



Documentação REST do Uash App

Grupo 09 / Marco Antônio Camargo (20211019) / Carlyne Melo (20210046) / Diogo Carvalho (20210008)

Uash App é um aplicativo de requisição de lavagens de automóveis, com usuários (users) que podem requisitar lavagens ou prestar essas como “uashers”. Usuários podem ter veículos, e veículos podem ter vários donos. Um uasher é um usuário com atributos e métodos adicionais, e possui uma lista de suas lavagens decorrentes / concluídas. Uma lavagem possui, obrigatoriamente, um veículo para lavagem; também irá possuir um uasher quando um demonstrar interesse e decidir realizar a lavagem.

Erros “default”:

400 - Bad Request

404 - Not Found

405 - Method Not Allowed

500 - Internal Server Error

Usuários (/api/users)

Exibir todos os Users (get): /api/users/

Sem parâmetros adicionais.

Devolve uma tabela com todos os usuários e seus dados, exceto suas senhas.

Resultado Esperado / Exemplo

http://localhost:8080/api/users

GET	http://localhost:8080/api/users
Params	Authorization
Headers (6)	Body
Pre-request Script	Tests
Settings	

Query Params

KEY
Key

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "name": "Carol",
4     "id": 1,
5     "localizacao": "Lisboa",
6     "email": "carolyne@gmail.com",
7     "telefone": 258741963,
8     "dataNasc": "2003-08-25"
9   },
10  {
11    "name": "Marco",
12    "id": 2,
13    "localizacao": "Oeiras",
14    "email": "marco@gmail.com",
15    "telefone": 236547891,
16    "dataNasc": "2012-05-12"
17  },
18  {
19    "name": "Diogo",
20    "id": 3,
21    "localizacao": "Cascais",
22    "email": "diogo@gmail.com",
23    "telefone": 214785369,
24    "dataNasc": "2016-01-02"
25  }
26 ]
```

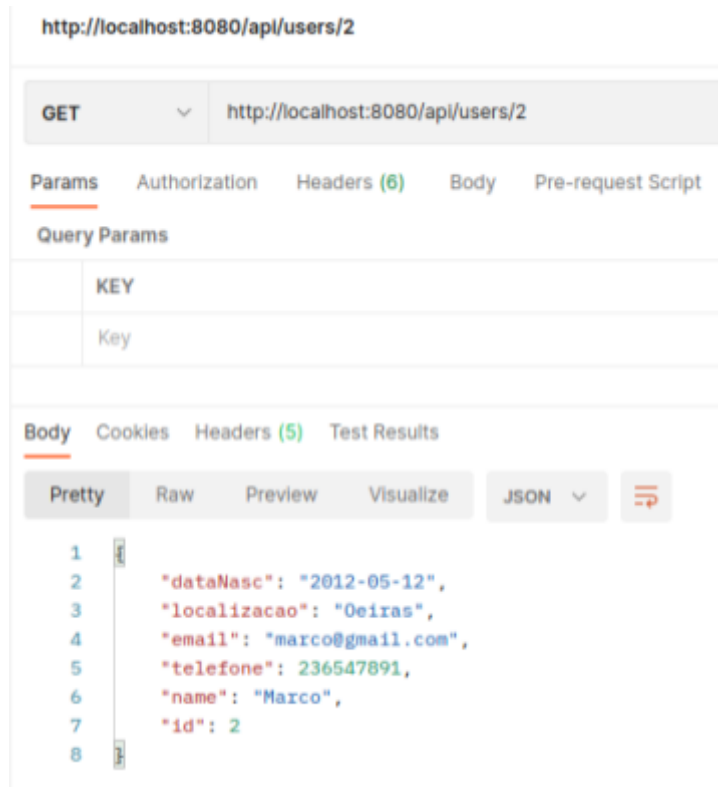
Sem códigos de erro programados

Exibir User por Id (get): /api/users/{id:[0-9]+}

Id: O Id do User desejado.

Devolve os dados do usuário desejado, exceto a senha.

Resultado Esperado / Exemplo



Sem códigos de erro programados

Adicionar User (post): /api/users/

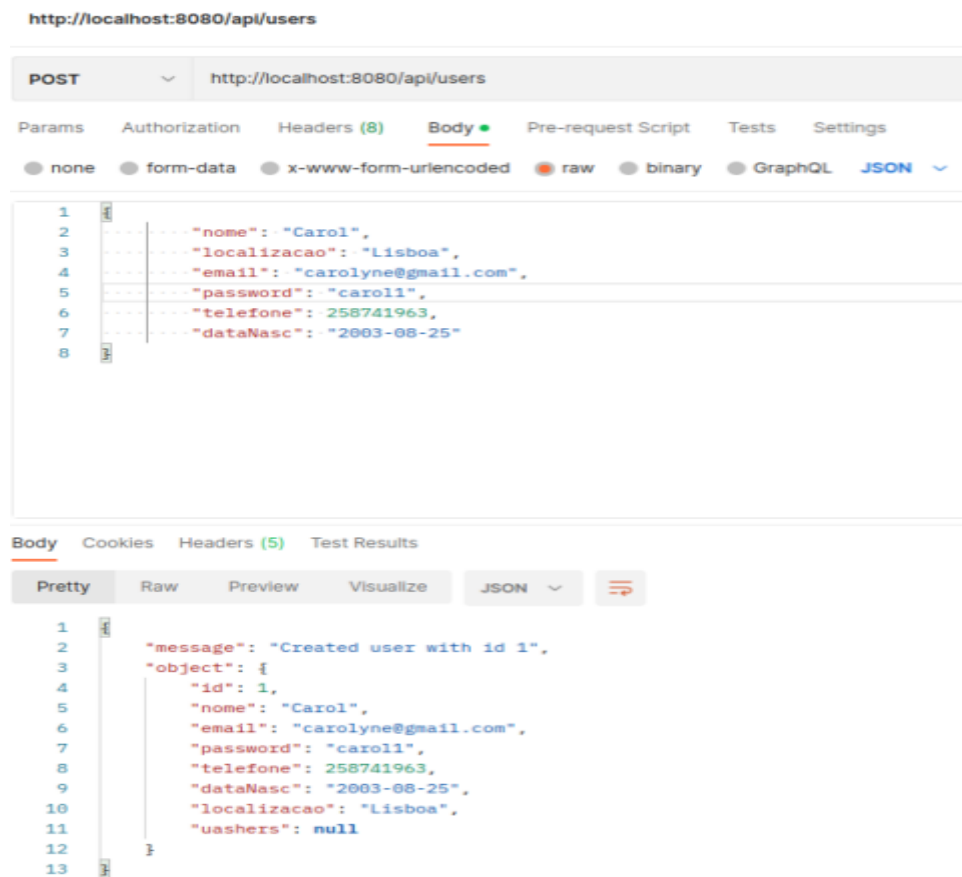
Sem parâmetros adicionais. Um JSONObject User deverá ser providenciado.

Resultado Esperado

