

API Campanhas:

- **INCLUIR CAMPANHA**

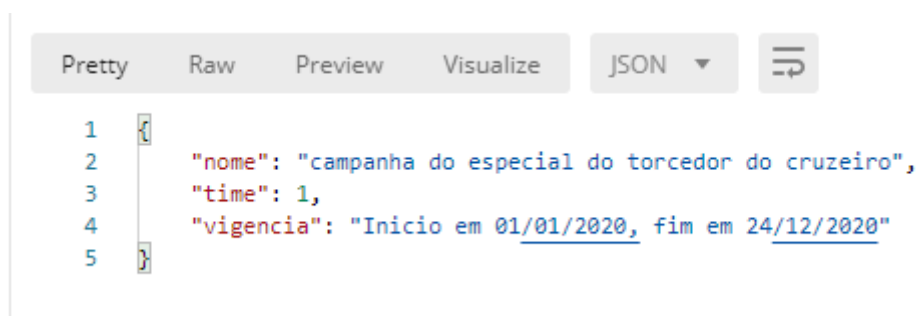
Uma campanha pode ser incluída através da seguinte estrutura:



Onde: o ID é omitido, uma vez que é gerado automaticamente. O início e o fim são variáveis criadas globalmente pelo um script no postman, conforme consta abaixo:



A resposta da requisição é recebida na seguinte estrutura, conforme solicitado:



Abaixo podemos ver o registro na tabela do banco de dados:

Data Output Explain Messages Notifications					
	id [PK] bigint	fim timestamp without time zone	inicio timestamp without time zone	nome character varying (255)	time bigint
1	11	2020-12-24 01:00:00	2020-01-01 01:00:00	campanha do especial do torc...	1

A especificação define que quando uma nova campanha estiver sendo cadastrada e a vigência for igual a de uma já cadastradas, deve-se acrescentar um dia ao final do período. Este caso é exposto abaixo:

Considere que na base na base de dados não existe nenhum registro:

Data Output Explain Messages Notifications					
	id [PK] bigint	fim timestamp without time zone	inicio timestamp without time zone	nome character varying (255)	time bigint

Considere também a seguinte campanha vai ser cadastrada:

POST

localhost:8080/api/campanha

Params

Authorization

Headers (9)

Body

```

1  var inicio = new Date(2020,0,1).getTime
2  var fim = new Date(2021, 0, 12).getTime
3  pm.globals.set("inicio", inicio)
4  pm.globals.set("fim", fim)
5

```

```

{
  "nome": "campanha do especial do torcedor do cruzeiro de 2021 atualizada",
  "time": 1,
  "inicio": {{inicio}},
  "fim": {{fim}}
}

```

Ao ser inserido pela primeira vez, é devolve ele mesmo, no padrão que deve ser apresentado:

POST localhost:8080/api/campanha

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "nome": "campanha do especial do torcedor do cruzeiro de 2021 atualizada",
3   "time": 1,
4   "inicio": {{inicio}},
5   "fim": {{fim}}
6 }
7
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "nome": "campanha do especial do torcedor do cruzeiro de 2021 atualizada",
3   "time": 0,
4   "vigencia": "Inicio em 01/01/2020, fim em 12/01/2021"
5 }
```

Ao realizar a requisição novamente, ele deve atualizar o período que foi inserido anteriormente:

POST localhost:8080/api/campanha

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "nome": "campanha do especial do torcedor do cruzeiro de 2021 atualizada",
3   "time": 1,
4   "inicio": {{inicio}},
5   "fim": {{fim}}
6 }
7
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   {
3     "nome": "campanha do especial do torcedor do cruzeiro de 2021 atualizada",
4     "time": 1,
5     "vigencia": "Inicio em 01/01/2020, fim em 13/01/2021"
6   }
7 }
```

Podemos identificar que ele fez o processo de acrescentar um dia no final no período que já existia, podemos confirmar na consulta:

```
3
4 SELECT * FROM TB_CAMPANHA;
```

	id	fim	inicio
	[PK] bigint	timestamp without time zone	timestamp without time zone
1	38	2021-01-12 01:00:00	2020-01-01 01:00:00
2	37	2021-01-13 01:00:00	2020-01-01 01:00:00

Ao realizar o procedimento mais uma vez, teremos:

