

Atividade Prática Serviços Web

Car Wash API

Vinicius Carpes de Freitas.

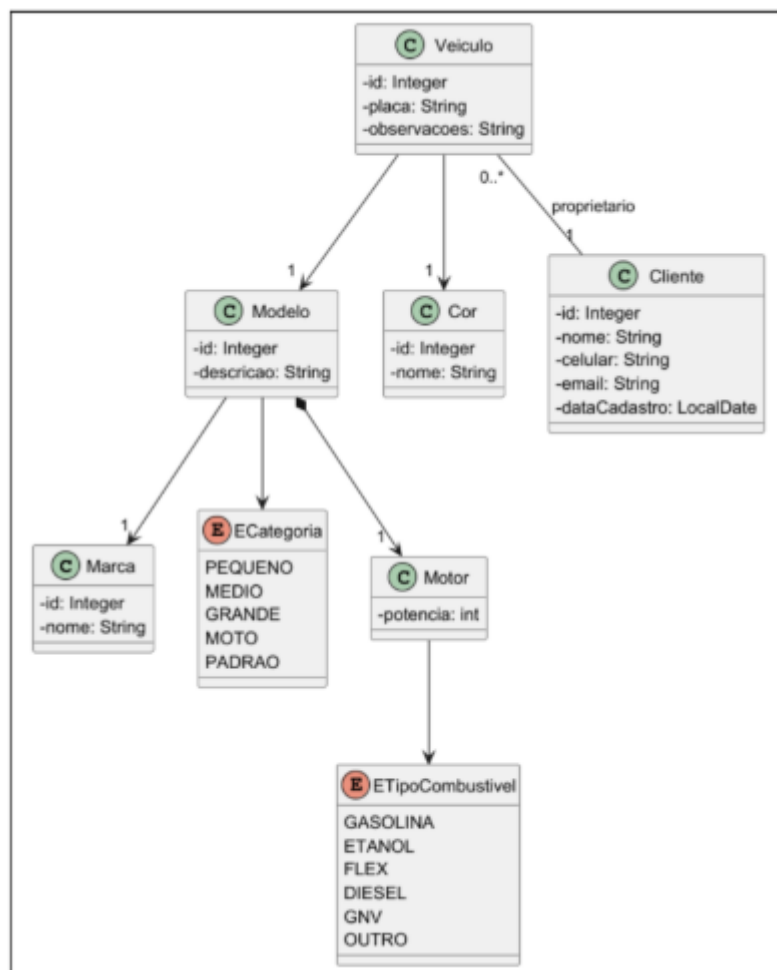
Florianópolis, 11 de junho de 2025.

1. Introdução

Este trabalho tem como objetivo exemplificar e explicar o funcionamento da API para o sistema de gerenciamento de uma empresa de lavação de automóveis - Car Wash - conforme proposto pelo Professor Dr. Marcos Pisching.

O desenvolvimento da API foi realizado conforme as seguintes instruções:

- Utilizar Java e o ecossistema Spring para o desenvolvimento da aplicação.
- Gerenciar as dependências de projeto com Maven
- Conectar com banco de dados (preferencialmente MySQL)
- Possuir ao menos as seguintes camadas: Model, Controller e Repository.
- Criar endpoints para operações CRUD para as entidades Marca, Cliente, Veículo e Modelo.
- Construir o domínio da aplicação com base no seguinte diagrama:



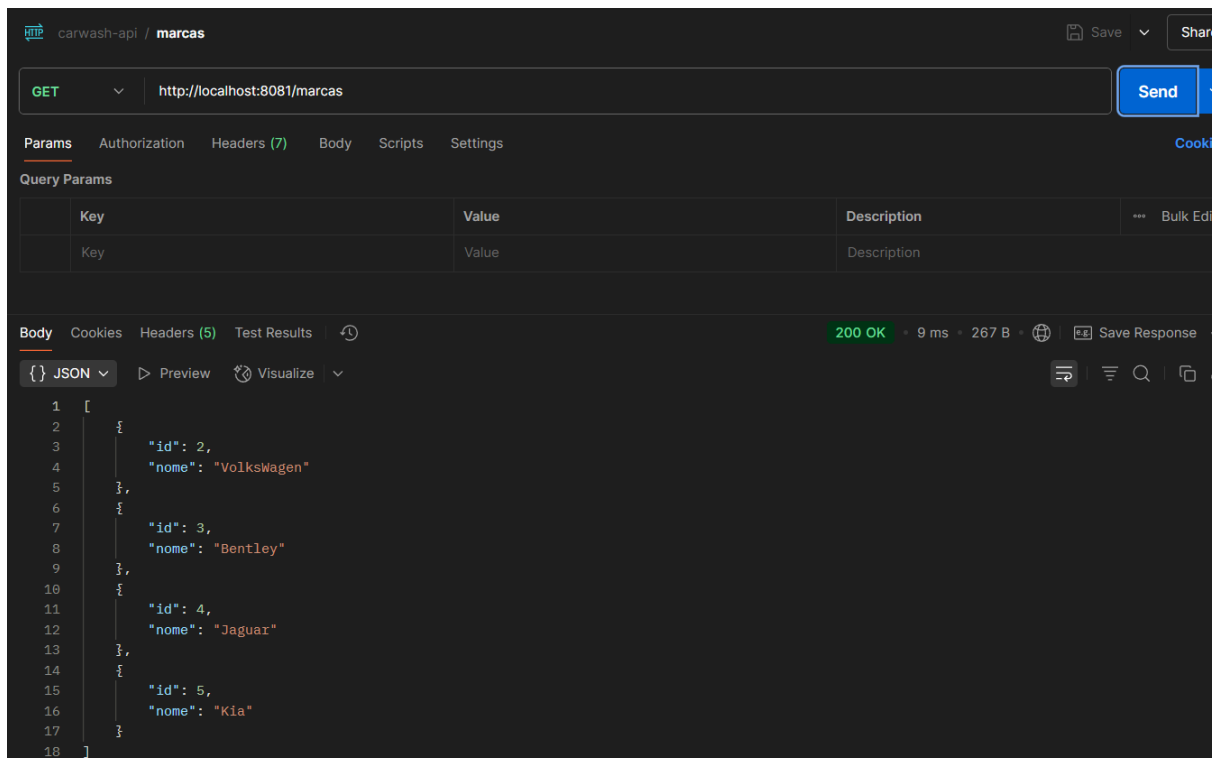
Fonte: Prof. Dr. Marcos Pisching

2. Endpoints

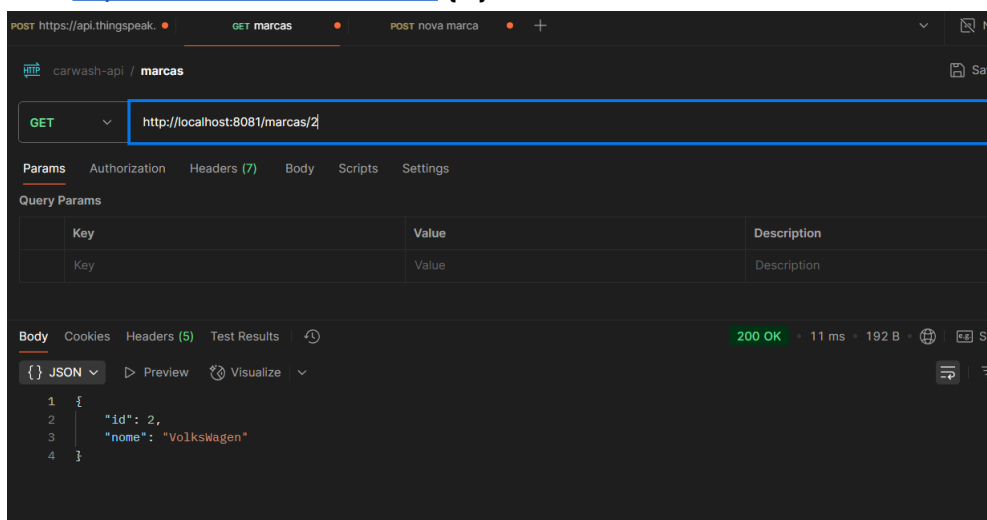
2.1 Marca

As requisições HTTP para CRUD de marca deverão ser feitas a partir da URL raiz <http://localhost:8081/marcas>.

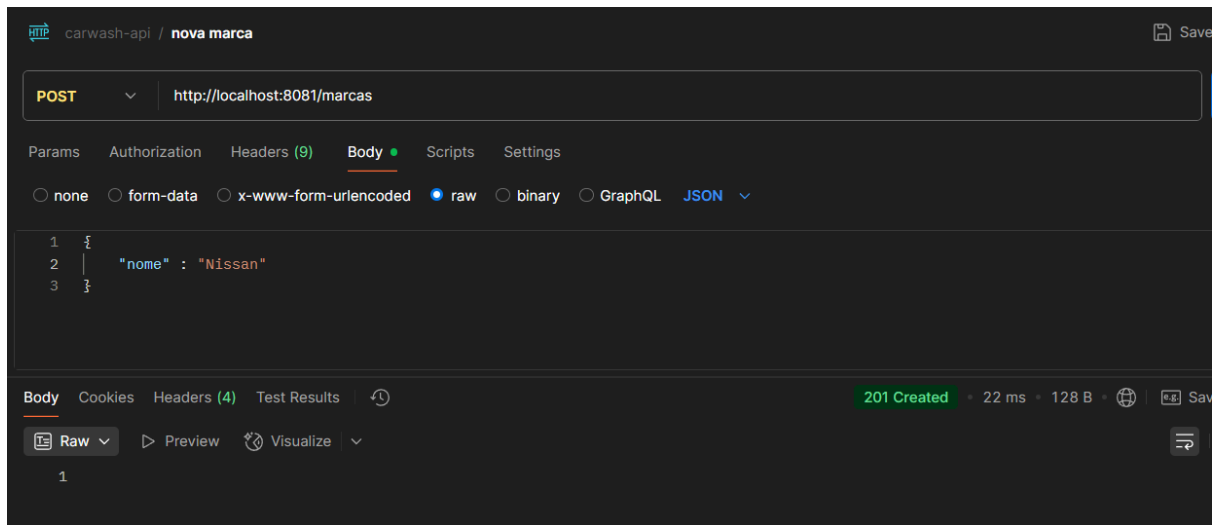
- a) GET <http://localhost:8081/marcas> - retorna todas as marcas.



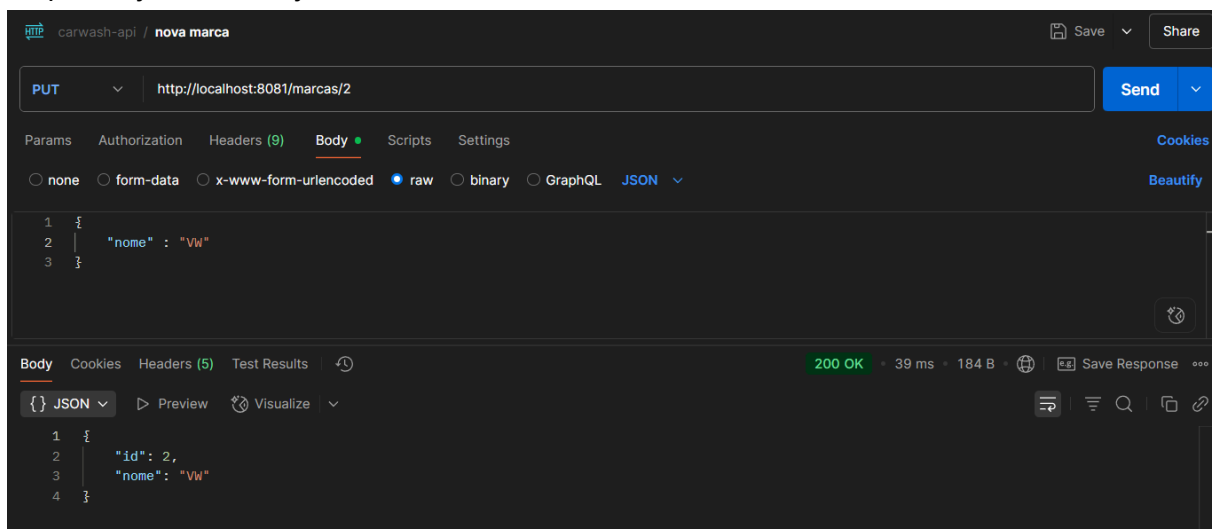
- b) GET <http://localhost:8081/marcas/{id}> - retorna a marca conforme o id.