```
{  "message": "Created user with id 1",  "object": {    "id": 1,    "nome": "Carol",    "email": "carolyne@gmail.com",    "password": "carol1",    "telefone": 258741963,    "dataNasc": "2003-08-25",    "localizacao": "Lisboa",    "uashers": null  }}
```

Sem códigos de erro programados

Exemplo



Atualizar User (put): /api/users/{id:[0-9]+}

Id: O Id do User desejado.

Atualiza o usuário com os dados providenciados, que devem ser em formato JSONArray.

Resultado Esperado

```
{
  "message": "Updating user with id 1",
  "object": {
    "id": 2,
    "nome": "MarcoAntonio",
    "email": "marcoantonio@gmail",
    "password": "marco45",
    "telefone": 210003654,
    "dataNasc": "2019-08-04",
    "localizacao": "Sintra", // Era "Oeiras"
    "uashers": null
  }
}
```

Sem códigos de erro programados

Exemplo

http://localhost:8080/api/users/2

PUT http://localhost:8080/api/users/2

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

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

```
1 {
2   "nome": "MarcoAntonio",
3   "email": "marcoantonio@gmail",
4   "password": "marco45",
5   "telefone": 210003654,
6   "dataNasc": "2019-08-04",
7   "localizacao": "Sintra"
8 }
```