14 `SELECT * FROM TB_CAMPANHA;`

	id [PK] bigint	fim timestamp without time zone	inicio timestamp without time zone
1	39	2021-01-12 01:00:00	2020-01-01 01:00:00
2	38	2021-01-13 01:00:00	2020-01-01 01:00:00
3	37	2021-01-14 01:00:00	2020-01-01 01:00:00

dy Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "nome": "campanha do especial do torcedor do cruzeiro de 2021 atualizada",
4     "time": 1,
5     "vigencia": "Inicio em 01/01/2020, fim em 13/01/2021"
6   },
7   {
8     "nome": "campanha do especial do torcedor do cruzeiro de 2021 atualizada",
9     "time": 1,
10    "vigencia": "Inicio em 01/01/2020, fim em 14/01/2021"
11  }
12 ]
```

O teste mostrou que foi produzido o resultado esperado

- **CONSULTAR**

Foram inseridos os registros abaixo:

	id [PK] bigint	fim timestamp without time zone	inicio timestamp without time zone	nome character varying (255)	time bigint
1	11	2020-12-24 01:00:00	2020-01-01 01:00:00	campanha do especial do torcedor do cruzeiro	1
2	12	2020-05-24 00:00:00	2020-01-01 01:00:00	campanha do especial do torcedor do cruzeiro	1
3	13	2020-10-24 00:00:00	2020-01-01 01:00:00	campanha do especial do torcedor do cruzeiro	1

Ao realizar a consulta das campanhas, através da API, recebemos o seguinte resultado:

```
GET localhost:8080/api/campanhas

Params Authorization Headers (6) Body Pre-request Script Test Results

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

1 [
2   {
3     "nome": "campanha do especial do torcedor do cruzeiro",
4     "time": 1,
5     "vigencia": "Inicio em 01/01/2020, fim em 24/12/2020"
6   },
7   {
8     "nome": "campanha do especial do torcedor do cruzeiro",
9     "time": 1,
10    "vigencia": "Inicio em 01/01/2020, fim em 24/10/2020"
11  }
12 ]
```

O retorno apresenta apenas 2 campanhas, uma vez que a campanha de ID 12, com vencimento em 20/05/2020, está vencida.

A API também oferece suporte para consulta de determinada campanha pelo ID, conforme mostrado abaixo:

```
GET localhost:8080/api/campanha/13

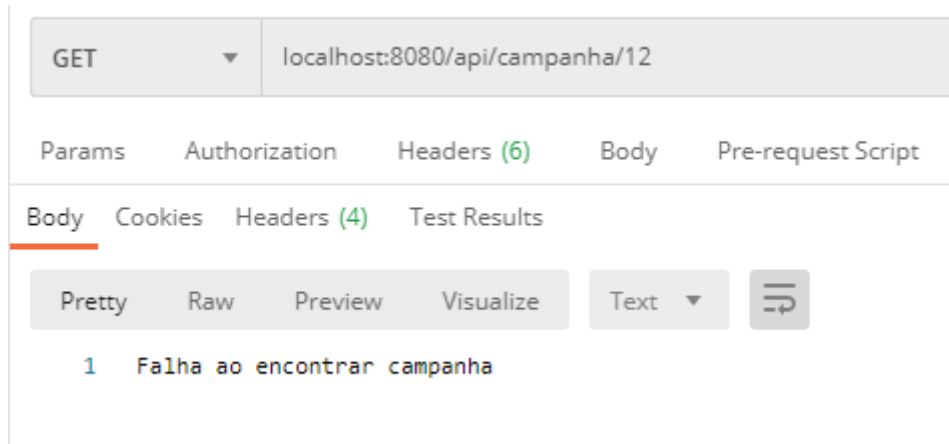
Params Authorization Headers (6) Body Pre-request Script Test Results

