Chewie - Documentação

Sumário

Introdução

explicar o problema (a falta de uma ferramenta no mercado que facilite o agendamento de serviços de petshop e de clínicas veterinárias) e a solução (Chewie, um sistema que realiza agendamentos de serviços de petshop);

Esta API foi desenvolvida com base na arquitetura REST.

Ferramentas e Links

As ferramentas utilizadas para o desenvolvimento da API foram as seguintes:

- NodeJS (10.19.0), runtime de JavaScript;
- AdonisJS (4.0.12), um framework web de NodeJS;
- SQLite (3.31.1), biblioteca de banco de dados SQL utilizado em desenvolvimento;
- Insomnia, aplicação para requisições REST;
- Heroku para a hospedagem da API, com add-on JawsDB para a utilização de MySQL em produção;

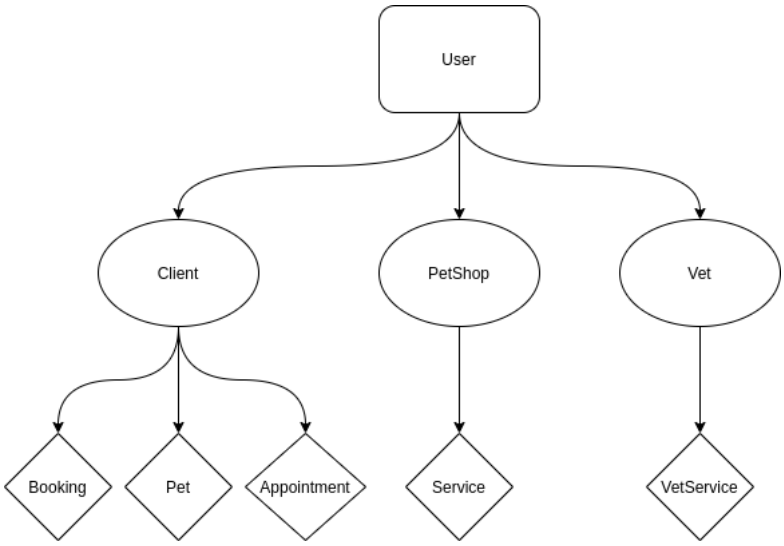
Links:

- Repositório GitHub
- Heroku

Autenticação

- Autorização nativa da API com uso de JWT

Endpoints



Recursos da API

User

- **Controller:** '/Controllers/Http/AuthController.js'
- **Model:** '/Models/User'
- **Métodos HTTP disponíveis:** Register e Authenticate

Utilizado para cadastro e login do usuário na plataforma. É necessária a execução do método para que sejam criados as sessões de **Client**, **PetShop** e **Vet**;

Register

- **POST**
- **Route:** `url/register`

Exemplo de Request

```
{
  "username":"andre", //tipo String
  "email":"andre@email.com", //tipo String
  "password":"123456", //tipo String
  "type": "client" //tipo String
}
```

Note que nessa requisição são encontrados parâmetros necessários tanto para o acesso (como e-mail e senha), como também necessários para validação de informações dentro da plataforma. Em especial o parâmetro type, que é utilizado para delimitar os níveis de acesso e permissão dos métodos da aplicação.

> Os tipos de usuário são: **'client'**,**'petshop'** e **'vet'**.

Authenticate

- **POST**
- **Route**: `url/authenticate`

Método utilizado para realizar o acesso do usuário (login) na aplicação.

Exemplo de Request:

```
{
  "email":"andre@email.com", //tipo String
  "password":"123456" //tipo String
}
```

Exemplo de Response:

```
{
  "user": {
    "id": 4,
    "username": "andre", //tipo String
    "email": "andre@email.com", //tipo String
    "password": "$2a$10$YFQTFQiv4CBRGq2UD4DszOcIl/U2oMZ..UV3C5FOH2MgpG0XrNYQC", //tipo String
    "type": "client", //tipo String
    "created_at": "2020-10-26 21:35:05",
    "updated_at": "2020-10-26 21:35:05"
  },
  "token": {
    "type": "bearer",
    "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1aWQiOjQsImlhdCI6MTYwMzc0ODExM30.LFSwmnhg-9vKnlxkD8K0iaVmlRjYMlj9ZoRHhAv2
    "refreshToken": null
  }
}
```

É importante mencionar que o token presente na *response* do presente método serve de autenticação para o usuário durante o acesso e manipulação de dados dentro da aplicação, sendo um parâmetro único para toda vez que é feita a autenticação do usuário.

Client

- **Controller:** '/Controllers/Http/ClientController.js'
- **Model:** '/Models/Client'
- **Pertence a:** User
- **Métodos HTTP disponíveis:** Store, Index, Show, Update e Destroy

O Client é basicamente o CRUD do "usuário final" da ferramenta, que realizará os agendamentos em petshops e clínicas veterinárias.