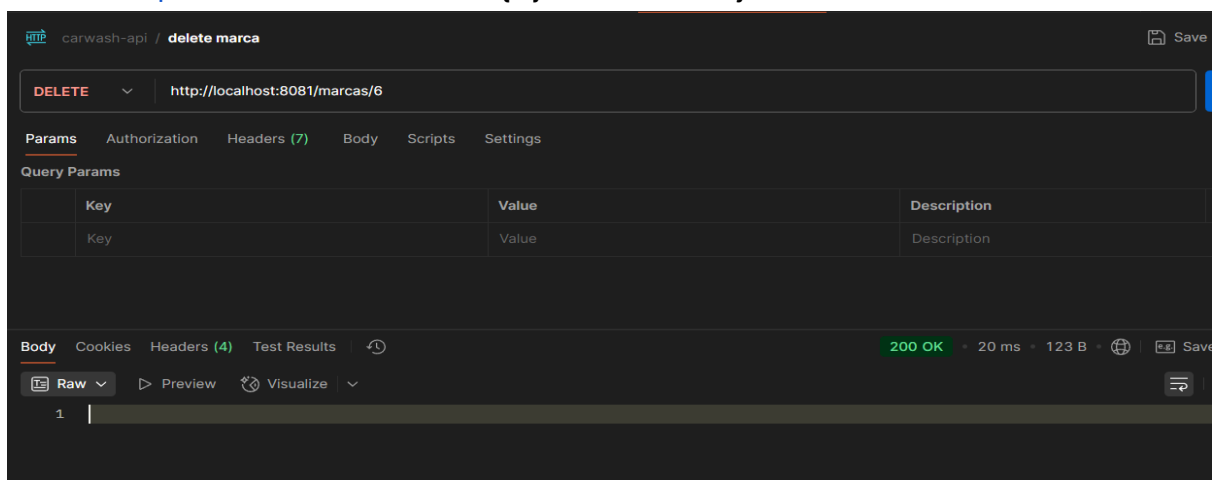
- c) POST <http://localhost:8081/marcas> - cria um objeto Marca.
Required: json com objeto de Marca.



- d) PUT <http://localhost:8081/marcas/{id}> - atualiza um objeto Marca conforme o id.
Required: json com objeto de Marca.



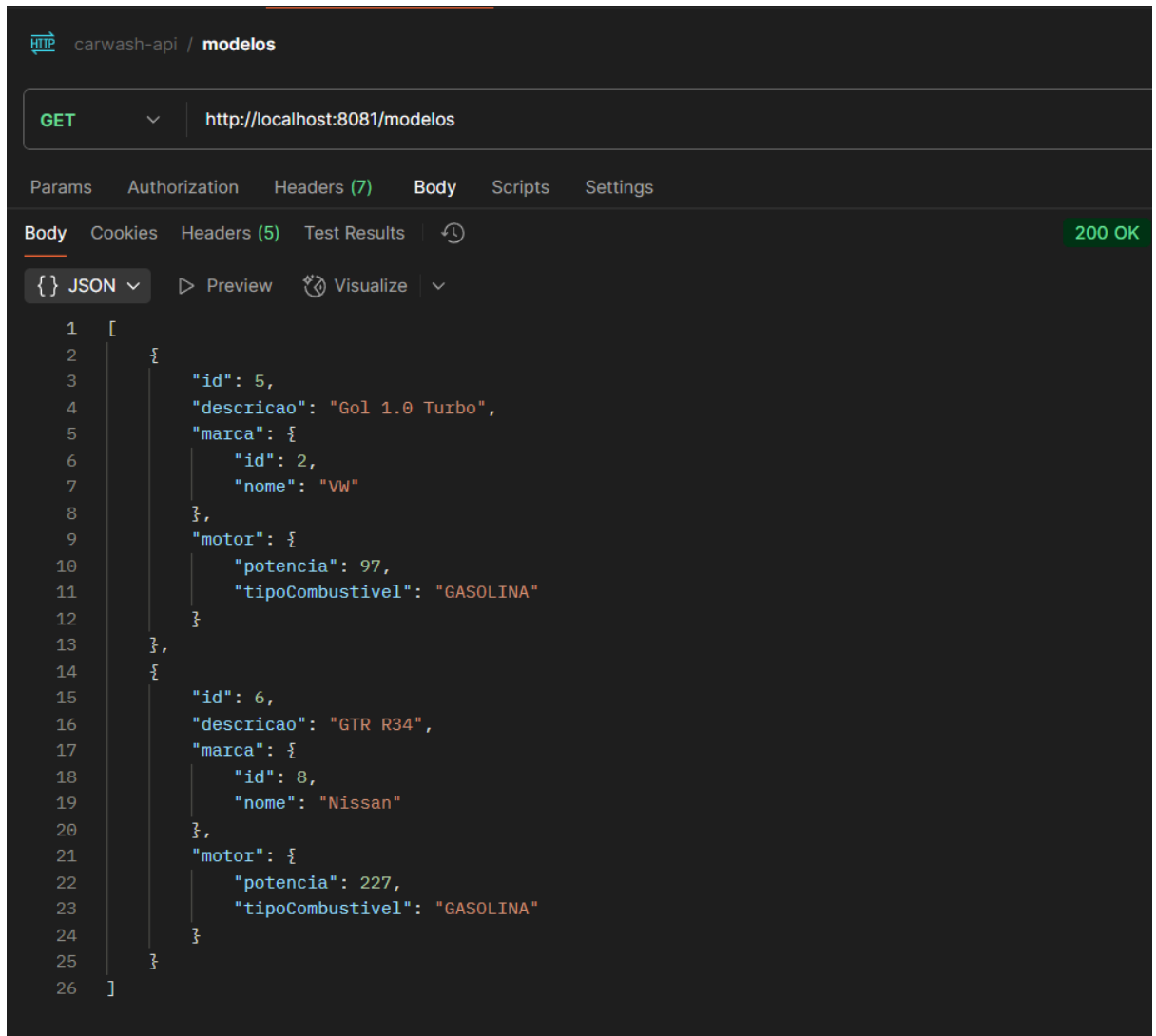
- e) DELETE <http://localhost:8081/marcas/{id}> - deleta um objeto Marca conforme o id.



2.2 Modelo

As requisições HTTP para CRUD de modelos deverão ser feitas a partir da URL raiz <http://localhost:8081/modelos>.

- a) GET <http://localhost:8081/modelos> - retorna todos os modelos.

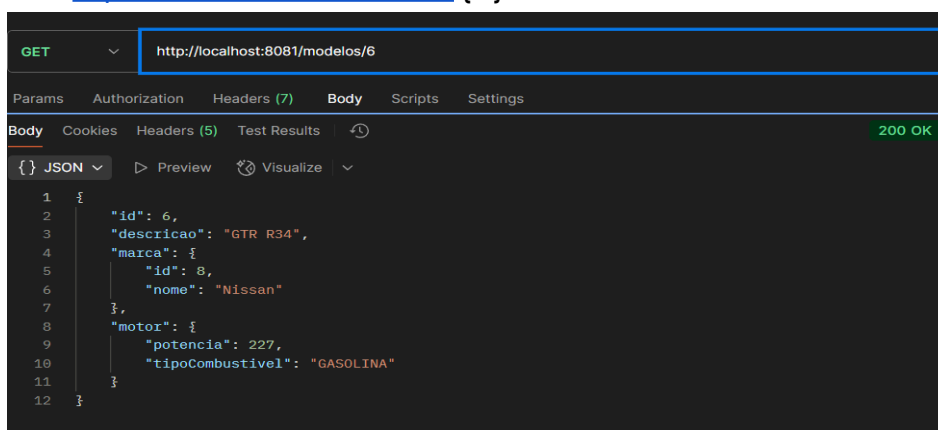


The screenshot shows a REST client interface with the following details:

- Method: GET
- URL: <http://localhost:8081/modelos>
- Status: 200 OK
- Response Body (JSON):

```
[
  {
    "id": 5,
    "descricao": "Gol 1.0 Turbo",
    "marca": {
      "id": 2,
      "nome": "VW"
    },
    "motor": {
      "potencia": 97,
      "tipoCombustivel": "GASOLINA"
    }
  },
  {
    "id": 6,
    "descricao": "GTR R34",
    "marca": {
      "id": 8,
      "nome": "Nissan"
    },
    "motor": {
      "potencia": 227,
      "tipoCombustivel": "GASOLINA"
    }
  }
]
```

- b) GET <http://localhost:8081/modelos/{id}> - retorna o modelo conforme id.

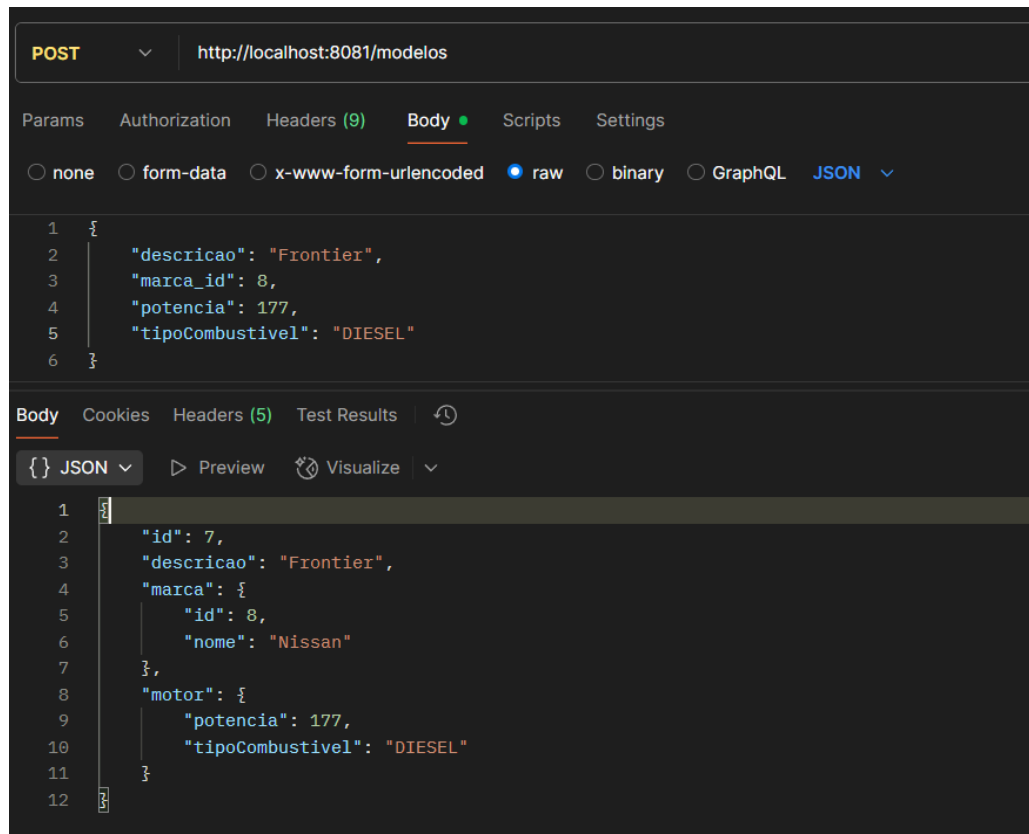


The screenshot shows a REST client interface with the following details:

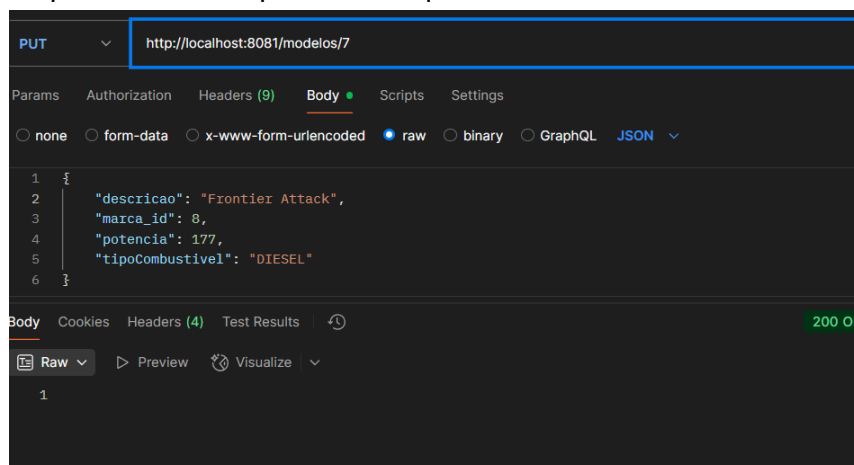
- Method: GET
- URL: <http://localhost:8081/modelos/6>
- Status: 200 OK
- Response Body (JSON):

```
{
  "id": 6,
  "descricao": "GTR R34",
  "marca": {
    "id": 8,
    "nome": "Nissan"
  },
  "motor": {
    "potencia": 227,
    "tipoCombustivel": "GASOLINA"
  }
}
```

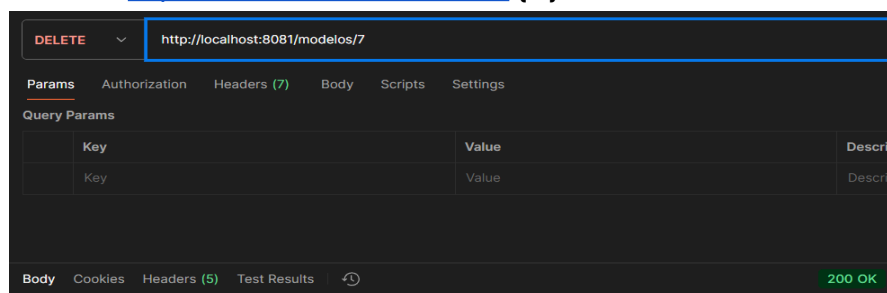
- c) POST <http://localhost:8081/modelos> - cria um novo modelo.
Required: Deve enviar no corpo descricao, marca_id, potencia e tipoCombustivel.



- d) PUT <http://localhost:8081/modelos/{id}> -atualiza modelo conforme id.
Required: Mesmos parâmetros que o item c.



