Livros S.A.

Projeto interdisciplinar das disciplinas do terceiro semestre do curso de tecnologia em sistemas para internet, que é uma plataforma para venda online de livros universitários.

Stack utilizada

Back-end: Node, Express

Estrutura do projeto

- **src/**
  - **controllers/**
    - booksController.js
  - **database/**
    - database.js
  - **routes/**
    - index.js
    - booksRouter.js
  - index.js

`/controllers` → Gerencia a lógica das requisições HTTP.

`/database` → Contém a configuração do banco de dados.

`/routes` → Define as rotas da API.

`index.js` → Responsável por iniciar o servidor.

Sobre o projeto

Configuração inicial

Após as pastas e arquivos base serem criados, no terminal é excutado o comando `npm init -y` que inicia o projeto e cria os arquivos de configuração `package.json` e `package-lock.json`.

Depedências utilizadas

- **Express.js:** O express é um framework para Node que auxilia na construção do back-end através de rotas, requisições HTTP e também para iniciar o servidor. Usa-se o seguinte comando para instalá-lo.

```
npm install express
```

- **Nodemon:** É usado para que o servidor reinicie automaticamente após alguma alteração no código ser salva. Para instalar o mesmo basta executar o seguinte comando no terminal:

```
npm install nodemon -D
```

Para executar escreva o seguinte script dentro do `package.json`

```
"scripts": {
   "dev": "nodemon ./src/index.js"
 }
```

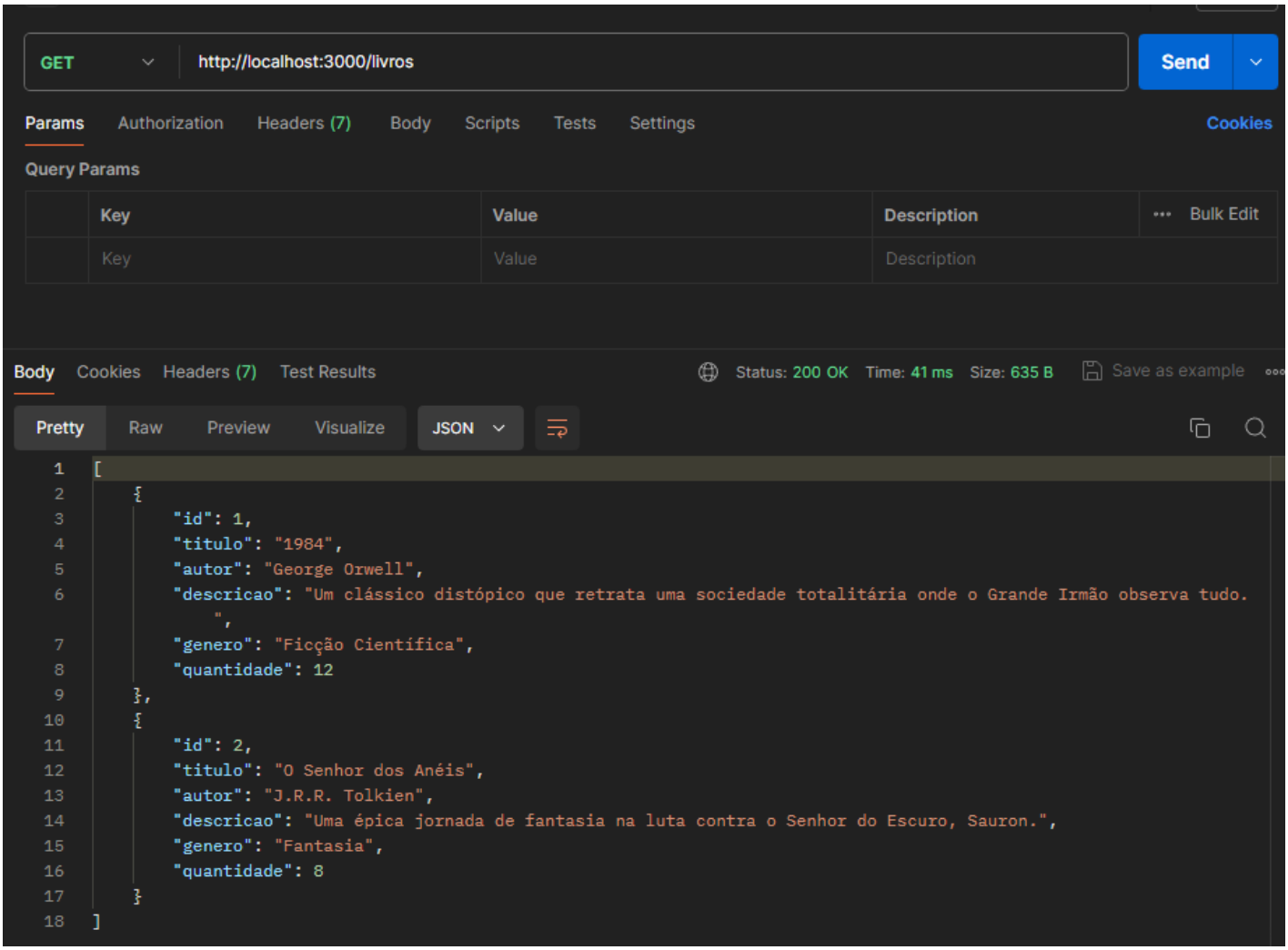
E para ser executado usa o seguinte comando no terminal:

```
npm run dev
```