Store

- **POST**
- **Route:** `url/clients`

Exemplo de Request:

```
{
  "name": "André Carneiro", //tipo String
  "phone": "81 00000-0000" //tipo String
}
```

É importante lembrar também que a request utiliza o token JWT para a autenticação. Portanto, é necessário vinculá-lo à request;

Exemplo de Response:

```
{
  "user_id": 1,
  "name": "André Carneiro",
  "phone": "81 00000-0000",
  "created_at": "2020-10-26 21:07:24",
  "updated_at": "2020-10-26 21:07:24",
  "id": 1
}
```

Index

- **GET**
- **Route:** `url/clients`

Request:

Por ser uma requisição de método HTTP GET e servir apenas para listagem de clientes cadastrados, ela não leva nenhum parâmetro JSON no seu corpo de requisição. Portanto, leva-se apenas em consideração para executá-la o token de autenticação JWT e a rota já mencionada.

Exemplo de Response:

```
[
  {
    "id": 1,
    "name": "André",
    "phone": "81 00000-0000",
    "user_id": 1,
    "created_at": "2020-10-26 18:06:22",
    "updated_at": "2020-10-26 18:06:22",
    "user": {
      "id": 11,
      "username": "andre",
      "email": "andre@email.com",
      "password": "$2a$10$0o0aFNqA2cklsfNyOv4NhOlmlGutsp7HctYrhSSjDTAoBHNRuMQGu",
      "type": "client",
      "created_at": "2020-10-26 18:05:32",
      "updated_at": "2020-10-26 18:05:32"
    }
  },
  {
    "id": 11,
    "name": "Luis de Lima",
    "phone": "81 99999-9999",
    "user_id": 1,
    "created_at": "2020-10-26 18:31:40",
    "updated_at": "2020-10-26 18:31:40",
    "user": {
      "id": 1,
      "username": "andre",
      "email": "andre@email.com",
      "password": "$2a$10$0o0aFNqA2cklsfNyOv4NhOlmlGutsp7HctYrhSSjDTAoBHNRuMQGu",
      "type": "client",
      "created_at": "2020-10-26 18:05:32",
      "updated_at": "2020-10-26 18:05:32"
    }
  }
]
```

Note que além de mostrar os dados do cliente cadastrado, é mostrado também o usuário que cadastrou o mesmo.

Show

- **GET**

- **Route:** `url/clients/:id`

Request:

A request do método show de Client não recebe nenhum parâmetro a não ser o id do cliente, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id do cliente e estar autenticado.

Exemplo de Response:


```
{
  "client": {
    "id": 2,
    "name": "Luis de Lima",
    "phone": "81 00000-0000",
    "user_id": 2,
    "created_at": "2020-10-26 15:17:29",
    "updated_at": "2020-10-26 15:17:29"
  },
  "pets": [
    {
      "id": 2,
      "name": "Joao",
      "type": "Cachorro",
      "breed": "Dálmata",
      "age": 7,
      "client_id": 2,
      "created_at": "2020-10-26 15:18:18",
      "updated_at": "2020-10-26 15:18:18"
    }
  ],
  "bookings": [
    {
      "id": 1,
      "name": "Banho",
      "date_start": "2020-10-20 14:30:00",
      "date_end": "2020-10-20 15:30:00",
      "client_id": 2,
      "petshop_id": 1,
      "service_id": 1,
      "pet_id": 2,
      "created_at": "2020-10-30 18:26:41",
      "updated_at": "2020-10-30 18:26:41"
    },
],
  "appointments": [
    {
      "id": 3,
      "name": "Vacina",
      "date_start": "2020-10-29 14:30:00",
      "date_end": "2020-10-29 15:30:00",
      "client_id": 2,
      "vet_id": 1,
      "vet_service_id": 2,
      "pet_id": 2,
      "created_at": "2020-10-26 15:21:38",
      "updated_at": "2020-10-26 15:21:38"
    }
  ]
}
```


Na resposta do método, são mostrados também todos os pets, reservas em petshops e consultas veterinárias a ele vinculados.

Update

- **PUT**
- **Route:** `url/clients/:id`

Exemplo de Request:

Caso desejemos atualizar o número de telefone do cliente, podemos enviar no corpo da requisição apenas o key/value referente ao telefone:

```
{
  "phone":"81 99999-9999"
}
```

Exemplo de Response:

```
{
  "id": 1,
  "name": "Luís de Lima",
  "phone": "81 99999-9999",
  "user_id": 11,
  "created_at": "2020-10-16 14:21:31",
  "updated_at": "2020-10-16 21:39:04"
}
```

Destroy

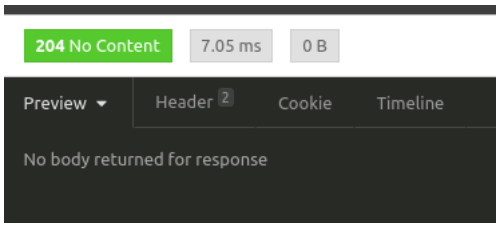
- **DELETE**
- **Route:** `url/clients/:id`

Request:

A request do método delete de Client não recebe nenhum parâmetro a não ser o id do cliente, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id do cliente e estar autenticado.