body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

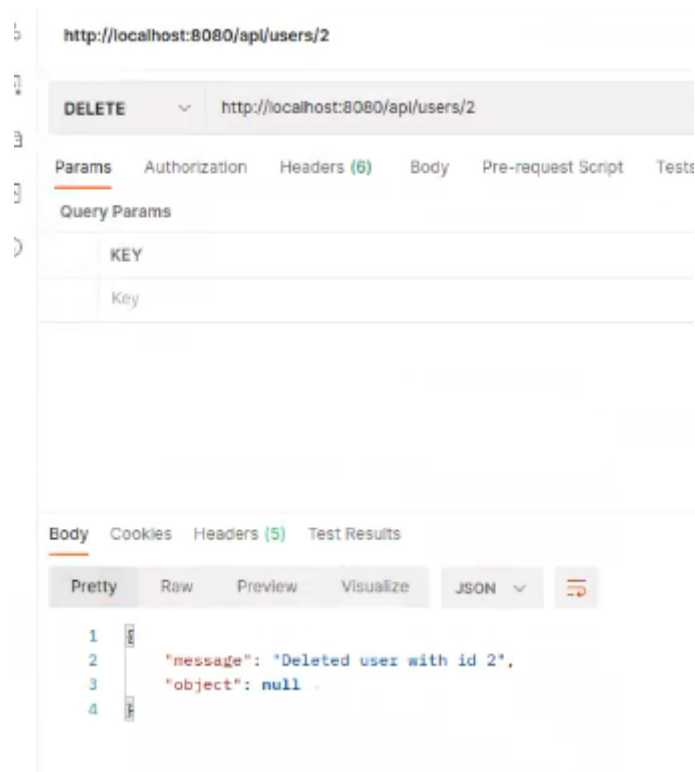
```
1 {
2   "message": "Updating user with id 2",
3   "object": {
4     "id": 2,
5     "nome": "MarcoAntonio",
6     "email": "marcoantonio@gmail",
7     "password": "marco45",
8     "telefone": 210003654,
9     "dataNasc": "2019-08-04",
10    "localizacao": "Sintra",
11    "uashers": null
12  }
13 }
```

Apagar User (delete): /api/users/{id:[0-9]+}

Id: O Id do User desejado.

Apaga o usuário desejado.

Resultado Esperado / Exemplo



Sem códigos de erro programados

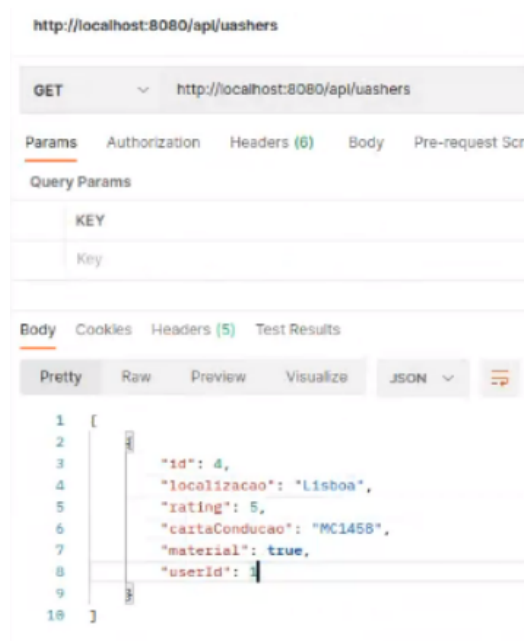
Uashers (/api/uashers)

Exibir todos os Uashers (get): /api/uashers/

Sem parâmetros adicionais.

Devolve uma tabela com todos os uashers e seus dados.

Resultado Esperado / Exemplo



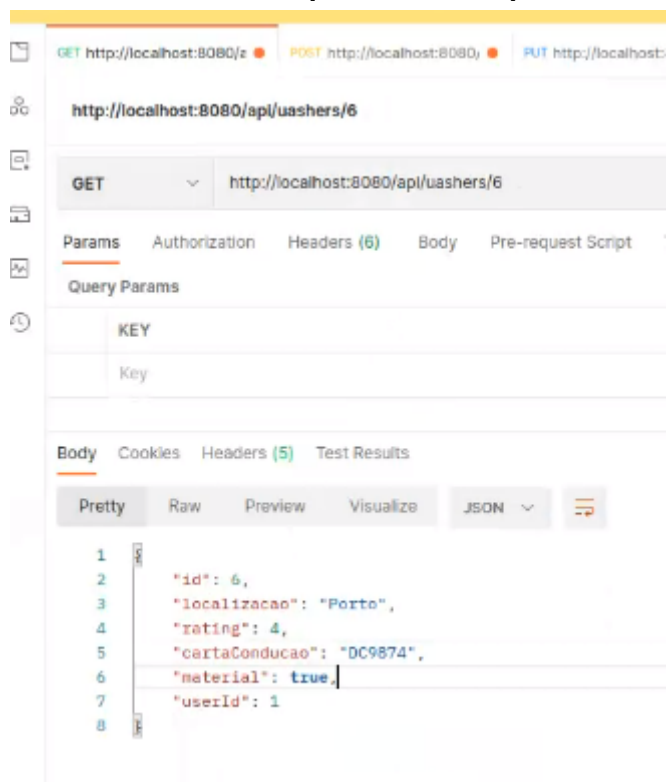
Sem códigos de erro programados

Exibir Uasher por Id (get): /api/uashers/{id:[0-9]+}

Id: O Id do Uasher desejado.

Devolve os dados do uasher desejado.

Resultado Esperado / Exemplo



Sem códigos de erro programados

Adicionar Uasher (post): /api/uashers/

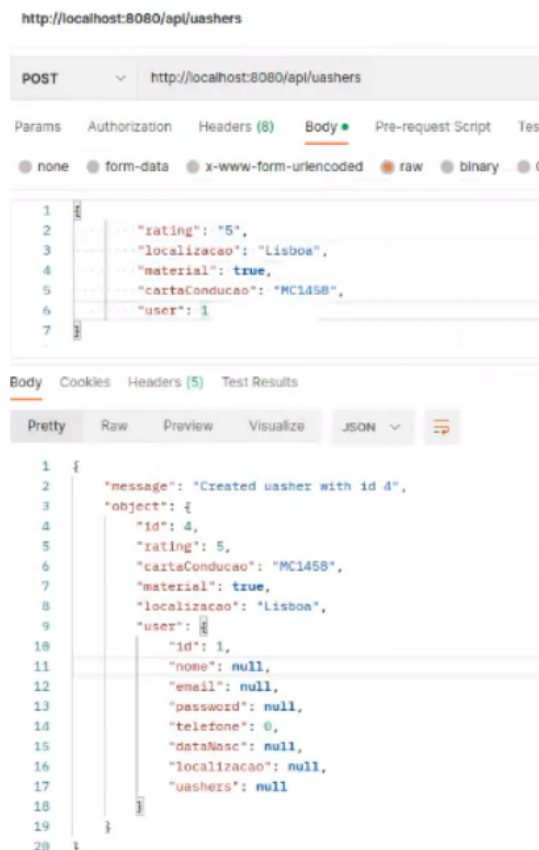
Sem parâmetros adicionais. Um JSONObject Uasher deverá ser providenciado.

Resultado Esperado