Documentação da API

Retorna todos os livros

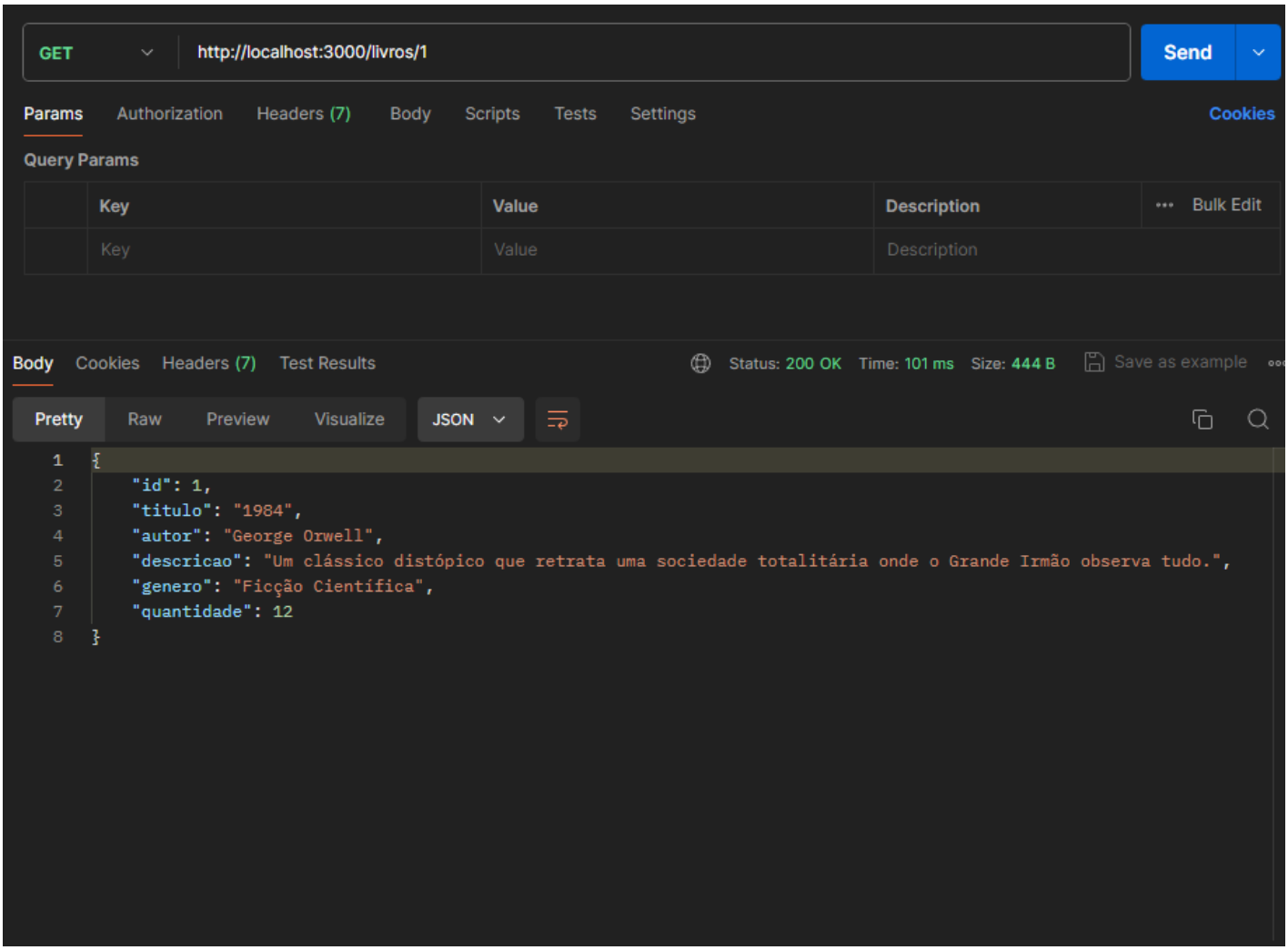
```
  GET /livros
```



Retorna um livro específico

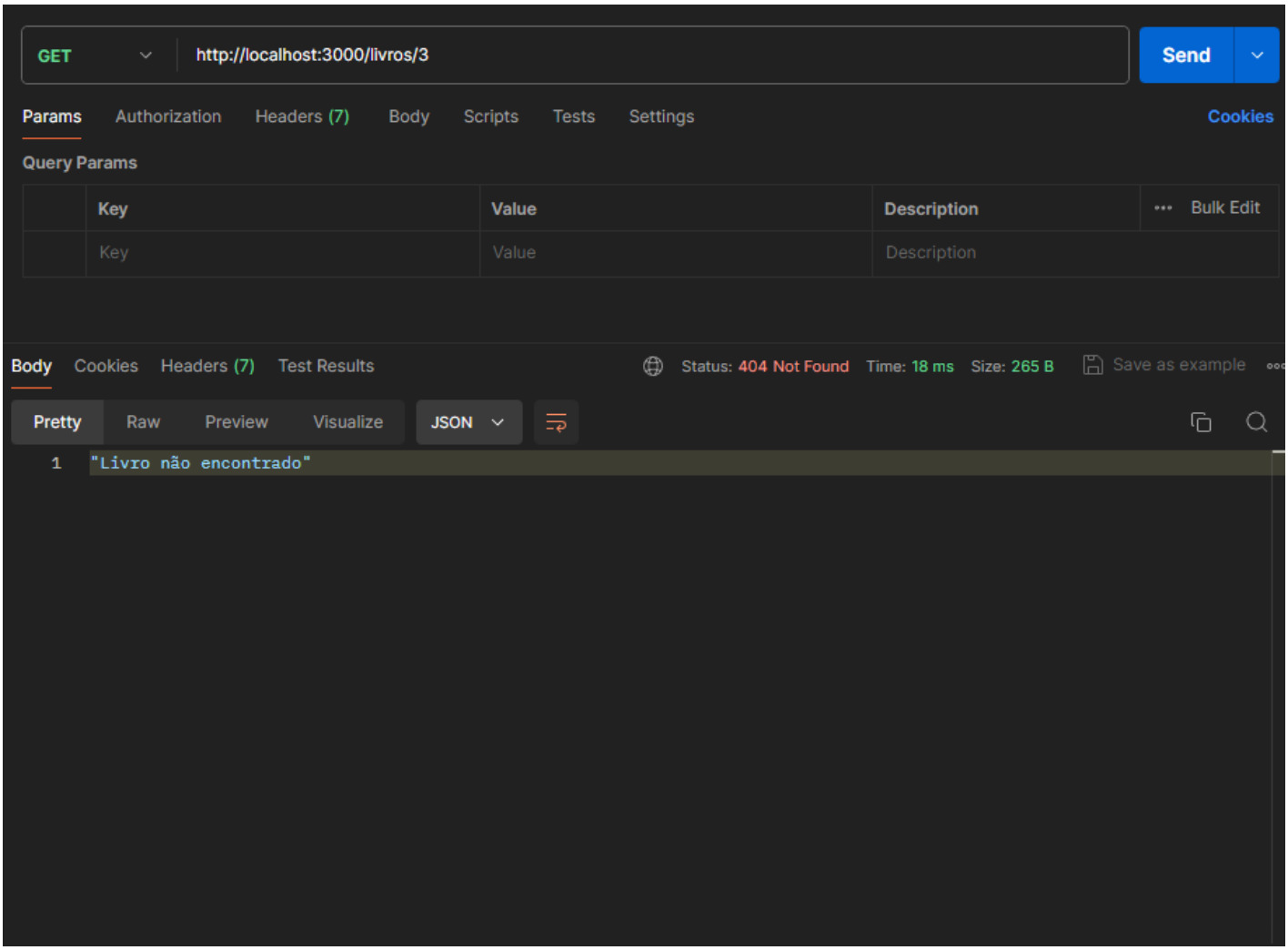
```
GET /livros/:id
```

Parâmetro	Tipo	Descrição
`id`	`string`	O ID do item que você quer



Caso não exita um livro associado ao id o aplicativo mostrará a seguinte mensagem com o status HTTP 404.

```
{"Mensagem": "Livro não encontrado"}
```



Adiciona um livro

```
POST /livro
```

Parâmetro	Tipo	Descrição
`body`	`objeto`	Objeto contendo as informações do livro a ser adicionado

O objeto a ser inserido no body deverá seguir o formato a seguir

```
{
  "id": Number,
  "titulo": "String",
  "autor": "String",
  "descricao": "String",
  "genero": "String",
  "quantidade": Number
}
```

Em caso de sucesso a seguinte mensagem será mostrada junto do status 201.

```
{"Mensagem": "Registro inserido com sucesso!"}
```