Response:

A resposta do método delete não retorna nada a não ser o código HTTP indicando o sucesso da requisição.



Pet

- **Controller:** '/Controllers/Http/PetController.js'
- **Model:** '/Models/Pet'
- **Pertence a:** Client
- **Métodos HTTP disponíveis:** Store, Index, Show, Update e Destroy

O CRUD de Pet é utilizado para que o cliente manipule os dados de seus pets e facilite a anexação dos mesmos ao realizar reservas em petshops ou clínicas veterinárias.

Store

- **POST**
- **Route:** `url/pets`

Exemplo de Request:

```
{
  "name": "Joao",
  "type": "Cachorro",
  "breed": "Dálmata",
  "age": 7
}
```

Exemplo de Response:

```
{
  "name": "Joao",
  "type": "Cachorro",
  "breed": "Dálmata",
  "age": 7,
  "client_id": 2,
  "created_at": "2020-10-26 15:18:18",
  "updated_at": "2020-10-26 15:18:18",
  "id": 2
}
```

Index

- **GET**
- **Route:** `url/pets`

Esse método lista todos os pets cadastrados.

Request:

Por ser uma requisição de método HTTP GET e servir apenas para listagem de pets cadastrados, ela não leva nenhum parâmetro JSON no seu corpo de requisição. Portanto, leva-se apenas em consideração para executá-la o token de autenticação JWT e a rota já mencionada.

Exemplo de Response:

```
{
    "id": 2,
    "name": "Joao",
    "type": "Cachorro",
    "breed": "Dálmata",
    "age": 7,
    "client_id": 2,
    "created_at": "2020-10-26 15:18:18",
    "updated_at": "2020-10-26 15:18:18",
    "client": {
      "id": 2,
      "name": "Luis de Lima",
      "phone": "81 00000-0000",
      "user_id": 1,
      "created_at": "2020-10-26 15:17:29",
      "updated_at": "2020-10-26 15:17:29"
    }
  }
```

Show

- **GET**
- **Route:** `url/pets/:id`

Request:

A request do método show de Pet não recebe nenhum parâmetro a não ser o id do pet, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id do pet e estar autenticado.

Exemplo de Response:

```
{
  "id": 1,
```

```
  "name": "Jake",
  "type": "Cachorro",
  "breed": "Dálmata",
  "age": 7,
  "client_id": 1,
  "created_at": "2020-10-23 14:15:13",
  "updated_at": "2020-10-23 14:15:13"
}
```


Update

- **PUT**
- **Route:** `url/pets/:id`

Exemplo de Request:

Caso desejarmos atualizar o nome do pet, por exemplo, precisamos especificar o id do pet e passarmos no corpo da requisição a chave referente ao nome do pet e o novo nome, conforme mostrado no exemplo a seguir:


```
{
  "name": "Baltazar",
}
```


Exemplo de Response:

Segue a resposta da atualização do pet feita com sucesso:


```
{
  "id": 1,
  "name": "Baltazar",
  "type": "Cachorro",
  "breed": "Dálmata",
  "age": 7,
  "client_id": 1,
  "created_at": "2020-10-23 14:15:13",
  "updated_at": "2020-10-23 14:25:18"
}
```


Destroy

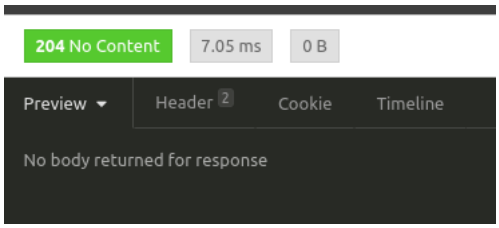
- **DELETE**
- **Route:** `url/pets/:id`

Request:

A request do método delete de Pet não recebe nenhum parâmetro a não ser o id do pet, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id do pet e estar autenticado.

Response:

A resposta do método delete não retorna nada a não ser o código HTTP indicando o sucesso da requisição.



Appointment

- **Controller:** '/Controllers/Http/AppointmentController.js'
- **Model:** '/Models/Appointment'
- **Pertence a:** Client
- **Métodos HTTP disponíveis:** Store, Index, Show, Update e Destroy

Store

- **POST**
- **Route:** `url/appointments`

Exemplo de Request:

```
{
  "name":"Vacina",
  "date_start":"2020-10-29 14:30:00",
  "date_end":"2020-10-29 15:30:00",
  "vet_username": "vet",
  "vet_service_id": 2,
  "pet_name": "Joao"
}
```