```
{
  "message": "Created uasher with id 4",
  "object": {
    "id": 4,
    "rating": 5,
    "cartaConducao": "MC1458",
    "material": true,
    "localizacao": "Lisboa",
    "user": { ... }
  }
}
```

Sem códigos de erro programados

Exemplo



Atualizar Uasher (put): /api/uashers/{id:[0-9]+}

Id: O Id do Uasher desejado.

Atualiza o uasher com os dados providenciados, que devem ser em formato JSONArray.

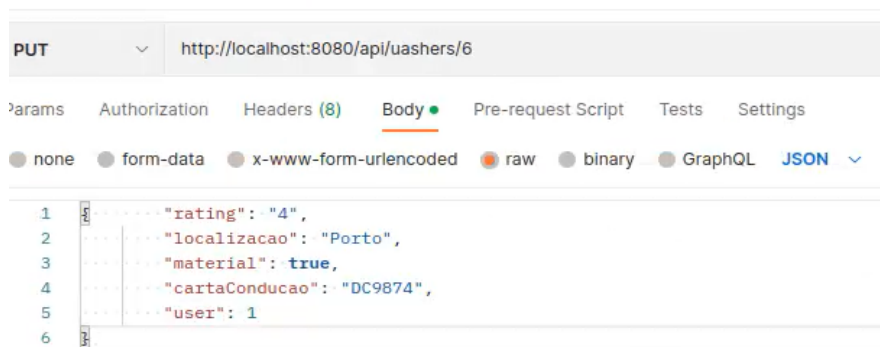
Resultado Esperado

