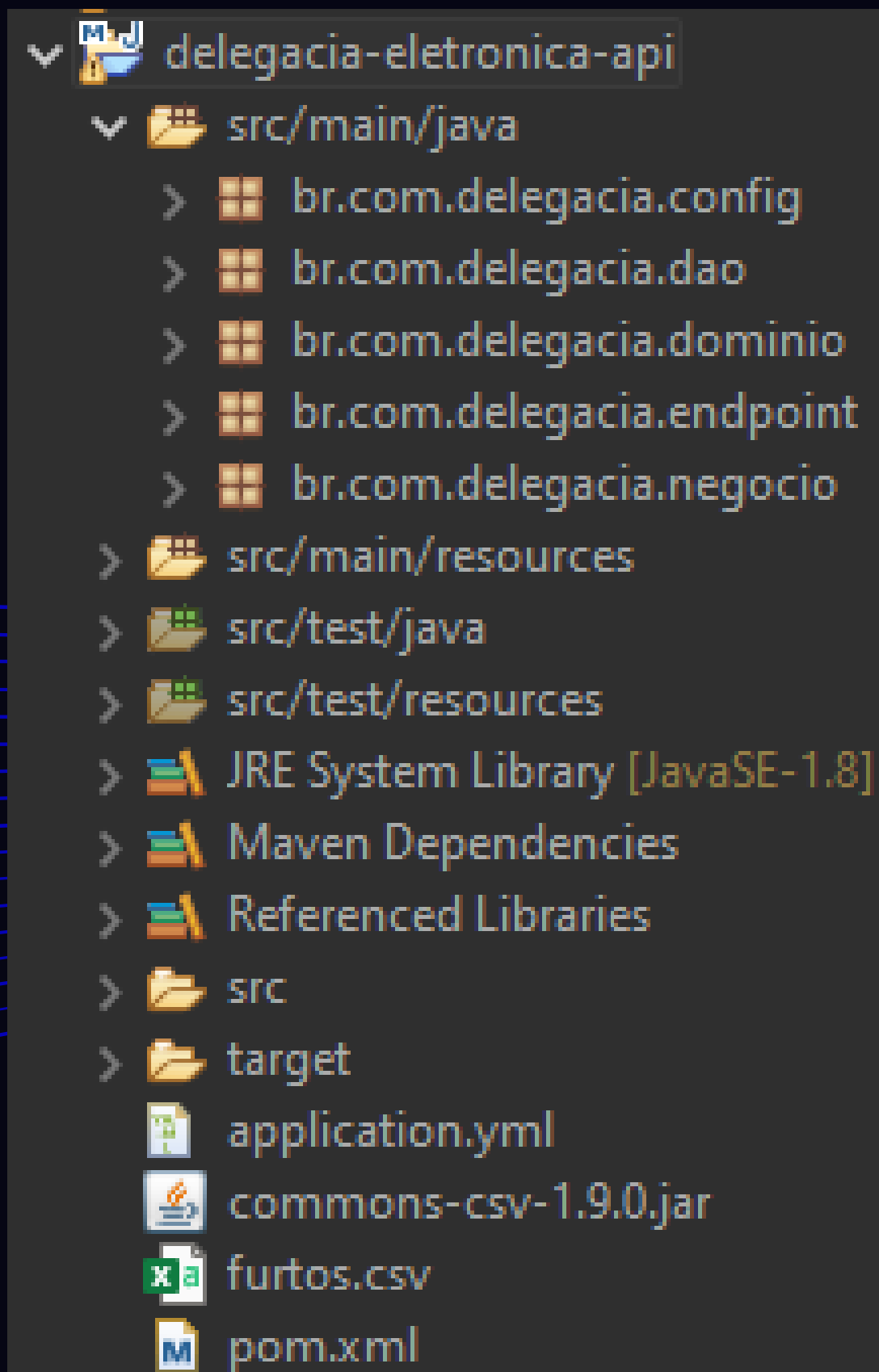


API Rest - Delegacia Eletrônica

01



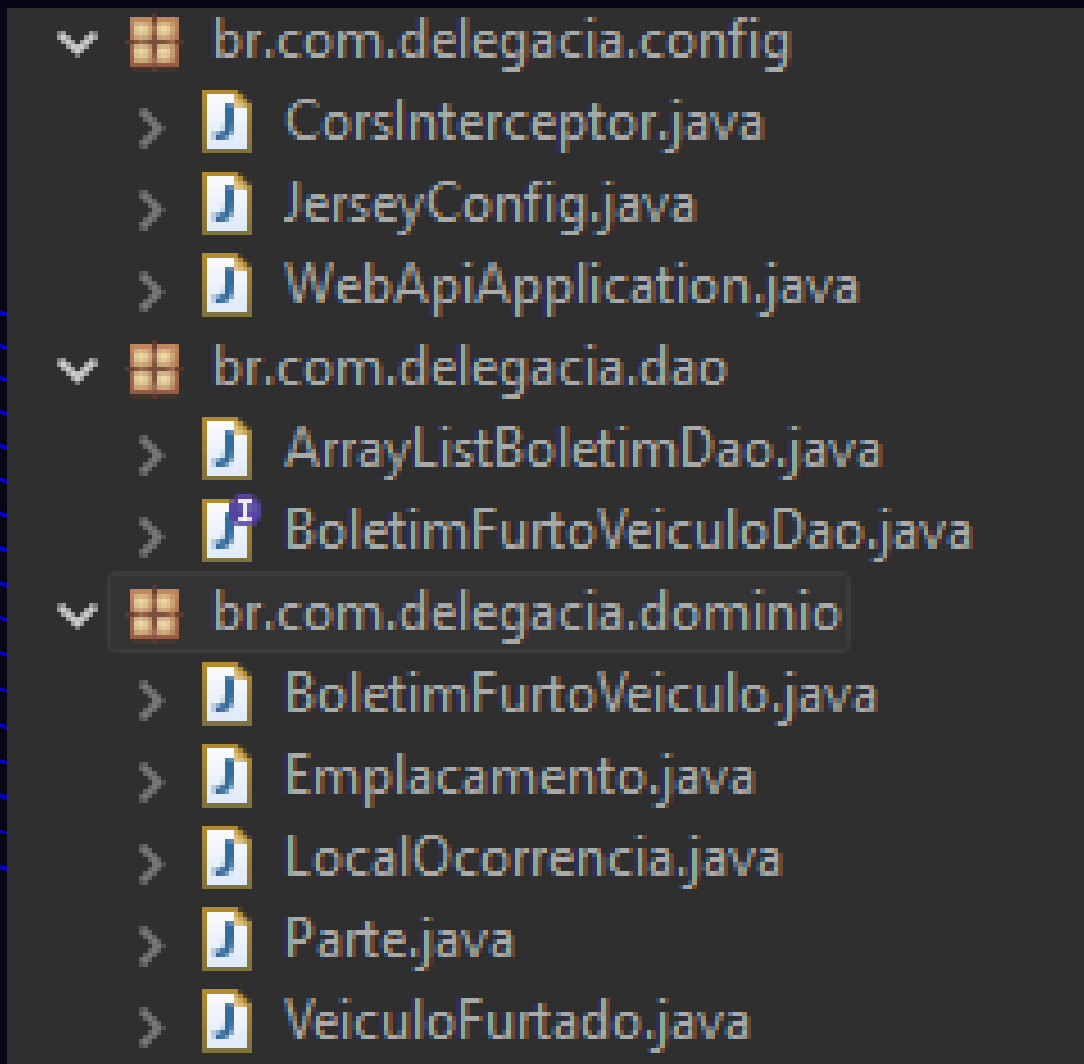
02



Estrutura de pacotes

Organização das classes em pacotes

03 Estrutura de pacotes



config

Reúne as classes referente a configuração da aplicação.

dao

Reúne a classe e a interface de persistência de dados.

dominio

Reúne as classes de domínio, que especifica quais são as informações manipuladas pelo sistema.

04 Estrutura de pacotes

```
▼ [icon] br.com.delegacia.endpoint
  > [icon] ListagemBoletinsEndpoint.java
  > [icon] ListagemVeiculosEndpoint.java
▼ [icon] br.com.delegacia.negocio
  > [icon] DadosObrigatoriosException.java
  > [icon] DataFormatoInvalidoException.java
  > [icon] EmailFormatoInvalidoException.java
  > [icon] GerenciadorBoletins.java
  > [icon] PlacaFormatoInvalidaException.java
  > [icon] RegrasBoletim.java
  > [icon] TelefoneFormatoInvalidoException.java
  > [icon] ValidadorBoletimVeiculo.java
  > [icon] Validadores.java
```

endpoint

Reúne as classes responsáveis por manipular as requisições feitas pelos usuários.

negocio

Reúne as classes e a interfaces de regras de negocios e as classes de tratamento de execução .

05

Detalhes

ACOPLAMENTO

Utilizamos as interfaces para comunicação entre camadas, deixando a aplicação com um acoplamento ideal, e o Spring Boot se responsabiliza por injetar cada classe em seu devido lugar, através da anotação @Autowired de injeção de dependência.