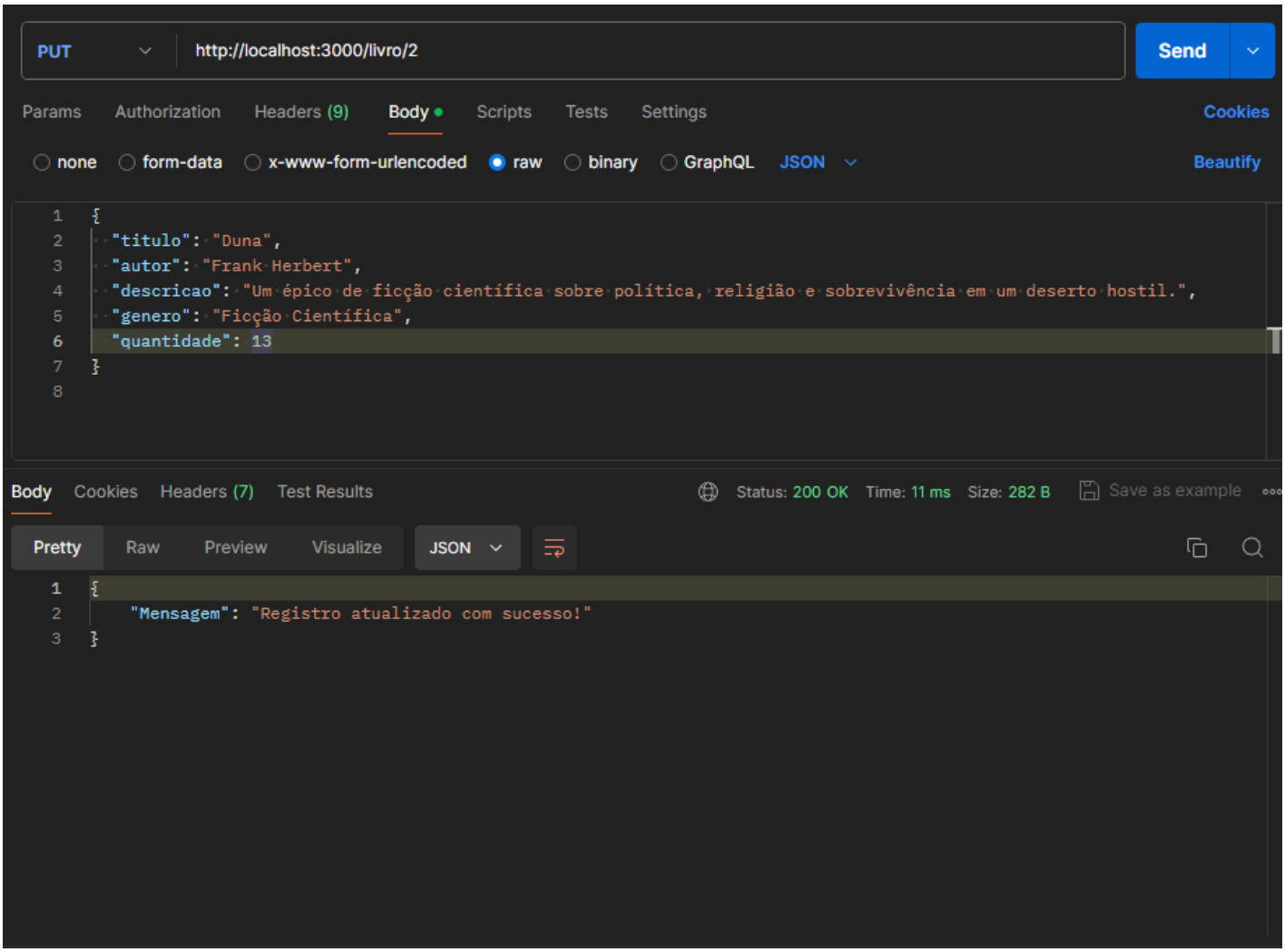
Atualiza as informações de um livro

```
PUT /livro/:id
```

Parâmetro	Tipo	Descrição
`body`	`objeto`	Objeto contendo as informações do livro a ser adicionado