Exemplo de Response:

```
{
  "appointment": {
    "name": "Vacina",
    "date_start": "2020-10-29 14:30:00",
    "date_end": "2020-10-29 15:30:00",
    "client_id": 2,
    "vet_id": 1,
    "vet_service_id": 2,
    "pet_id": 2,
    "created_at": "2020-10-26 15:21:38",
    "updated_at": "2020-10-26 15:21:38",
    "id": 3
  },
  "pet": [
    {
      "id": 2,
      "name": "Joao",
      "type": "Cachorro",
      "breed": "Dálmata",
      "age": 7,
      "client_id": 2,
      "created_at": "2020-10-26 15:18:18",
      "updated_at": "2020-10-26 15:18:18"
    }
  ],
  "vet": {
    "id": 1,
    "name": "miau veterinaria",
    "address": "Rua das Ninfas, 24, Soledade, Recife - PE",
    "phone": "81 99999-9999",
    "username": "vet",
    "rating": 0,
    "user_id": 2,
    "created_at": "2020-10-26 14:53:51",
    "updated_at": "2020-10-26 14:53:51"
  },
  "vetService": [
    {
      "id": 2,
      "name": "Vacina",
      "value": 10,
      "duration": "0h10m",
      "employee": "Dr Alex",
      "description": "Vacina",
      "vet_id": 1,
      "created_at": "2020-10-26 15:15:55",
      "updated_at": "2020-10-26 15:15:55"
    }
  ]
}
```

Index

- **GET**
- **Route:** `url/appointments`

Request:

Por ser uma requisição de método HTTP GET e servir apenas para listagem de agendamentos feitos em veterinários, ela não leva nenhum parâmetro JSON no seu corpo de requisição. Portanto, leva-se apenas em consideração para executá-la o token de autenticação JWT e a rota já mencionada.

Exemplo de Response:


```
{
    "id": 1,
    "name": "Vacina",
    "date_start": "2020-10-29 14:30:00",
    "date_end": "2020-10-29 15:30:00",
    "client_id": 1,
    "vet_id": 1,
    "vet_service_id": 1,
    "pet_name": "Joao",
    "created_at": "2020-10-26 14:54:23",
    "updated_at": "2020-10-26 14:54:23",
    "client": {
      "id": 2,
      "name": "Luis de Lima",
      "phone": "81 00000-0000",
      "user_id": 1,
      "created_at": "2020-10-26 15:17:29",
      "updated_at": "2020-10-26 15:17:29"
    }
  },
  {
    "id": 2,
    "name": "Vacina",
    "date_start": "2020-10-29 14:30:00",
    "date_end": "2020-10-29 15:30:00",
    "client_id": 2,
    "vet_id": 1,
    "vet_service_id": 2,
    "pet_name": "Joao",
    "created_at": "2020-10-26 15:19:13",
    "updated_at": "2020-10-26 15:19:13",
    "client": {
      "id": 2,
      "name": "Luis de Lima",
      "phone": "81 00000-0000",
      "user_id": 1,
      "created_at": "2020-10-26 15:17:29",
      "updated_at": "2020-10-26 15:17:29"
    }
  }
```


Show

- **GET**
- **Route:** `url/appointments/:id`

Request:

A request do método show de Appointment não recebe nenhum parâmetro a não ser o id do agendamento, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id do agendamento e estar autenticado.

Response:


```
{
  "id": 1,
  "name": "Castração",
  "date_start": "2020-10-18 15:30:00",
  "date_end": "2020-10-18 16:30:00"
  "user_id": 1,
  "vet_id": 1,
  "vet_service_id": 1,
  "pet_name": "João",
  "created_at": "2020-10-16 03:45:05",
```

```
  "updated_at": "2020-10-16 03:45:05"
}
```

Update

- **PUT**
- **Route:** `url/appointments/:id`

Exemplo de Request:

Caso desejarmos alterar algum dado do agendamento, como por exemplo o horário do agendamento, basta definirmos o id do agendamento na URL e o horário atualizado pelo corpo da requisição, conforme demonstrado no exemplo:

```
{
  "date_start": "2020-10-18 16:30:00"
}
```

Exemplo de Response:

```
{
  "id": 1,
  "name": "Castração",
  "date_start": "2020-10-18 16:30:00",
  "date_end": "2020-10-18 17:30:00"
  "user_id": 1,
  "vet_id": 1,
  "vet_service_id": 1,
  "pet_name": "João",
  "created_at": "2020-10-16 03:45:05",
  "updated_at": "2020-10-16 03:55:23"
}
```

Destroy

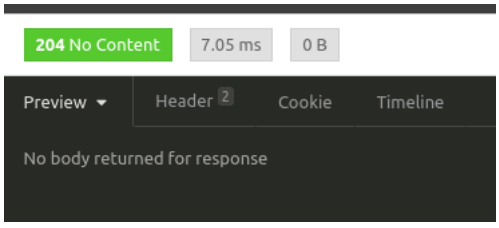
- **DELETE**
- **Route:** `url/appointments/:id`

Request:

A request do método delete de Appointment não recebe nenhum parâmetro a não ser o id do agendamento, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id do agendamento e estar autenticado.