TRATAMENTO DE EXCEÇÕES

Desenvolvemos 5 classes que lançam exceções de origens diferentes, para ficar mais claro qual erro foi lançado, sendo elas:

- DadosObrigatoriosException
- EmailFormatoInvalidoException
- PlacaFormatoInvalidaException
- TelefoneFormatoInvalidoException
- DataFormatoInvalidoException

MODELAGEM

Utilizamos a modelagem sugerida no enunciado do trabalho, porém eliminamos o tipo de envolvimento da classe Parte.

06 Modelagem do domínio da Aplicação



SEPARAÇÃO DAS RESPONSABILIDADES

Além da separação da aplicação entre camadas, fizemos a separação de responsabilidades por classes do tipo `exception` e `validators`, utilizadas pelas regras de negócio quando necessárias

GRAU DE DETALHE DAS REGRAS DE NEGÓCIO

As regras que definimos para o cadastramento adequado de um boletim de ocorrência na aplicação foram:

- Formato da data
- Formato da placa do carro
- Formato do e-mail da parte
- Formato do telefone da parte
- Todos os dados são obrigatórios

VALIDAÇÃO

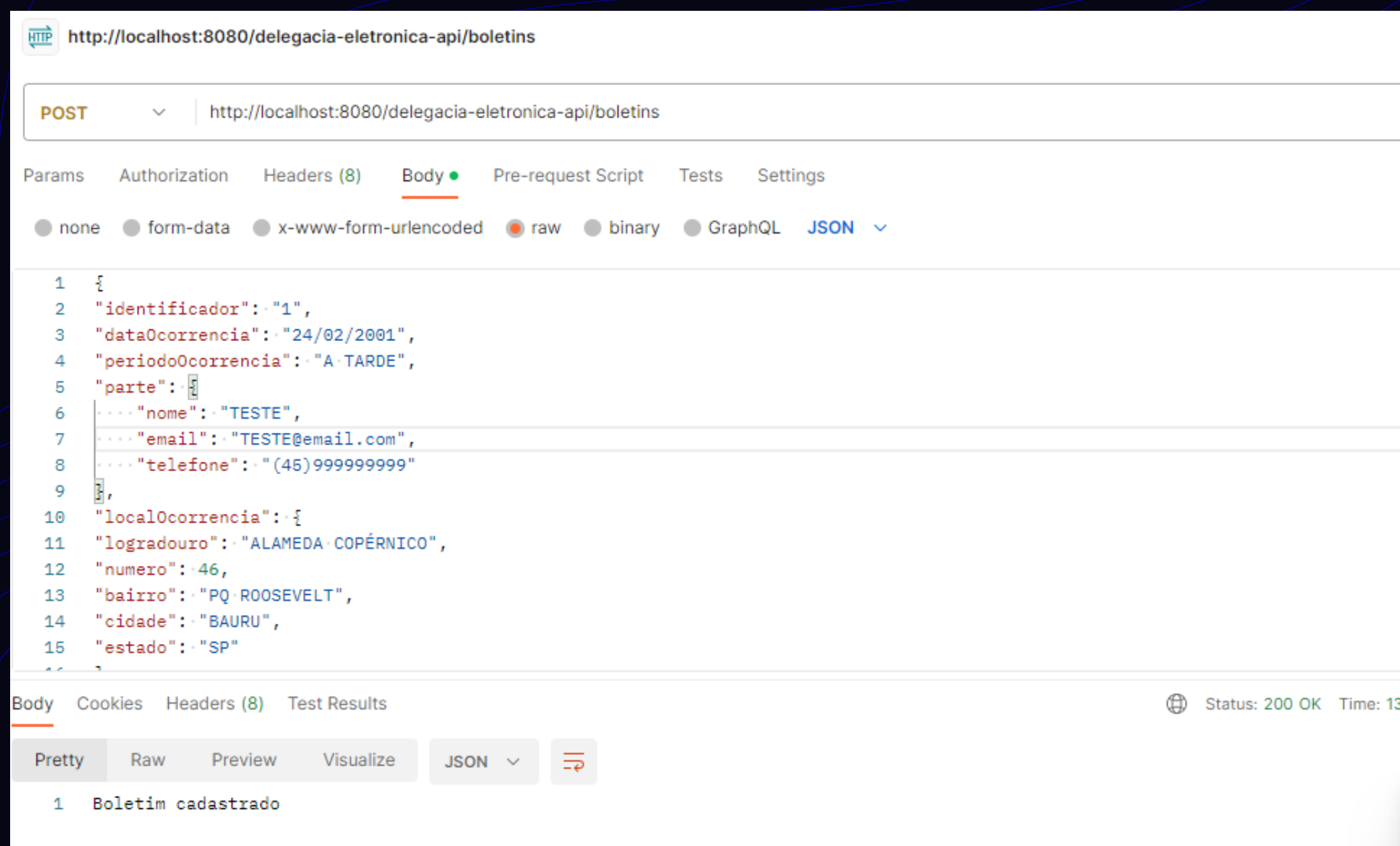
Os formatos são validados usando `regex` (regular expression) e os dados obrigatórios são verificados por um método que compara propriedade por propriedade verificando se nenhuma delas está nula ou vazia