- e) DELETE <http://localhost:8081/modelos/{id}> - deleta modelo conforme id.



2.3 Cliente

As requisições HTTP para CRUD de clientes deverão ser feitas a partir da URL raiz <http://localhost:8081/clientes>.

- a) GET <http://localhost:8081/clientes> - retorna todos os clientes.

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

- Method: GET
- URL: <http://localhost:8081/clientes>
- Status: 200 OK
- Response Time: 34 ms
- Response Size: 948 B
- Body Type: JSON

The JSON response is as follows:

```
{  "id": 1,  "nome": "Vinicius",  "celular": 4884848480,  "email": "vinicdf@gmail.com",  "dataCadastro": "2025-06-10",  "veiculos": [    {      "id": 3,      "placa": "AJK88P4",      "cor": "Vermelho",      "modelo": "Gol 1.0 Turbo",      "cliente": {        "id": 1,        "nome": "Vinicius",        "celular": 4884848480,        "email": "vinicdf@gmail.com",        "dataCadastro": "2025-06-10"      }    },    {      "id": 4,      "placa": "AJK88P4",      "cor": "Vermelho",      "modelo": "GTR R34",      "cliente": {        "id": 1,        "nome": "Vinicius",        "celular": 4884848480,        "email": "vinicdf@gmail.com",        "dataCadastro": "2025-06-10"      }    }  ]}
```

- b) GET <http://localhost:8081/clientes/{id}> - retorna o cliente conforme id.

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

- Method: GET
- URL: <http://localhost:8081/clientes/1>
- Status: 200 OK
- Body Type: JSON

The JSON response is as follows:

```
{  "id": 1,  "nome": "Vinicius",  "celular": 4884848480,  "email": "vinicdf@gmail.com",  "dataCadastro": "2025-06-10",  "veiculos": [  ]}
```

- c) POST <http://localhost:8081/clientes> - criar um novo cliente.

Required: deve informar nome, celular e email no json.

POST <http://localhost:8081/clientes>

Params Authorization Headers (9) **Body** Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▾

```
1 {
2   "nome": "Marcos Pisching",
3   "celular": "4812345678",
4   "email": "mp@gmail.com"
5 }
```

Body Cookies Headers (5) Test Results 201 Created

{} JSON ▾ ▶ Preview 🔗 Visualize ▾

```
1 {
2   "id": 3,
3   "nome": "Marcos Pisching",
4   "celular": "4812345678",
5   "email": "mp@gmail.com",
6   "dataCadastro": "2025-06-11",
7   "veiculos": []
8 }
```

d) PUT <http://localhost:8081/clientes/{id}> - atualiza o cliente conforme id.

PUT <http://localhost:8081/clientes/3>

Params Authorization Headers (9) **Body** Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▾

```
1 {
2   "nome": "Marcos André Pisching",
3   "celular": "4812345678",
4   "email": "mp@gmail.com"
5 }
```

Body Cookies Headers (4) Test Results 200 OK

Raw ▾ ▶ Preview 🔗 Visualize ▾

```
1
```

e) DELETE <http://localhost:8081/clientes/{id}> - deleta o cliente conforme o id.

DELETE <http://localhost:8081/clientes/3>

Params Authorization Headers (7) **Body** Scripts Settings

Query Params

	Key	Value	Description
	Key	Value	Description

Body Cookies Headers (4) Test Results 200 OK

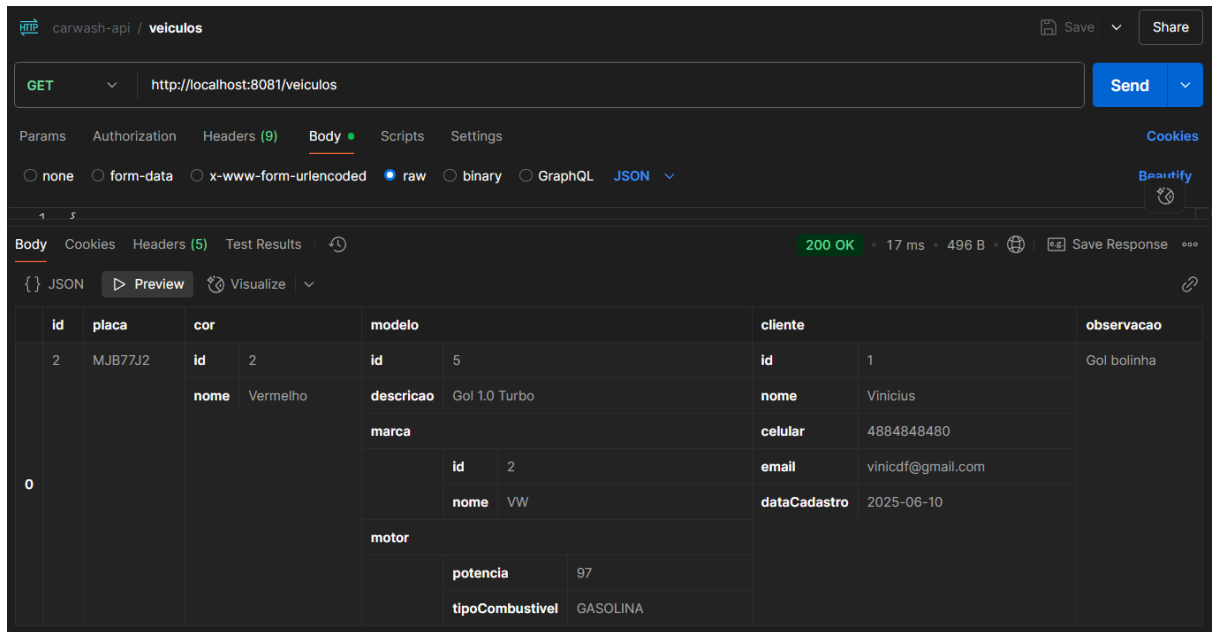
Raw ▾ ▶ Preview 🔗 Visualize ▾

```
1
```


2.4 Veículo

As requisições HTTP para CRUD de veículos deverão ser feitas a partir da URL raiz <http://localhost:8081/veiculos>.

- f) GET <http://localhost:8081/veiculos> - retorna todos os veículos.