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

1 {
2   "nome": "campanha do especial do torcedor do cruzeiro",
3   "time": 1,
4   "vigencia": "Inicio em 01/01/2020, fim em 24/10/2020"
5 }
```

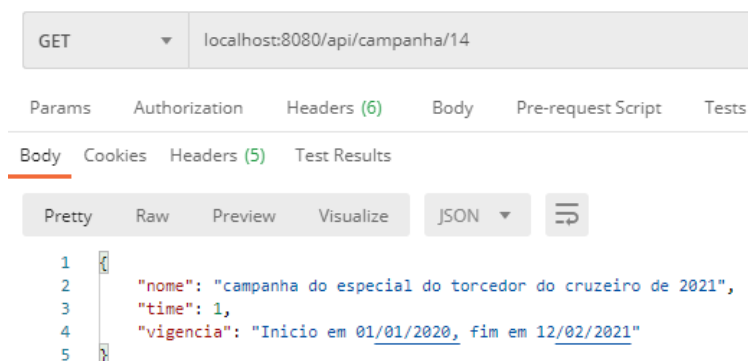
No caso da campanha com ID 12, cuja vigência está vencida, o sistema retorna uma mensagem informando que não foi possível encontrar a campanha:



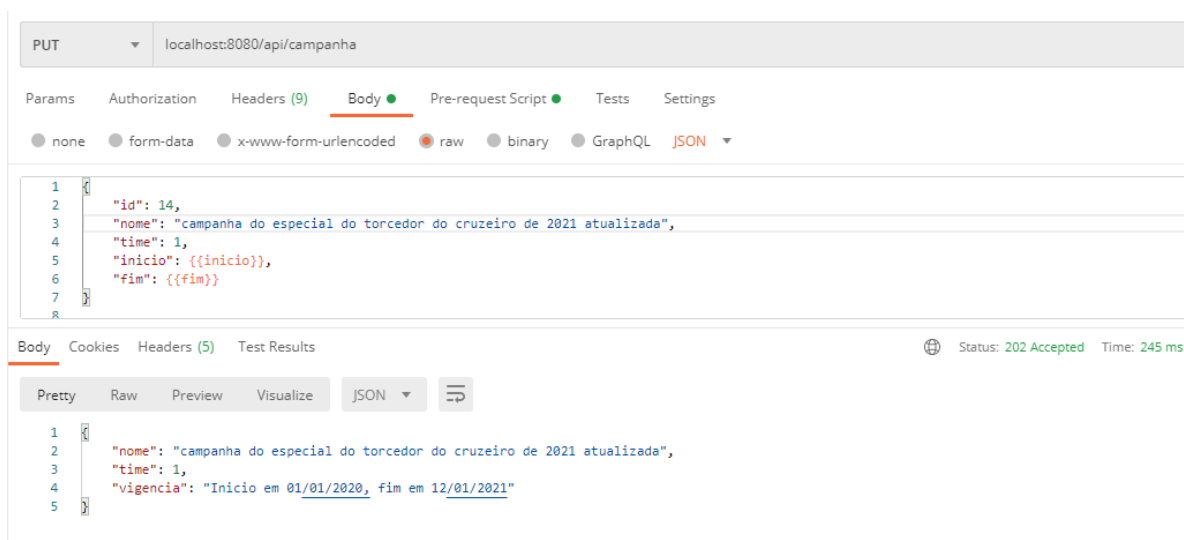
- **ATUALIZAR**

A API também possibilita que um registro seja editado.

Consideremos o seguinte registro, campanha de ID 14 que atualmente está neste estado:

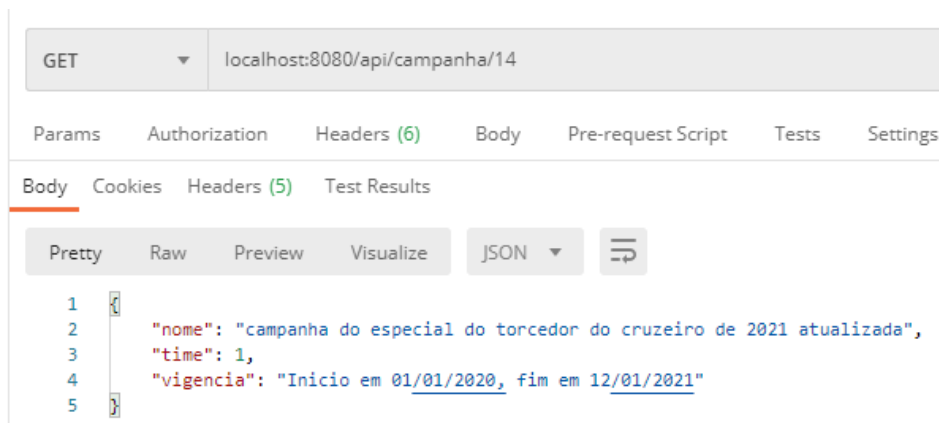


Vamos atualizar o registro alterando o nome e a data final da vigência, pronto!



A campanha foi atualizada e foi retornada a forma que a mesma ficou salva na base de dados.

Para ter certeza, podemos consultar pelo ID:

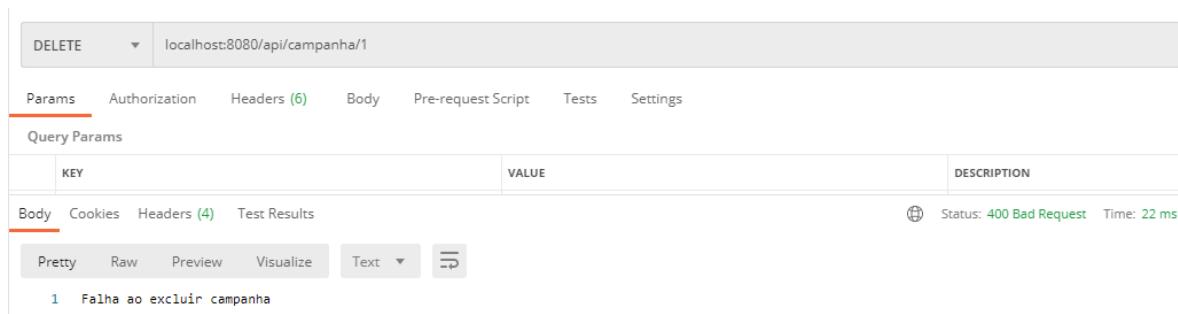


- **EXCLUIR**

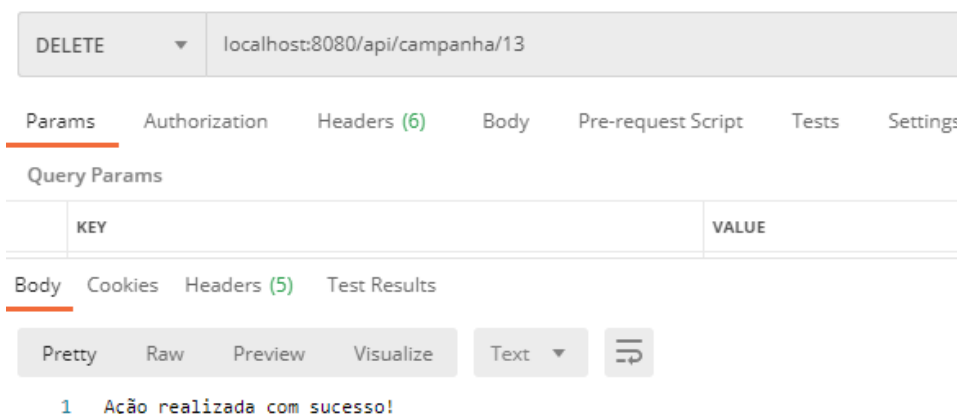
Conforme solicitado, a API oferece a possibilidade de apagar campanha da seguinte forma:

URI: localhost:8080/api/campanha/id