09

ENDPOINTS - BOLETIM

POST

<http://localhost:8080/delegacia-eletronica-api/boletins>

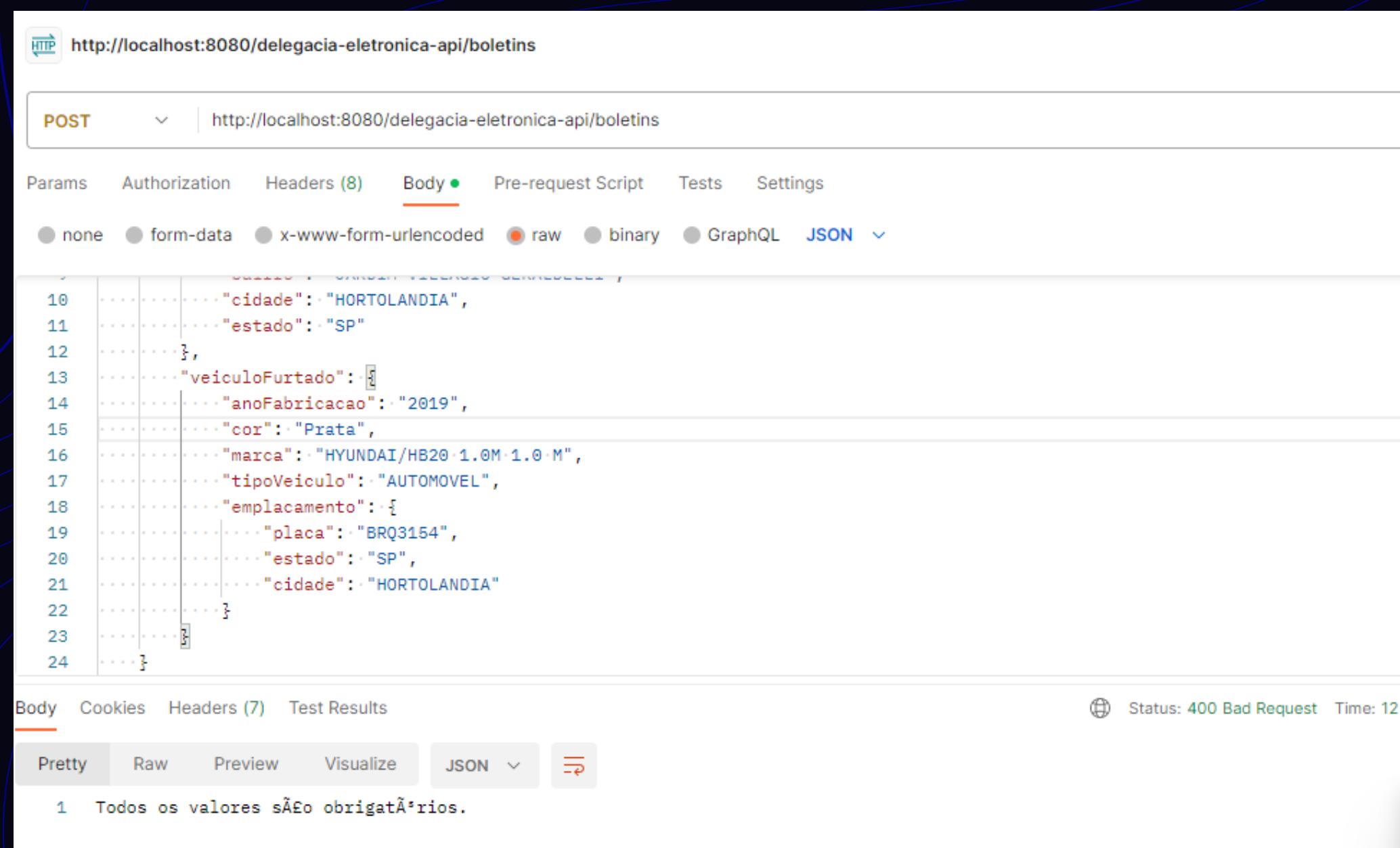


10

ENDPOINTS - BOLETIM

POST - TRATAMENTO DE EXCEÇÃO

<http://localhost:8080/delegacia-eletronica-api/boletins>



11

ENDPOINTS - BOLETIM

GET

`http://localhost:8080/delegacia-eletronica-api/boletins?`
`identificador=ID&periodoOcorrencia=PERIODO&cidade=CIDADE`