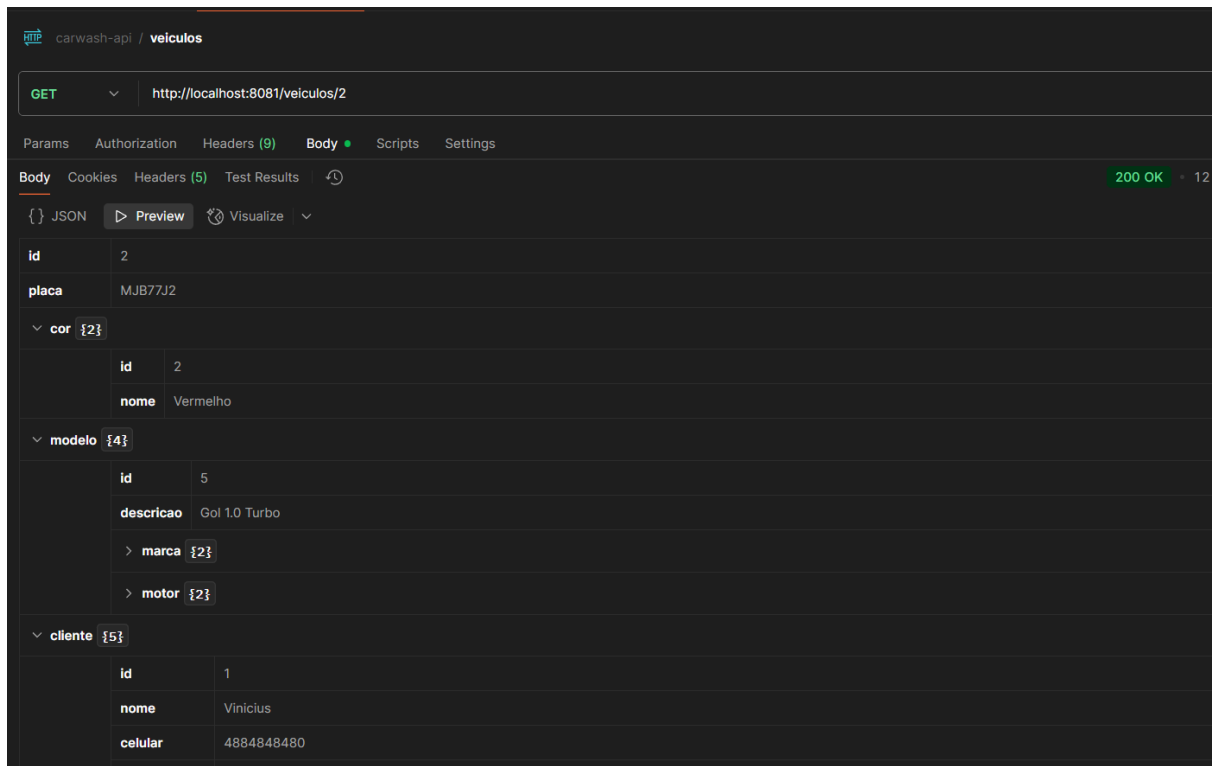


The screenshot shows a REST client interface with the following details:

- Method: GET
- URL: <http://localhost:8081/veiculos>
- Status: 200 OK
- Response Type: JSON
- Response Body (JSON):

id	placa	cor	modelo	cliente	observacao																														
2	MJB77J2	<table border="1"><tr><td>id</td><td>2</td></tr><tr><td>nome</td><td>Vermelho</td></tr></table>	id	2	nome	Vermelho	<table border="1"><tr><td>id</td><td>5</td></tr><tr><td>descricao</td><td>Gol 1.0 Turbo</td></tr><tr><td>marca</td><td><table border="1"><tr><td>id</td><td>2</td></tr><tr><td>nome</td><td>VW</td></tr></table></td></tr><tr><td>motor</td><td><table border="1"><tr><td>potencia</td><td>97</td></tr><tr><td>tipoCombustivel</td><td>GASOLINA</td></tr></table></td></tr></table>	id	5	descricao	Gol 1.0 Turbo	marca	<table border="1"><tr><td>id</td><td>2</td></tr><tr><td>nome</td><td>VW</td></tr></table>	id	2	nome	VW	motor	<table border="1"><tr><td>potencia</td><td>97</td></tr><tr><td>tipoCombustivel</td><td>GASOLINA</td></tr></table>	potencia	97	tipoCombustivel	GASOLINA	<table border="1"><tr><td>id</td><td>1</td></tr><tr><td>nome</td><td>Vinicius</td></tr><tr><td>celular</td><td>4884848480</td></tr><tr><td>email</td><td>vinicdf@gmail.com</td></tr><tr><td>dataCadastro</td><td>2025-06-10</td></tr></table>	id	1	nome	Vinicius	celular	4884848480	email	vinicdf@gmail.com	dataCadastro	2025-06-10	Gol bolinha
id	2																																		
nome	Vermelho																																		
id	5																																		
descricao	Gol 1.0 Turbo																																		
marca	<table border="1"><tr><td>id</td><td>2</td></tr><tr><td>nome</td><td>VW</td></tr></table>	id	2	nome	VW																														
id	2																																		
nome	VW																																		
motor	<table border="1"><tr><td>potencia</td><td>97</td></tr><tr><td>tipoCombustivel</td><td>GASOLINA</td></tr></table>	potencia	97	tipoCombustivel	GASOLINA																														
potencia	97																																		
tipoCombustivel	GASOLINA																																		
id	1																																		
nome	Vinicius																																		
celular	4884848480																																		
email	vinicdf@gmail.com																																		
dataCadastro	2025-06-10																																		

- g) GET <http://localhost:8081/veiculos/{id}> - retorna o veículo dado o id.