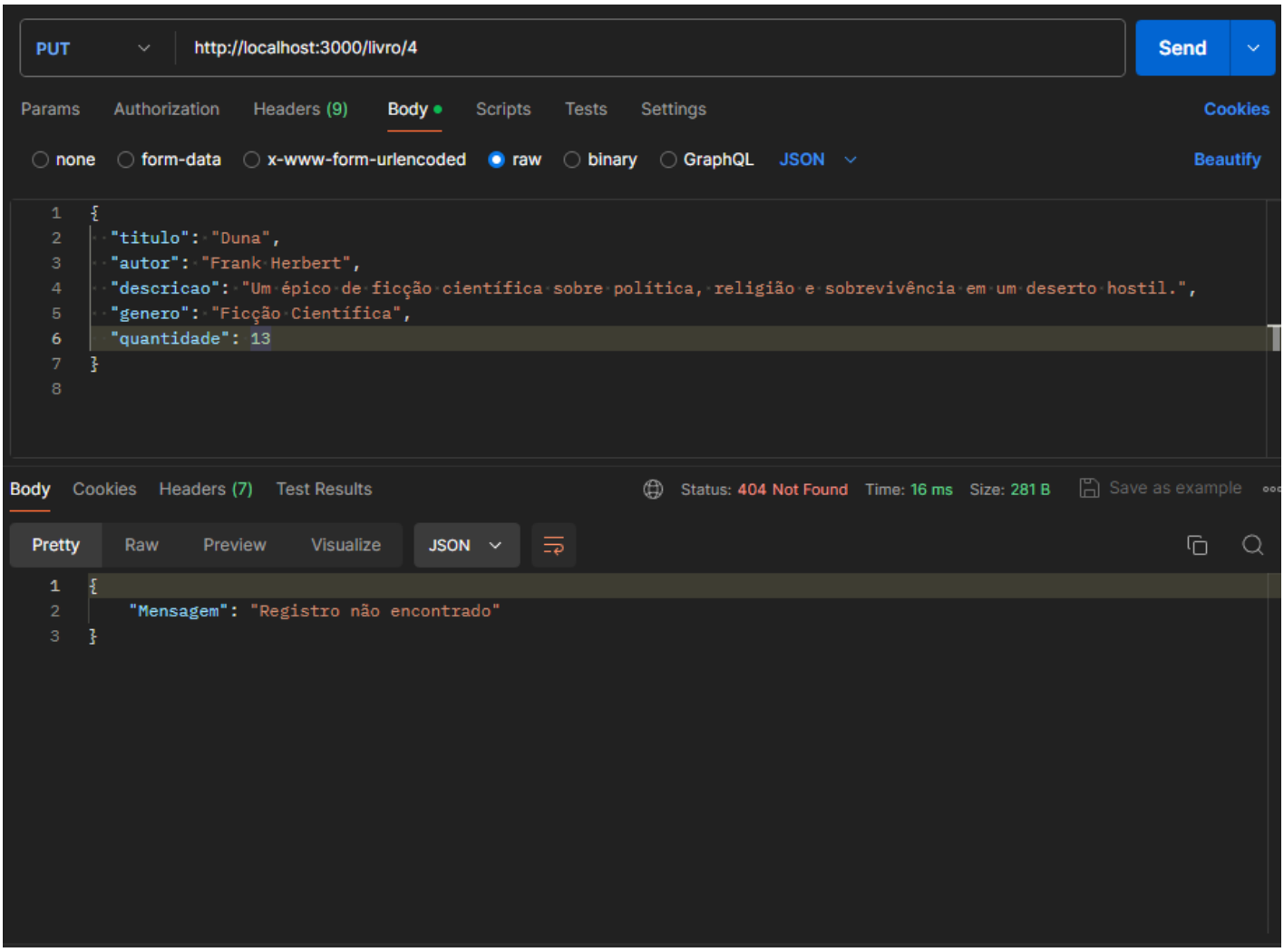
O objeto a ser inserido no body deverá seguir o formato a seguir

```
{
  "titulo": "String",
  "autor": "String",
  "descricao": "String",
  "genero": "String",
  "quantidade": Number
}
```



Caso não exita um livro associado ao id o aplicativo mostrará a seguinte mensagem com o status HTTP 404.

```
{"Mensagem": "Registro não encontrado"}
```