```
{
  "message": "Updating uasher with id 6",
  "object": {
    "id": 6,
    "rating": 4,
    "cartaConducao": "DC9874",
    "material": true,
    "localizacao": "Porto",
    "user": { ... }
  }
}
```

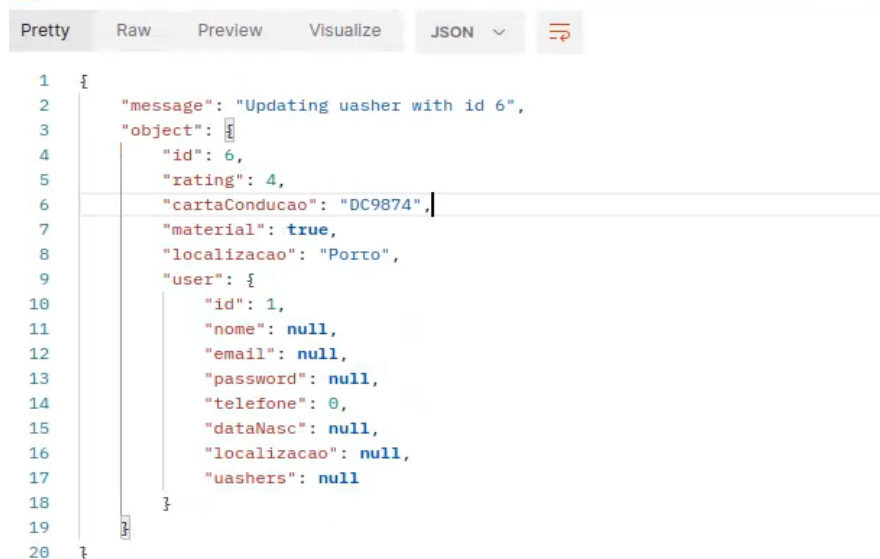
Sem códigos de erro programados

Exemplo

http://localhost:8080/api/uashers/6



body Cookies Headers (5) Test Results



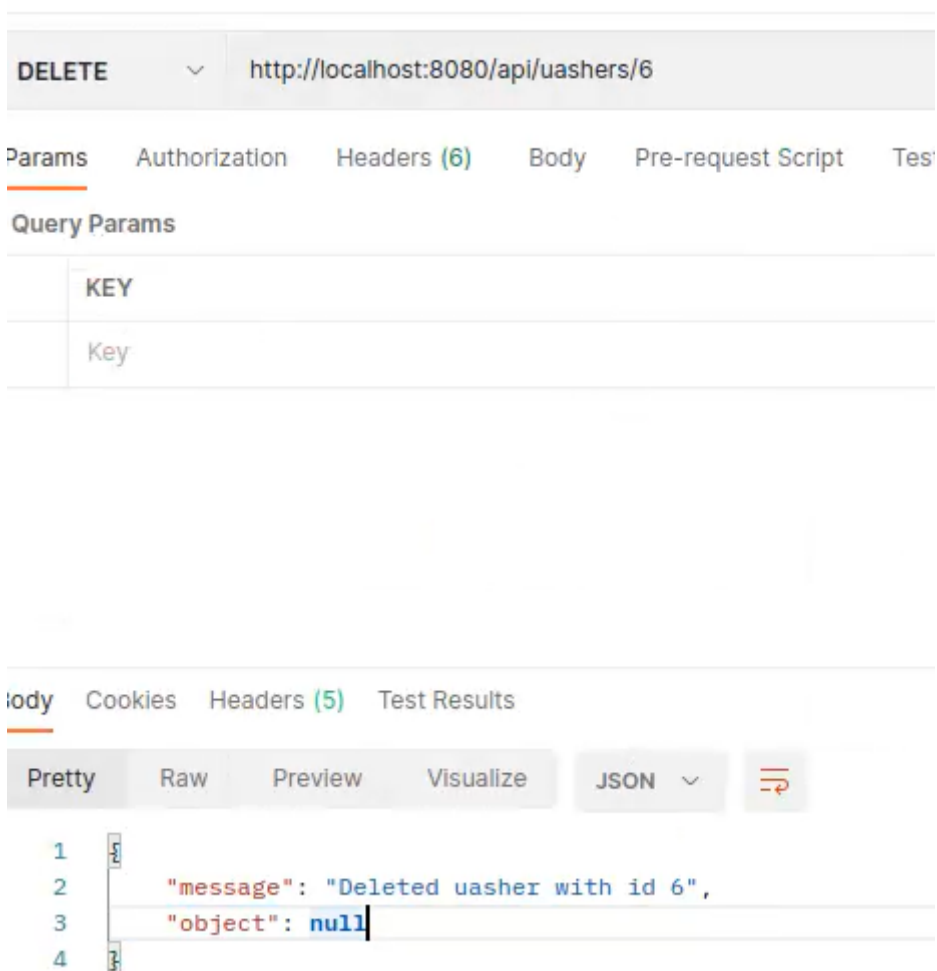
Apagar Uasher (delete): /api/uashers/{id:[0-9]+}

Id: O Id do Uasher desejado.

Apaga o uasher desejado.

Resultado Esperado / Exemplo

http://localhost:8080/api/uashers/6



Sem códigos de erro programados

Veículos (/api/veiculos)

Exibir todos os Veiculos (get): /api/veiculos/

Sem parâmetros adicionais.

Devolve uma tabela com todos os veículos e seus dados.

Resultado Esperado / Exemplo

The screenshot shows a REST client interface with a GET request to `http://localhost:8080/api/veiculos`. The response is displayed in the 'Body' tab, showing a JSON array with one object. The object contains the following fields: `id` (2), `tipo` (Fiat Tipo), `localizacao` (Lisboa), and `matricula` (CM45SS).

```
GET http://localhost:8080/api/veiculos
```

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

Query Params

KEY
Key

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 2,
4     "tipo": "Fiat Tipo",
5     "localizacao": "Lisboa",
6     "matricula": "CM45SS"
7   }
8 ]
```

Sem códigos de erro programados

Exibir Veiculo por Id (get): /api/veiculos/{id:[0-9]+}

Id: O Id do Veiculo desejado.

Devolve os dados do veículo desejado.

Exemplo

GET http://localhost:8080/api/veiculos/2

Params Authorization Headers (6) Body Pre-request Scripts

Query Params

KEY
Key

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 2,
3   "tipo": "Fiat Tipo",
4   "localizacao": "Lisboa",
5   "matricula": "CM45SS"
6 }
```

Adicionar Veiculo (post): /api/veiculos/

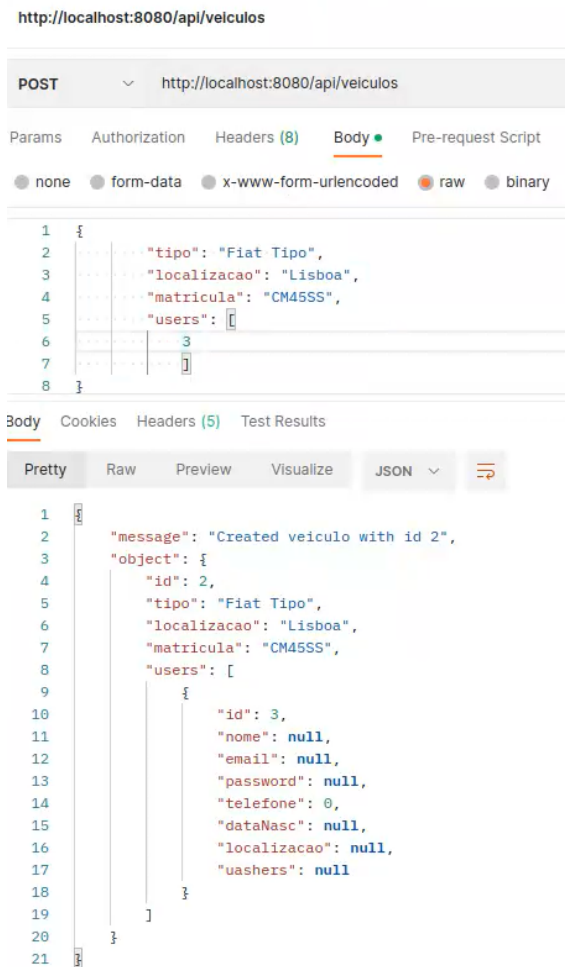
Sem parâmetros adicionais. Um JSONObject Veiculo deverá ser providenciado.

Resultado Esperado