Response:

A resposta do método delete não retorna nada a não ser o código HTTP indicando o sucesso da requisição.



Booking

- **Controller:** '/Controllers/Http/BookingController.js'
- **Model:** '/Models/Booking'
- **Pertence a:** Client
- **Métodos HTTP disponíveis:** Store, Index, Show, Update e Destroy

Store

- **POST**
- **Route:** `url/bookings`

Exemplo de Request:

```
{
  "name":"Banho", //tipo String
  "date_start":"2020-10-20 14:30:00", //tipo Datetime
  "date_end":"2020-10-20 15:30:00", //tipo Datetime
  "petshop_username": "andre123", //tipo String
  "service_id": 1, //tipo Integer
  "pet_name": "Joao"  //tipo String
}
```

Exemplo de Response:

```
"booking": {
    "name": "Banho",
    "date_start": "2020-10-20 14:30:00",
    "date_end": "2020-10-20 15:30:00",
    "client_id": 2,
    "petshop_id": 1,
    "service_id": 1,
    "pet_name": "Joao",
    "created_at": "2020-11-01 19:25:15",
    "updated_at": "2020-11-01 19:25:15",
    "id": 2
  },
  "pet": [
    {
      "id": 2,
      "name": "Joao",
      "type": "Cachorro",
      "breed": "Dálmata",
      "age": 7,
      "client_id": 2,
      "created_at": "2020-10-26 15:18:18",
      "updated_at": "2020-10-26 15:18:18"
    }
  ],
  "petshop": {
    "id": 1,
    "name": "Pet & RePet",
    "address": "Rua da Guia, 135, Recife, Recife - PE",
    "phone": "81 99009-0000",
    "username": "alex",
    "rating": 0,
    "user_id": 3,
    "created_at": "2020-10-30 18:10:25",
    "updated_at": "2020-10-30 18:10:25"
  },
  "service": [
    {
      "id": 1,
      "name": "Banho",
      "value": 20,
      "duration": "1h00m",
      "employee": "Alex",
      "description": "Banho completo no animal",
      "petshop_id": 1,
      "created_at": "2020-10-30 18:11:17",
      "updated_at": "2020-10-30 18:11:17"
    }
  ]
```

Index

- **GET**
- **Route:** `url/bookings`

Request:

Por ser uma requisição de método HTTP GET e servir apenas para listagem de reservas feitas em petshops, ela não leva nenhum parâmetro JSON no seu corpo de requisição. Portanto, leva-se apenas em consideração para executá-la o token de autenticação JWT e a rota já mencionada.

Exemplo de Response:

Segue um exemplo de resposta da API listando as reservas realizadas:


```
{
  {
    "id": 1,
    "name": "Banho",
    "date_start": "2020-10-20T14:30",
    "date_end": "2020-10-20T15:30",
    "client_id": 1,
    "petshop_id": 1,
    "service_id": null,
    "pet_id": 1,
    "created_at": "2020-10-26 00:18:55",
    "updated_at": "2020-10-26 00:18:55",
    "client": {
      "id": 1,
      "name": "Luis das Neves",
      "phone": "81 00000-0000",
      "user_id": 2,
      "created_at": "2020-10-26 00:17:32",
      "updated_at": "2020-10-26 00:17:32"
    }
  },
  {
    "id": 2,
    "name": "Banho",
    "date_start": "2020-10-20T14:30",
    "date_end": "2020-10-20T15:30",
    "client_id": 1,
    "petshop_id": 1,
    "service_id": null,
    "pet_id": 1,
    "created_at": "2020-10-26 00:20:28",
    "updated_at": "2020-10-26 00:20:28",
    "client": {
      "id": 1,
      "name": "Luis das Neves",
      "phone": "81 00000-0000",
      "user_id": 2,
      "created_at": "2020-10-26 00:17:32",
      "updated_at": "2020-10-26 00:17:32"
    }
  },
  {
    "id": 3,
    "name": "Banho",
    "date_start": "2020-10-20T14:30",
    "date_end": "2020-10-20T15:30",
    "client_id": 1,
    "petshop_id": 1,
    "service_id": null,
    "pet_id": 1,
    "created_at": "2020-10-26 00:21:06",
    "updated_at": "2020-10-26 00:21:06",
    "client": {
      "id": 1,
      "name": "Luis das Neves",
      "phone": "81 00000-0000",
      "user_id": 2,
      "created_at": "2020-10-26 00:17:32",
      "updated_at": "2020-10-26 00:17:32"
    }
  },
}
```


Show

- **GET**
- **Route:** `url/bookings/:id`

Request:

A request do método show de Booking não recebe nenhum parâmetro a não ser o id da reserva, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id da reserva e estar autenticado.