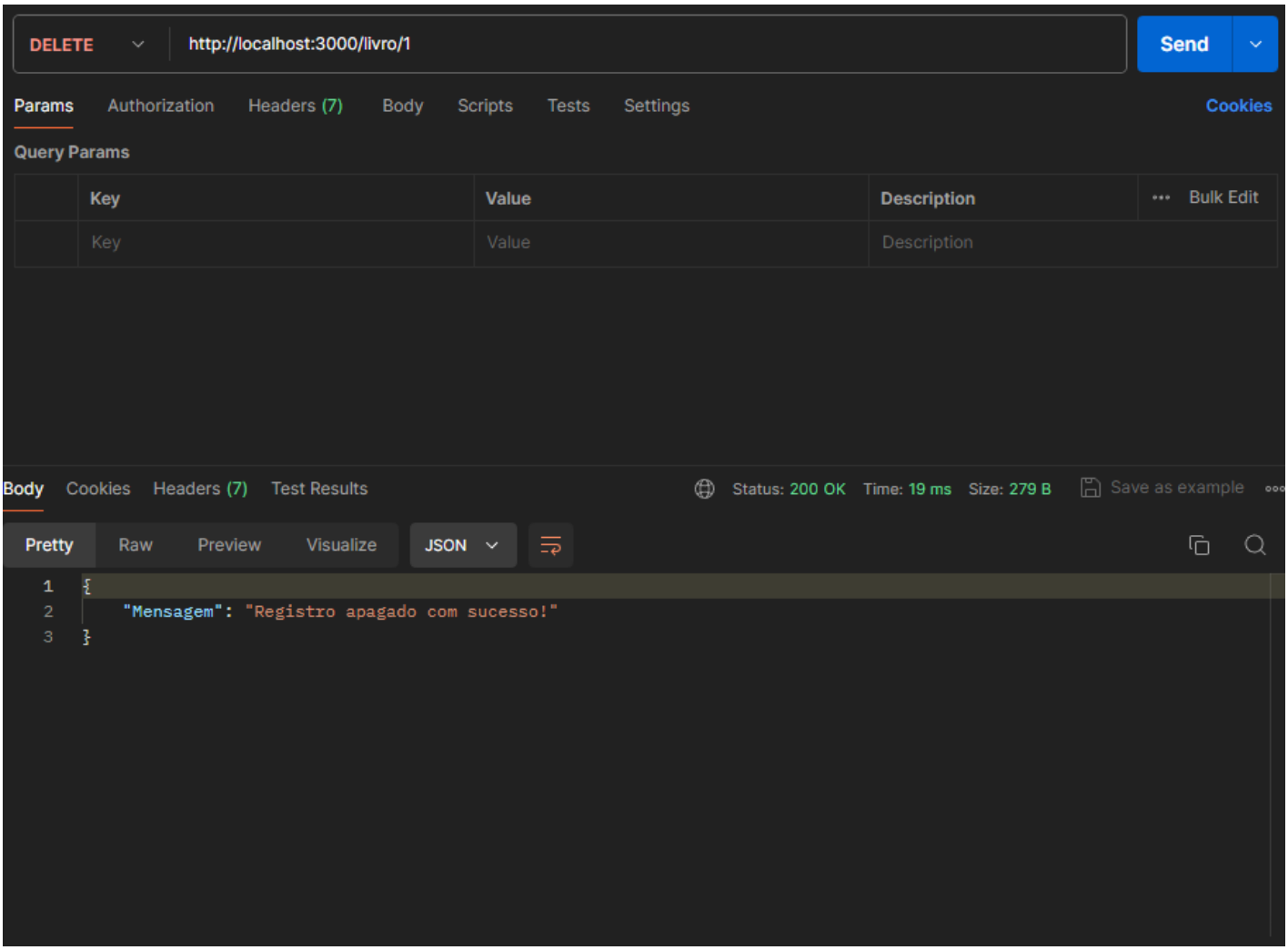
Apaga um livro

```
DELETE /livros/:id
```