```
{
  "message": "Created veiculo with id 2",
  "object": {
    "id": 2,
    "tipo": "Fiat Tipo",
    "localizacao": "Porto",
    "matricula": "CM45SS",
    "users": { ... }
  }
}
```

Sem códigos de erro programados

Exemplo



Atualizar Veiculo (put): /api/veiculos/{id:[0-9]+}

Id: O Id do Veiculo desejado.

Atualiza o veículo com os dados providenciados, que devem ser em formato JSONArray.

Resultado Esperado

```
{
  "message": "Updating veiculo with id 2",
  "object": {
    "id": 2,
    "tipo": "Fiat Panda",
    "localizacao": "Alges",
    "matricula": "CM45SS",
    "users": { ... }
  }
}
```

Sem códigos de erro programados

Exemplo

http://localhost:8080/api/veiculos/3

PUT http://localhost:8080/api/veiculos/3

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

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

```
1 {
2   "tipo": "Fiat Panda",
3   "localizacao": "Alges",
4   "matricula": "CM45SS",
5   "users": [
6     {
7       id: 3
8     }
9   ]
10 }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

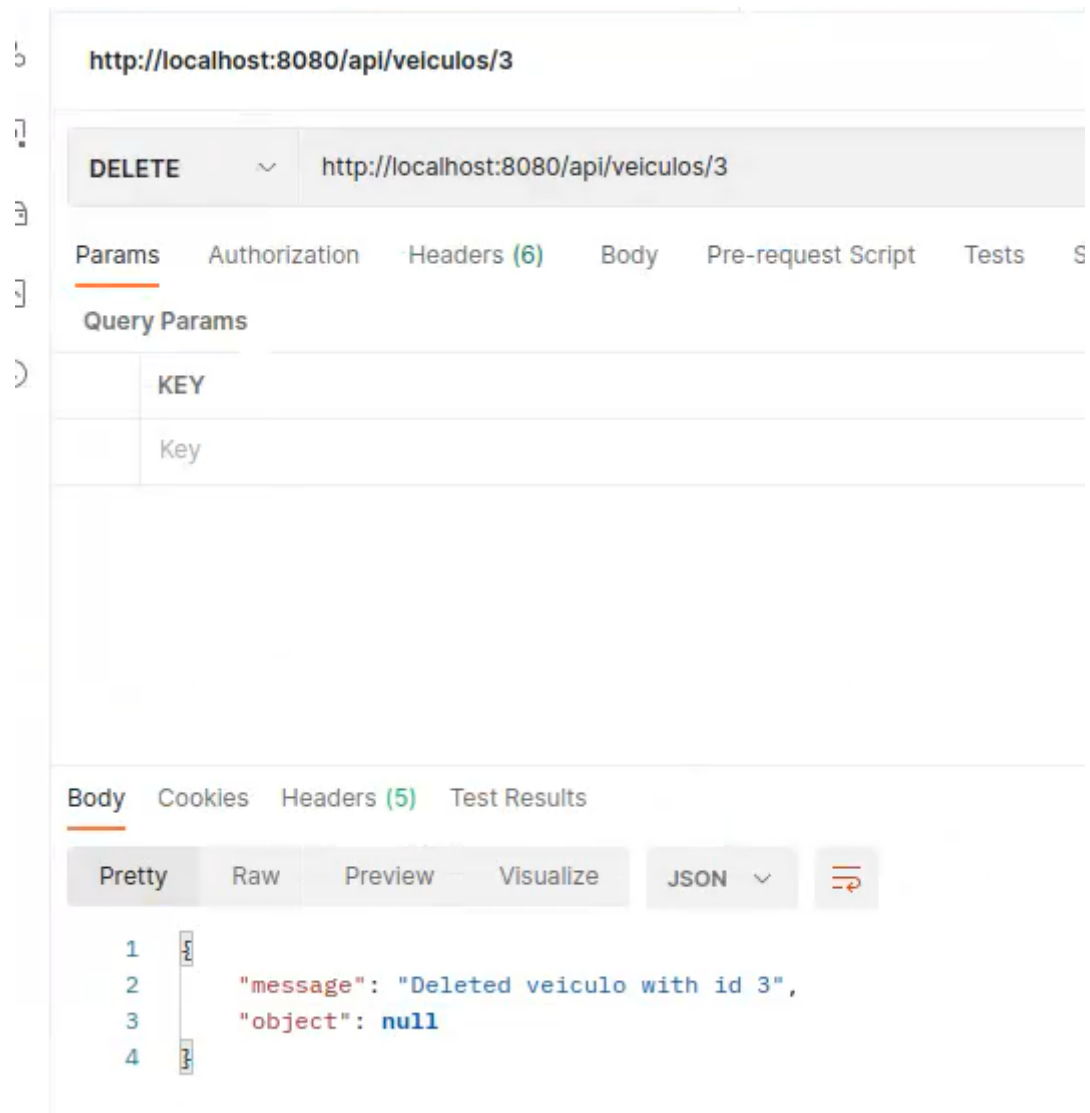
```
1 {
2   "message": "Updating veiculo with id 3",
3   "object": {
4     "id": 3,
5     "tipo": "Fiat Panda",
6     "localizacao": "Alges",
7     "matricula": "CM45SS",
8     "users": [
9       {
10        "id": 3,
11        "nome": null,
12        "email": null,
13        "password": null,
14        "telefone": 0,
15        "dataNasc": null,
16        "localizacao": null,
17        "uashers": null
18      }
19    ]
20  }
21 }
```

