos status.

→ **pickUp**(Todos os nomes de clientes e seus respectivos status)