Exemplo de Response:

Update

- **PUT**
- **Route:** `url/bookings/:id`

Exemplo de Request:

Caso desejarmos alterar um dado da reserva, como por exemplo o horário da reserva, devemos especificar a reserva através do id passado pela URL, e também enviar pelo corpo da requisição o dado atualizado, conforme é mostrado no exemplo:

```
{
  "date_start": "2020-10-18 16:30:00"
}
```

Exemplo de Response:

Destroy

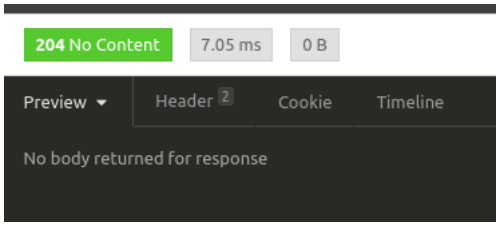
- **DELETE**
- **Route:** `url/bookings/:id`

Request:

A request do método delete de Booking não recebe nenhum parâmetro a não ser o id da reserva, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id da reserva e estar autenticado.

Response:

A resposta do método delete não retorna nada a não ser o código HTTP indicando o sucesso da requisição.



QueryService

- **Controller:** '/Controllers/Http/QueryServiceController.js'
- **Pertence a:** Client
- **Métodos HTTP disponíveis:** Index, Show

Esse simples método surgiu a partir da necessidade de que o cliente possa acessar todos os serviços pertencentes a um petshop específico, como também verificar quais são as reservas já existentes para um

Index

- **GET**
- **Route:** `url/:petshop_username/services`

Exemplo de Response:

```
{
  "id": 1,
  "name": "Banho",
  "value": 20,
  "duration": "1h00m",
  "employee": "Alex",
  "description": "Banho completo no animal",
  "petshop_id": 1,
  "created_at": "2020-10-30 18:11:17",
  "updated_at": "2020-10-30 18:11:17"
},
{
  "id": 2,
  "name": "Tosa",
  "value": 40,
  "duration": "1h00m",
  "employee": "Alex",
  "description": "Tosa completa no cachorro",
  "petshop_id": 1,
  "created_at": "2020-10-30 18:11:38",
  "updated_at": "2020-10-30 18:11:38"
}
```

Show

- **GET**
- **Route:** `url/:petshop_username/services/:id`

Exemplo de Response:

```
{
  "id": 1,
  "name": "Banho",
  "date_start": "2020-10-20 14:30:00",
  "date_end": "2020-10-20 15:30:00",
  "client_id": 2,
  "petshop_id": 1,
  "service_id": 1,
  "pet_id": 2,
  "created_at": "2020-10-30 18:26:41",
  "updated_at": "2020-10-30 18:26:41"
}
```

Petshop

- **Controller:** '/Controllers/Http/PetshopController.js'
- **Model:** '/Models/Petshop'
- **Pertence a:** User
- **Métodos HTTP disponíveis:** Store, Index, Show, Update e Destroy

Store

- **POST**
- **Route:** `url/petshops`

Exemplo de Request:

```
{
  "name": "Pet & RePet",
  "address": "Rua da Guia, 135, Recife, Recife - PE",
  "phone": "81 99009-0000"
}
```

Exemplo de Response:

```
{
  "name": "Pet & RePet",
  "address": "Rua da Guia, 135, Recife, Recife - PE",
  "phone": "81 99009-0000",
  "username": "alex",
  "user_id": 3,
  "created_at": "2020-10-30 18:10:25",
  "updated_at": "2020-10-30 18:10:25",
  "id": 2
}
```

Index

- **GET**
- **Route:** `url/petshops`

Esse método lista os petshops cadastrados no banco de dados.

Exemplo de Response:

```
[
  {
    "id": 1,
    "name": "Gato e Rato",
    "address": "Rua da Imperatriz, 123, Santo Antonio, Recife - PE",
    "phone": "81 99009-0000",
    "username": "andre123",
    "rating": 0,
    "user_id": 1,
    "created_at": "2020-10-26 00:15:33",
    "updated_at": "2020-10-26 00:15:33",
    "user": {
      "id": 1,
      "username": "andre123",
      "email": "andre123@email.com",
      "password": "$2a$10$mwWbFjuv1QNXI92ob8gE7eJ70B9bKJ20L5iC.Cqs0Ra41joUayW5S",
      "type": "petshop",
      "created_at": "2020-10-26 00:15:13",
      "updated_at": "2020-10-26 00:15:13"
    }
  },
  {
    "id": 2,
    "name": "Pet & RePet",
    "address": "Rua da Guia, 135, Recife, Recife - PE",
    "phone": "81 99009-0000",
    "username": "alex",
    "user_id": 3,
    "created_at": "2020-10-30 18:10:25",
    "updated_at": "2020-10-30 18:10:25",
    "user":{
      "id": 2,
      "username": "alex",
      "email": "alex@email.com",
      "password": "2a$10$mwWbFjuv1QNXI92ob8gE7eJ70B9bKJ20L5iC.Cqs0Ra41joUayW5",
      "type": "petshop",
      "created_at": "2020-10-30 18:06:52",
      "updated_at": "2020-10-30 18:06:52",
    }
  }
]
```

Show

- **GET**
- **Route:** `url/petshops/:id`

Esse método mostra um petshop especificado pelo id pela URL, listando os serviços disponibilizados pelo petshop.