Exemplo: Ao selecionar a opção de excluir a campanha de ID 1, o sistema irá retornar uma mensagem de erro, um vez que tal registro no existe no banco de dados:



Ao passo que se optarmos por excluir a campanha de ID 13 que existe na base de dados:



Para conferir o resultado, basta usar o método de consulta por ID:

GET localhost:8080/api/campanha/13

Params Authorization Headers (6) Body Pre-reqs

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize Text

1 Falha ao encontrar campanha

13
14 `SELECT * FROM TB_CAMPANHA;`

Data Output Explain Messages Notifications

	id [PK] bigint	fin timestamp without time zone	inicio timestamp without time zone	nome character varying (255)
1	11	2020-12-24 01:00:00	2020-01-01 01:00:00	campanha do especial do torcedor do cruzeiro

Logo a ação foi realizada conforme esperado.

API Clientes

Ao consultar os clientes cadastrados recebemos a seguinte mensagem:

GET localhost:8080/api/clientes

Params Authorization Headers (6) Body Pre-request Sc

Query Params

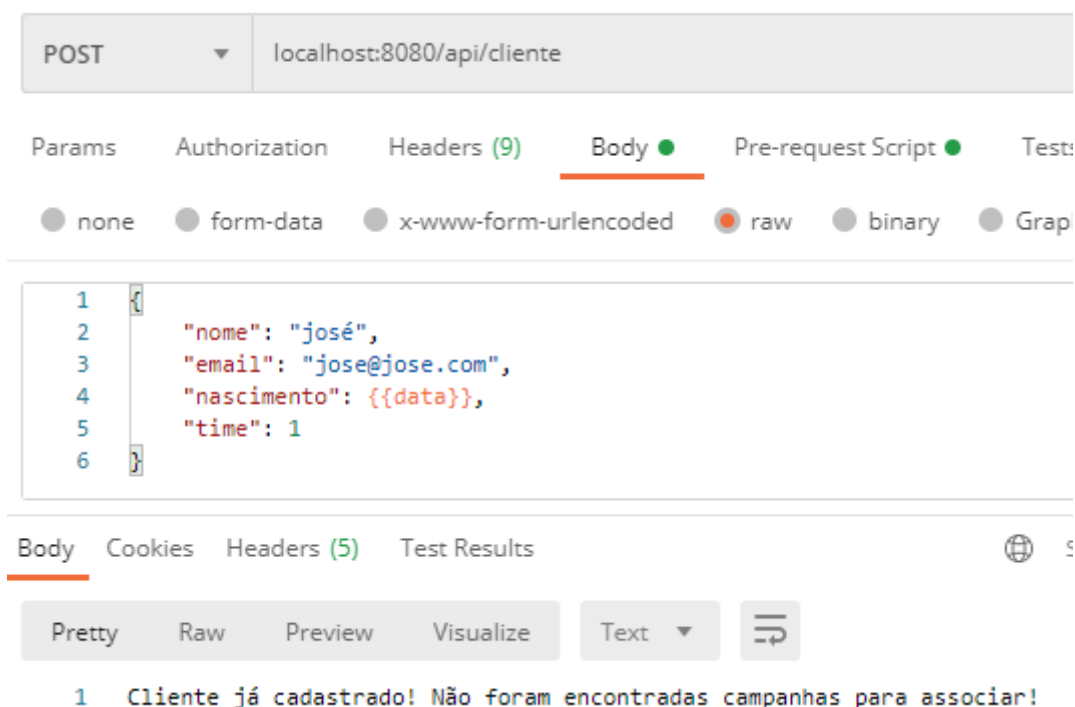
KEY	VALUE