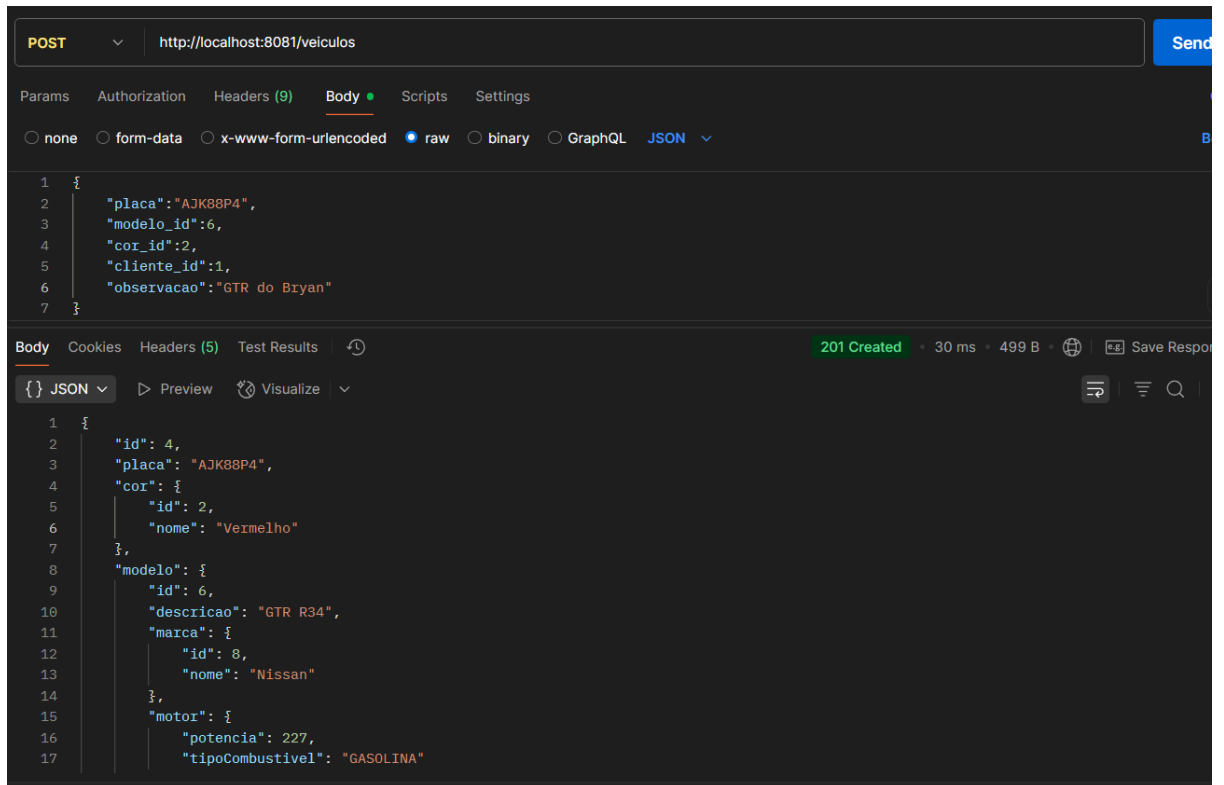


The screenshot shows a REST client interface with the following details:

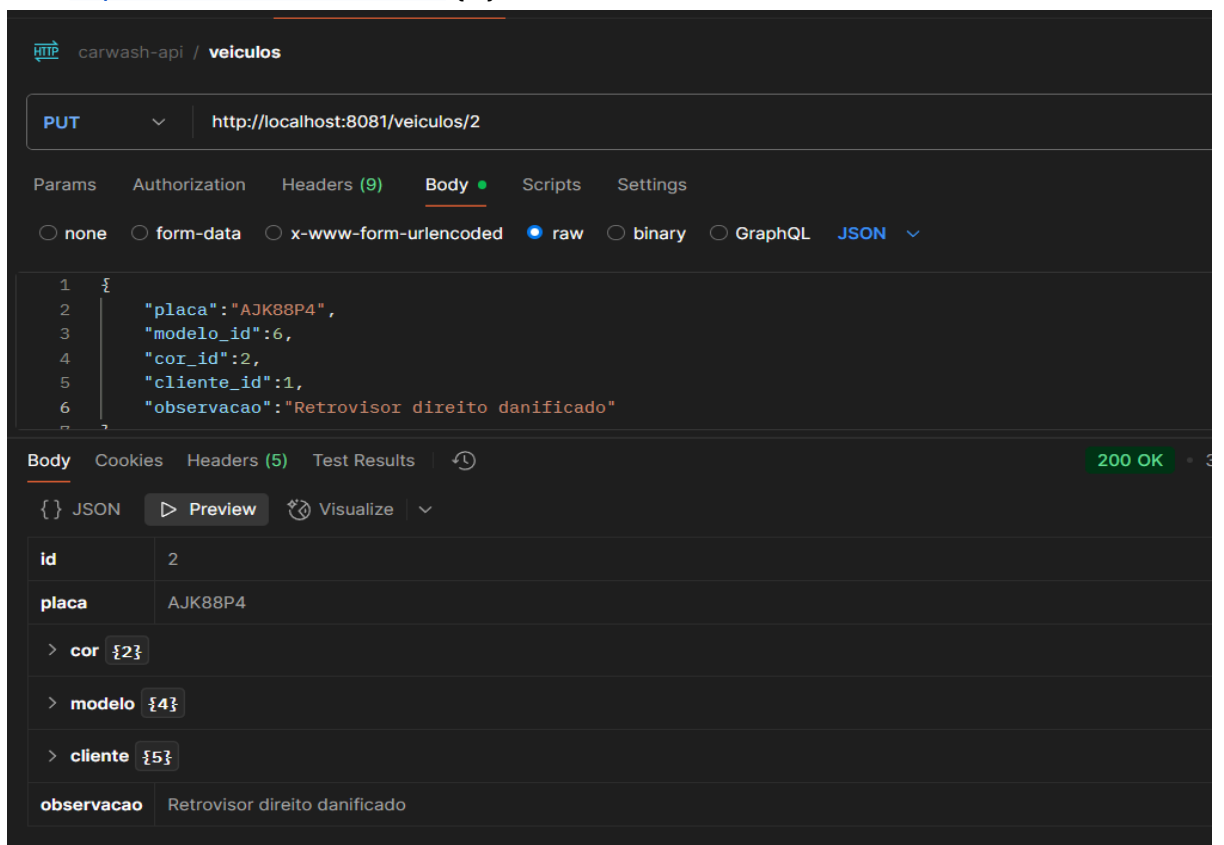
- Method: GET
- URL: <http://localhost:8081/veiculos/2>
- Status: 200 OK
- Response Type: JSON
- Response Body (JSON):

```
{  "id": 2,  "placa": "MJB77J2",  "cor": {    "id": 2,    "nome": "Vermelho"  },  "modelo": {    "id": 5,    "descricao": "Gol 1.0 Turbo",    "marca": {      "id": 2,      "nome": "VW"    },    "motor": {      "potencia": 97,      "tipoCombustivel": "GASOLINA"    }  },  "cliente": {    "id": 1,    "nome": "Vinicius",    "celular": "4884848480"  },  "observacao": "Gol bolinha"}
```