Exemplo de Response:

```
{
    "id": 1,
    "name": "Gato e Rato",
```

```
    "address": "Rua da Imperatriz, 123, Santo Antonio, Recife - PE",
    "phone": "81 99009-0000",
    "username": "andre123",
    "rating": 0,
    "user_id": 1,
    "created_at": "2020-10-26 00:15:33",
    "updated_at": "2020-10-26 00:15:33",
    "services": [
      {
        "id": 1,
        "name": "Banho",
        "value": 20,
        "duration": "0h30m",
        "employee": "Alex",
        "description": "Banho completo no animal",
        "petshop_id": 1,
        "created_at": "2020-10-26 00:15:43",
        "updated_at": "2020-10-26 00:15:43"
      }
    ]
  }
```


Update

- **PUT**
- **Route:** `url/petshops/:id`

Exemplo de Request:

Caso desejamos alterar algum dado do petshop cadastrado, por exemplo o nome, devemos especificar o petshop pelo id na URL e o dado atualizado no corpo da requisição:


```
{
  "name": "Fala Bicho"
}
```


Exemplo de Response:

A resposta da API é o objeto constando o parâmetro atualizado:


```
{
  "id": 1,
  "name": "Fala Bicho",
  "address": "Rua da Imperatriz, 123, Santo Antonio, Recife - PE",
  "phone": "81 99009-0000",
  "username": "andre123",
  "rating": 0,
  "user_id": 1,
  "created_at": "2020-10-25 23:49:15",
  "updated_at": "2020-10-25 23:52:25"
}
```


Destroy

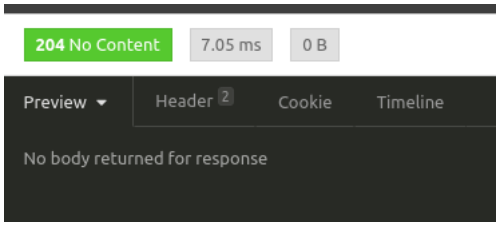
- **DELETE**
- **Route:** `url/petshops/:id`

Request:

A request do método delete de Petshop não recebe nenhum parâmetro a não ser o id do petshop, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id do pershop e estar autenticado.

Response:

A resposta do método delete não retorna nada a não ser o código HTTP indicando o sucesso da requisição.



Service

- **Controller:** '/Controllers/Http/ServiceController.js'
- **Model:** '/Models/Service'
- **Pertence a:** Petshop
- **Métodos HTTP disponíveis:** Store, Index, Show, Update e Destroy

Store

- **POST**
- **Route:** `url/petshop/services`

Exemplo de Request:

```
{
  "name":"Tosa",
  "value": 40.00,
  "duration":"1h00m",
  "employee":"Alex",
  "description":"Tosa completa no cachorro"
}
```

Exemplo de Response:

```
{
  "name": "Tosa",
  "value": 40,
  "duration": "1h00m",
  "employee": "Alex",
  "description": "Tosa completa no cachorro",
  "petshop_id": 1,
  "created_at": "2020-10-30 18:11:38",
  "updated_at": "2020-10-30 18:11:38",
  "id": 2
}
```

Index

- **GET**
- **Route:** `url/petshop/services`

Exemplo de Response:

```

```

Show

- **GET**
- **Route:** `url/petshop/services/:id`

Update

- **PUT**
- **Route:** `url/petshop/services/:id`

Exemplo de Request:

```
{
  "value": 20.5,
  "employee": "Dan"
}
```

Exemplo de Response:

```
{
  "id": 3,
  "name": "Banho",
  "value": 20.5,
  "duration": "0h30m",
  "employee": "Dan",
  "description": "Serviço feito com higiene e qualidade, sem maltratar o pet",
  "petshop_id": 2,
  "created_at": "2020-10-16 18:05:54",
  "updated_at": "2020-10-16 21:45:16"
}
```

Destroy

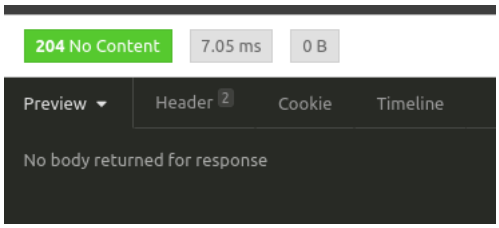
- **DELETE**
- **Route:** `url/petshop/services/:id`

Request:

A request do método delete do Service não recebe nenhum parâmetro a não ser o id do serviço, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id do serviço e estar autenticado.

Response:

A resposta do método delete não retorna nada a não ser o código HTTP indicando o sucesso da requisição.