Parâmetro	Tipo	Descrição
`id`	`string`	O ID do item que você quer apagar

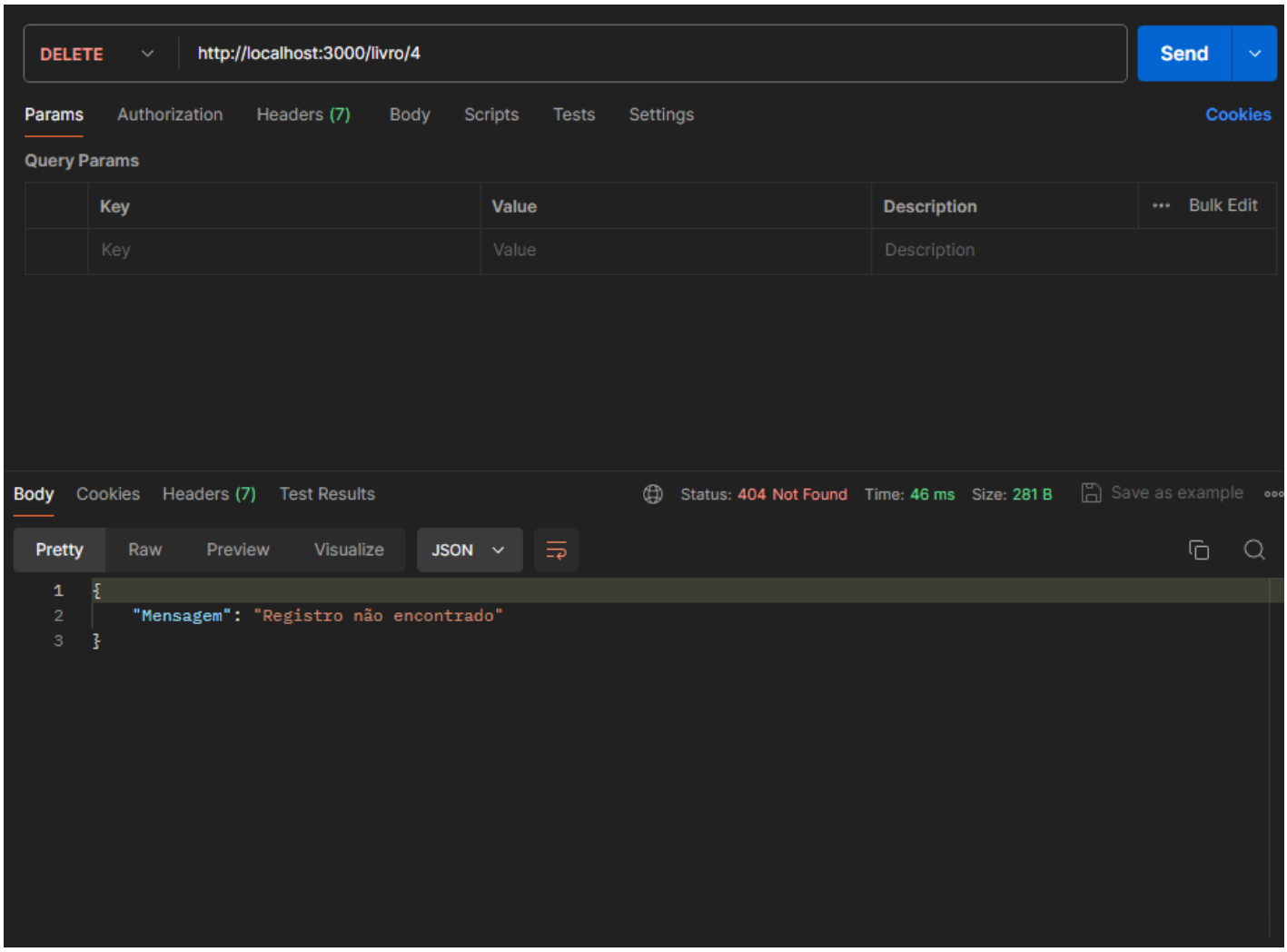
Em caso de sucesso a seguinte mensagem será mostrada junto do status 200.

```
{"Mensagem": "Registro apagado com sucesso!"}
```



Caso não exita um livro associado ao id o aplicativo mostrará a seguinte mensagem com o status HTTP 404.

```
{"Mensagem": "Registro não encontrado"}
```


DELETE
http://localhost:3000/livro/4
Send
Params
Authorization
Headers (7)
Body
Scripts
Tests
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Key
Value
Description
Body
Cookies
Headers (7)
Test Results
Status: 404 Not Found
Time: 46 ms
Size: 281 B
Save as example
Pretty
Raw
Preview
Visualize
JSON
1 {
2 "Mensagem": "Registro não encontrado"
3 }