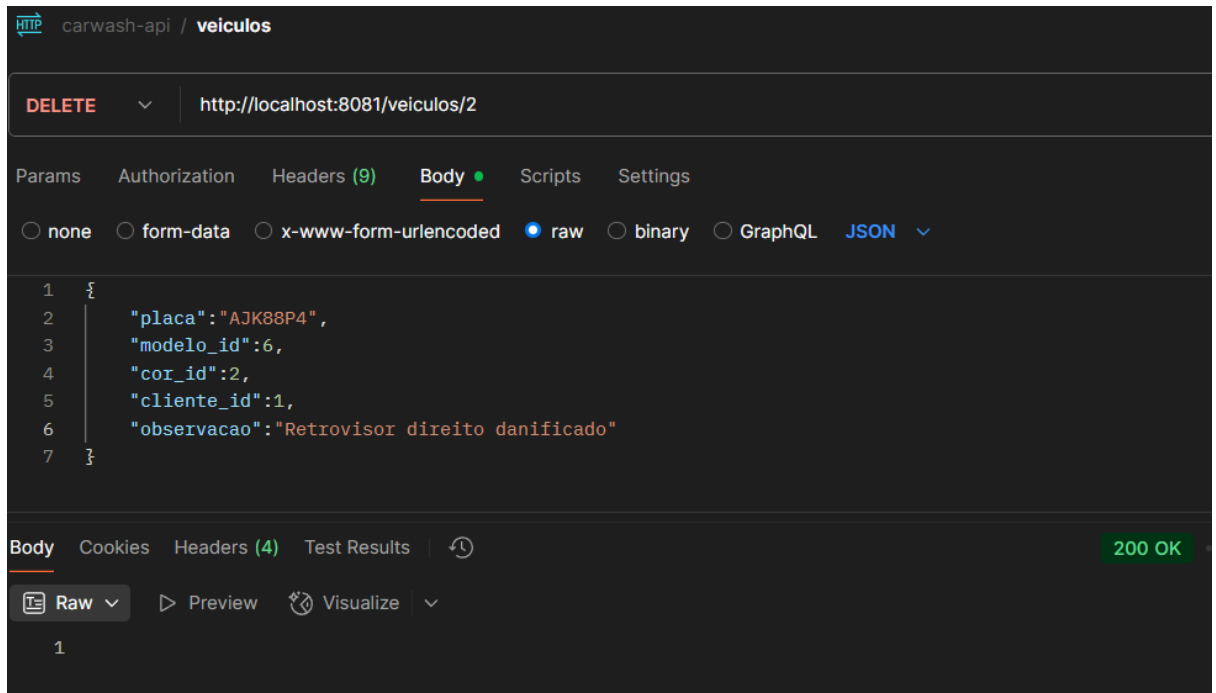
- h) POST <http://localhost:8081/veiculos> - cria um novo veículos.
Required: Deve enviar um body com VeiculoDTO, informando placa, modelo_id, cor_id, cliente_id e observacao;



- i) PUT <http://localhost:8081/veiculos/{id}> -atualiza o veículo conforme id.



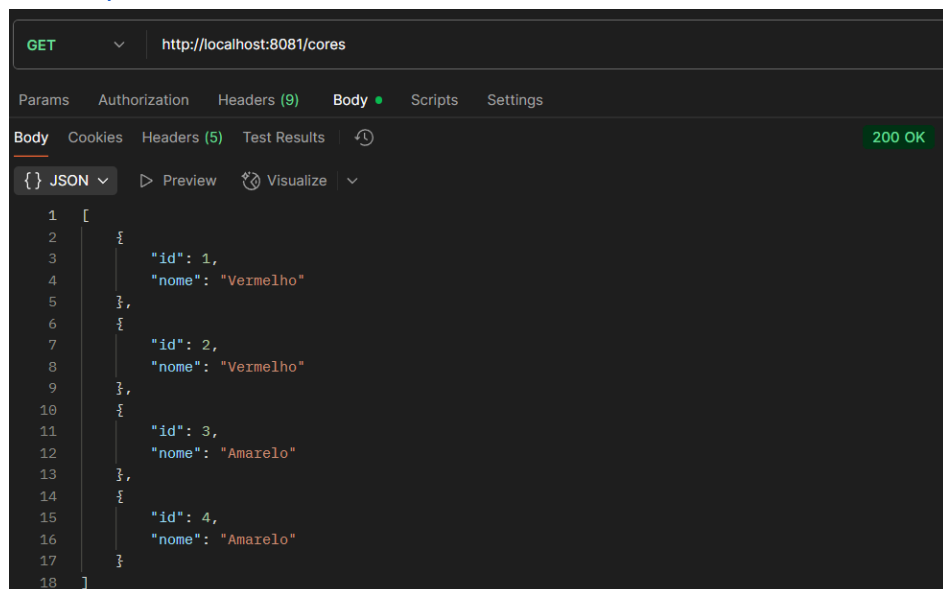
- j) DELETE <http://localhost:8081/modelos/{id}> - deleta o veículo conforme o id.



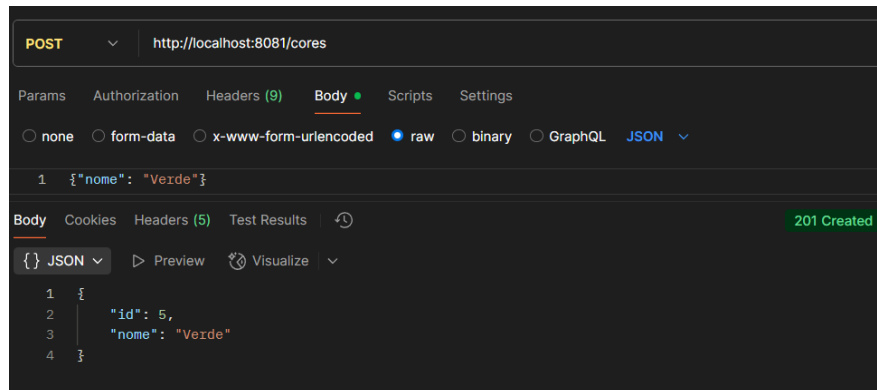
2.5 Extra: Cor

As requisições HTTP para CRUD de cores deverão ser feitas a partir da URL raiz <http://localhost:8081/cores>.

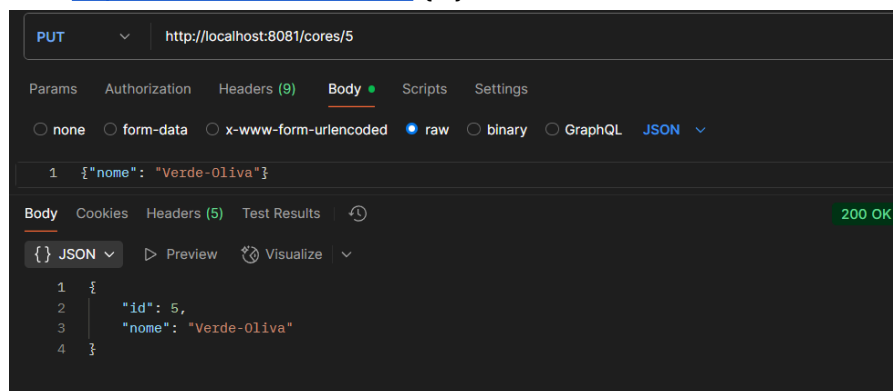
- a) GET <http://localhost:8081/cores> - retorna todas as cores.



- b) POST <http://localhost:8081/cores>- cria uma nova cor.
Required: Requer campo nome



- c) PUT <http://localhost:8081/cores/{id}>- atualiza a cor conforme id.



- d) DELETE <http://localhost:8081/cores/{id}>- deleta a cor conforme id.