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

- URL:** `http://localhost:8080/delegacia-eletronica-api/boletins?identificador=2431/2023&periodoOcorrencia=A NOITE&cidade=HORTOLANDIA`
- Method:** GET
- Query Params Table:**

| | Key | Value | Description |
|-------------------------------------|-------------------|-------------|-------------|
| <input checked="" type="checkbox"/> | identificador | 2431/2023 | |
| <input checked="" type="checkbox"/> | periodoOcorrencia | A NOITE | |
| <input checked="" type="checkbox"/> | cidade | HORTOLANDIA | |

Body: The response is in JSON format, displayed in 'Pretty' view:

```
1 {
2   "identificador": "2431/2023",
3   "dataOcorrencia": "31/12/2022",
4   "periodoOcorrencia": "A NOITE",
5   "parte": null,
6   "localOcorrencia": {
7     "logradouro": "RUA DE OLIVEIRA ",
8     "numero": "80",
9     "bairro": "JARDIM VILLAGIO GERALDELLI",
10    "cidade": "HORTOLANDIA",
11    "estado": "SP"
12  }
```

FILTROS

- identificador
- periodoOcorrencia
- cidade

12

ENDPOINTS - BOLETIM

DELETE

<http://localhost:8080/delegacia-eletronica-api/boletins?identificador=ID>

HTTP <http://localhost:8080/delegacia-eletronica-api/boletins?identificador=102/2023>

DELETE <http://localhost:8080/delegacia-eletronica-api/boletins?identificador=102/2023>

Params ☒ Authorization Headers (8) Body ☒ Pre-request Script Tests Settings

Query Params

| | Key | Value | Description |
|-------------------------------------|---------------|----------|-------------|
| <input checked="" type="checkbox"/> | identificador | 102/2023 | |
| | Key | Value | Description |

Body Cookies Headers (8) Test Results 🌐 Status: 200 OK Time: 17 m

Pretty Raw Preview Visualize Text ☒

```
1 Boletim Deletetado
```

13

ENDPOINTS - BOLETIM

PUT

<http://localhost:8080/delegacia-eletronica-api/boletins>

The screenshot displays a REST client interface for a PUT request to the endpoint `http://localhost:8080/delegacia-eletronica-api/boletins`. The request body is a JSON object with the following structure:

```
1 {
2   "identificador": "1",
3   "dataOcorrencia": "24/02/2001",
4   "periodoOcorrencia": "A TARDE",
5   "parte": {
6     "nome": "teste1",
7     "email": "TESTE@email.com",
8     "telefone": "(45)999999999"
9   },
10  "localOcorrencia": {
```

The response status is `200 OK` with a time of `12 ms`. The response body is `Boletim atualizado`.

14

ENDPOINTS - VEICULOS

GET

http://localhost:8080/delegacia-eletronica-api/veiculos?
placa=PLACA&cor=COR&tipoVeiculo=VEICULO

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

- URL:** `http://localhost:8080/delegacia-eletronica-api/veiculos?placa=BRQ3154&cor=Prata&tipoVeiculo=AUTOMOVEL`
- Method:** GET
- Query Params:**

| Key | Value | Description |
|-------------|-----------|-------------|
| placa | BRQ3154 | |
| cor | Prata | |
| tipoVeiculo | AUTOMOVEL | |
- Body:** Pretty JSON view showing the response:

```
1 {
2   "anoFabricacao": "2019",
3   "cor": "Prata",
4   "marca": "HYUNDAI/HB20 1.0M 1.0 M",
5   "tipoVeiculo": "AUTOMOVEL",
6   "emplacamento": {
7     "placa": "BRQ3154",
8     "estado": "SP",
9     "cidade": "HORTOLANDIA"
10  }
11 }
12 }
```
- Status:** 200 OK

FILTROS

- Placa
- Cor
- Tipo de veiculo

15



200
OK



400
Bad Request



404
Not Found

HTTP STATUS
CODES