Apagar Veiculo (delete): /api/veiculos/{id:[0-9]+}

Id: O Id do Veiculo desejado.

Apaga o veículo desejado.

Resultado Esperado / Exemplo



Sem códigos de erro programados

Lavagens (/api/lavagens)

Exibir todas as Lavagens (get): /api/lavagens/

Sem parâmetros adicionais.

Devolve uma tabela com todas as lavagens e seus dados.

Resultado Esperado / Exemplo

GET http://localhost:8080/ POST http://localhost:8080/ PUT http://localhost:8080/ DEL http://localhost:8080/a + ...

http://localhost:8080/api/lavagens

GET http://localhost:8080/api/lavagens

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "status": "Confirmado",
4     "id": 10,
5     "valor": 30,
6     "tipoLavagem": "GaragemPrivada",
7     "localizacao": "Lisboa",
8     "horario": "2016-11-09",
9     "rating": 5,
10    "veiculoTipo": null,
11    "uasherId": 4,
12    "veiculoId": 2
13  },
14  {
15    "status": "Confirmado",
16    "id": 11,
17    "valor": 25,
18    "tipoLavagem": "Lavagem a Seco",
19    "localizacao": "Alges",
20    "horario": "2019-05-20",
21    "rating": 5,
22    "veiculoTipo": null,
23    "uasherId": 1,
24    "veiculoId": 2
25  }
26 ]
```

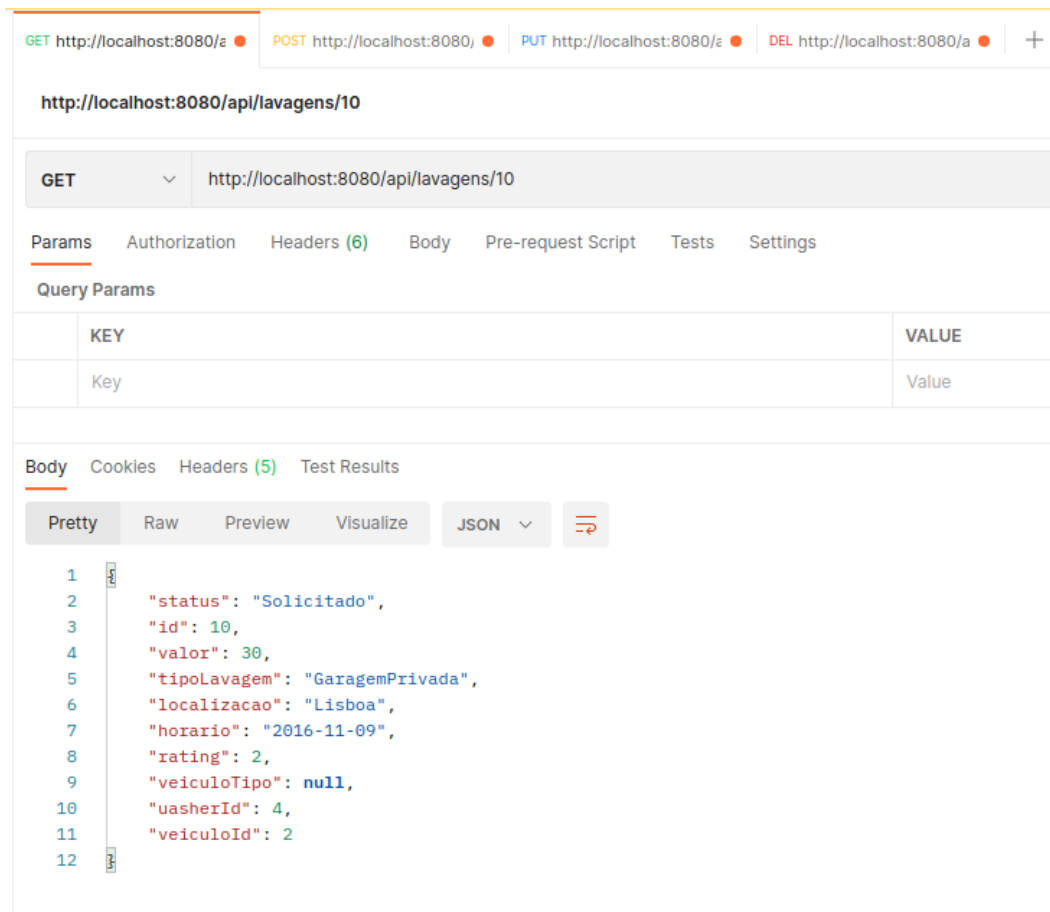
Sem códigos de erro programados

Exibir Lavagem por Id (get): /api/lavagens/{id:[0-9]+}

Id: O Id da Lavagem desejada.

Devolve os dados da lavagem desejada.

Resultado Esperado / Exemplo



Adicionar Lavagem (post): /api/lavagens/

Sem parâmetros adicionais. Um JSONObject Lavagem deverá ser providenciado.

Resultado Esperado