Key	Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

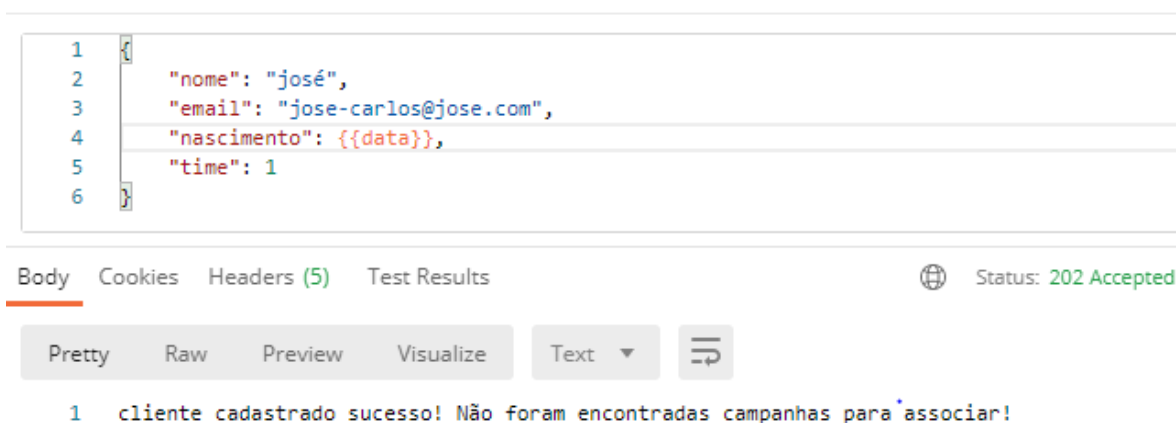
```
1 [
2   {
3     "id": 73,
4     "nome": "josé",
5     "email": "jose@jose.com",
6     "nascimento": "2019-06-05T03:00:00.000+00:00",
7     "time": 1
8   }
9 ]
```


Ao usar o serviço de cadastrar clientes informando o mesmo e-mail do cliente consultado, recebemos a seguinte mensagem, indicando que o cliente já está cadastrado:



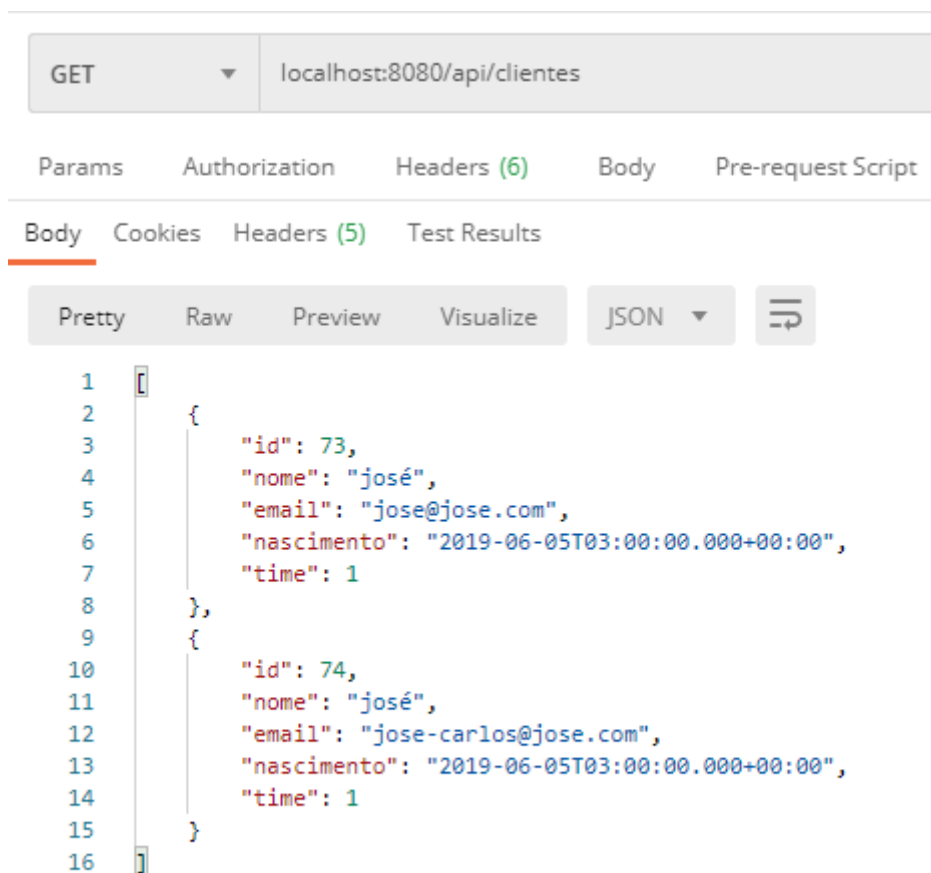
Também somos informados que não existem campanhas para associar, segundo a consulta que foi feita na API campanhas. Caso exista campanhas, são associadas todas que ainda não estão vinculadas ao cliente.

Ao informar um cliente com e-mail diferente, somos informados que o cliente foi cadastrado:



Obs: quando existirem campanhas cujo time seja o mesmo que o do cliente, estão serão associadas no momento do cadastro para novos clientes, ou serão atualizadas quando um cliente já cadastrado usar este serviço.

Ao realizar a consulta recebemos o JSON com os dois clientes que foram cadastrados:



The screenshot displays a REST client interface. At the top, a GET request is configured to `localhost:8080/api/clientes`. Below the request bar, tabs for Params, Authorization, Headers (6), Body, and Pre-request Script are visible. The 'Body' tab is selected, showing a JSON response. Within the 'Body' tab, sub-tabs for Pretty, Raw, Preview, and Visualize are present, with 'Pretty' being active. The JSON response is displayed in a formatted, color-coded manner, showing an array of two client objects. The first object has an id of 73, name 'José', email 'jose@jose.com', and a birthdate of '2019-06-05T03:00:00.000+00:00'. The second object has an id of 74, name 'José', email 'jose-carlos@jose.com', and the same birthdate. Both objects have a 'time' field set to 1.

```
1  [
2    {
3      "id": 73,
4      "nome": "José",
5      "email": "jose@jose.com",
6      "nascimento": "2019-06-05T03:00:00.000+00:00",
7      "time": 1
8    },
9    {
10     "id": 74,
11     "nome": "José",
12     "email": "jose-carlos@jose.com",
13     "nascimento": "2019-06-05T03:00:00.000+00:00",
14     "time": 1
15   }
16 ]
```