```
{  
  "valor": "25",  
  "tipoLavagem": "Lavagem a Seco",  
  "horario": "2019-05-20T11:44:44.797",  
  "localizacao": "Alges",  
  "status": "Confirmado",  
  "rating": 5,  
  "veiculo": 2,  
  "uasher": 1  
}
```

Sem códigos de erro programados

Exemplo

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/api/lavagens`. The request body is a JSON object with the following fields: `valor` (25), `tipoLavagem` (Lavagem a Seco), `horario` (2019-05-20T11:44:44.797), `localizacao` (Alges), `status` (Confirmado), `rating` (5), `veiculo` (2), and `uasher` (1). The response is a JSON object with a `message` field ("Created lavagem with id 11") and an `object` field containing the details of the created lavagem, including its `id` (11), `valor` (25), `tipoLavagem` (Lavagem a Seco), `localizacao` (Alges), `horario` (2019-05-20T11:44:44.797), `status` (Confirmado), `rating` (5), `veiculo` (2), and `uasher` (1).

```
1 POST http://localhost:8080/api/lavagens
2
3 {
4   "valor": "25",
5   "tipoLavagem": "Lavagem a Seco",
6   "horario": "2019-05-20T11:44:44.797",
7   "localizacao": "Alges",
8   "status": "Confirmado",
9   "rating": 5,
10  "veiculo": 2,
11  "uasher": 1
12 }
13
14
15
16
17
18
19
20
21
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Created lavagem with id 11",
3   "object": {
4     "id": 11,
5     "valor": 25,
6     "tipoLavagem": "Lavagem a Seco",
7     "localizacao": "Alges",
8     "horario": "2019-05-20T11:44:44.797",
9     "status": "Confirmado",
10    "rating": 5,
11    "veiculo": {
12      "id": 2,
13      "tipo": null,
14      "localizacao": null,
15      "matricula": null,
16      "users": null
17    },
18    "uasher": {
19      "id": 1,
20      "rating": 0,
21      "cartaConducao": null
22    }
23  }
24 }
```

Sem códigos de erro programados

Atualizar Lavagem (put): /api/lavagens/{id:[0-9]+}

Id: O Id da Lavagem desejada.

Atualiza a lavagem com os dados providenciados, que devem ser em formato JSONArray.

Resultado Esperado

```
{
  "status": "Solicitado", //Antes estava Confirmado
  "rating": 2 //Antes estava 5
}
```

Sem códigos de erro programados

Exemplo

GET http://localhost:8080/ POST http://localhost:8080/ PUT http://localhost:8080/ DEL http://localhost:8080/

http://localhost:8080/api/lavagens/11

PUT http://localhost:8080/api/lavagens/11

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

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

```
1  {
2    "status": "Solicitado",
3    "rating": 2
4  }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1  {
2    "message": "Updating lavagem with id 11",
3    "object": {
4      "id": 11,
5      "valor": 0,
6      "tipoLavagem": null,
7      "localizacao": null,
8      "horario": null,
9      "status": "Solicitado",
10     "rating": 2,
11     "veiculo": null,
12     "uasher": null
13   }
14 }
```

Apagar Lavagem (delete): /api/lavagens/{id:[0-9]+}

Id: O Id da Lavagem desejada.

Apaga a lavagem desejada.

Resultado Esperado/ Exemplo

GET http://localhost:8080/ε ●

POST http://localhost:8080/ ●

PUT http://localhost:8080/ε ●

DEL http://localhost:8080/a ●

http://localhost:8080/api/lavagens/11

DELETE ▼ http://localhost:8080/api/lavagens/11

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize

JSON ▼

```
1 {
2   "message": "Deleted lavagem with id 11",
3   "object": null
4 }
```

Sem códigos de erro programados