Vet

- **Controller:** '/Controllers/Http/VetController.js'
- **Model:** '/Models/Vet'
- **Pertence a:** Vet
- **Métodos HTTP disponíveis:** Store, Index, Show, Update e Destroy

Store

- **POST**
- **Route:** `url/vets`

Index

- **GET**
- **Route:** `url/vets`

Exemplo de Response:

```
{
  "id": 1,
  "name": "miau veterinaria",
  "address": "Rua das Ninfas, 24, Soledade, Recife - PE",
  "phone": "81 99999-9999",
  "rating": 0,
  "user_id": 1,
  "created_at": "2020-10-26 14:45:14",
  "updated_at": "2020-10-26 14:45:14",
  "user": {
    "id": 1,
    "username": "vet",
    "email": "vet@email.com",
    "password": "$2a$10$xjp/Q.GjNrUDbv0wL3dgbeqgCMs2NFf0IOsSiPtp3HLPx5MGD6tlS",
    "type": "vet",
    "created_at": "2020-10-26 14:44:31",
    "updated_at": "2020-10-26 14:44:31"
  }
}
```

Show

- **GET**
- **Route:** `url/vets/:id`

Exemplo de Response:

```
{
  "id": 1,
  "name": "miau veterinaria",
  "address": "Rua das Ninfas, 24, Soledade, Recife - PE",
  "phone": "81 99999-9999",
  "rating": 0,
  "user_id": 4,
  "created_at": "2020-10-26 13:57:24",
  "updated_at": "2020-10-26 13:57:24"
}
```

Update

- **PUT**
- **Route:** `url/vets/:id`

Exemplo de Request:

```
{
  "name": "Vet Care"
}
```

Exemplo de Response:

```
{
  "id": 1,
  "name": "Pet Care",
  "address": "Rua das Ninfas, 24, Soledade, Recife - PE",
  "phone": "81 99999-9999",
  "rating": 0,
  "user_id": 4,
  "created_at": "2020-10-26 14:45:14",
  "updated_at": "2020-10-28 09:24:36"
}
```

Destroy

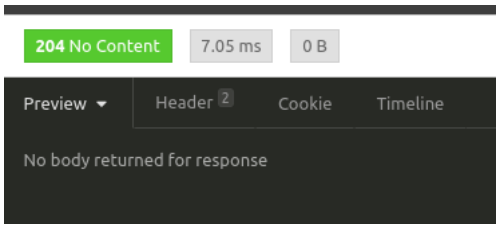
- **DELETE**
- **Route:** `url/vets/:id`

Request:

A request do método delete de Booking não recebe nenhum parâmetro a não ser o id da reserva, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id da reserva e estar autenticado.

Response:

A resposta do método delete não retorna nada a não ser o código HTTP indicando o sucesso da requisição.



VetService

- **Controller:** '/Controllers/Http/VetServiceController.js'
- **Model:** '/Models/VetService'
- **Pertence a:** Vet
- **Métodos HTTP disponíveis:** Store, Index, Show, Update e Destroy

Store

- **POST**
- **Route:** `url/vet/services`

Exemplo de Request:

```
{
  "name":"Vacina",
  "value": 10.00,
  "duration":"0h10m",
  "employee":"Dr Alex",
  "description":"Vacina"
}
```

Exemplo de Response:

```
{
  "name": "Vacina",
  "value": 10,
  "duration": "0h10m",
  "employee": "Dr Alex",
  "description": "Vacina",
  "vet_id": 1,
  "created_at": "2020-10-26 15:15:55",
  "updated_at": "2020-10-26 15:15:55",
  "id": 2
}
```

Index

- **GET**
- **Route:** `url/vet/services`

Exemplo de Response:

```
[
  {
    "id": 2,
    "name": "Vacina",
    "value": 10,
    "duration": "0h10m",
    "employee": "Dr Alex",
    "description": "Vacina",
    "vet_id": 1,
    "created_at": "2020-10-26 15:15:55",
    "updated_at": "2020-10-26 15:15:55",
    "vet": {
      "id": 1,
      "name": "miau veterinaria",
      "address": "Rua das Ninfas, 24, Soledade, Recife - PE",
      "phone": "81 99999-9999",
      "username": "vet",
      "rating": 0,
      "user_id": 4,
      "created_at": "2020-10-26 14:53:51",
      "updated_at": "2020-10-26 14:53:51"
    }
  }
]
```

Show

- **GET**
- **Route:** `url/vet/services/:id`

Exemplo de Response:

```
{
  "id": 2,
  "name": "Vacina",
  "value": 10,
  "duration": "0h10m",
  "employee": "Dr Alex",
  "description": "Vacina",
  "vet_id": 2,
  "created_at": "2020-10-16 21:07:56",
  "updated_at": "2020-10-16 21:07:56"
}
```

Update

- **PUT**
- **Route:** `url/vet/services/:id`

Exemplo de Request:

```
{
  "value": 20.5,
  "employee": "Dan"
}
```

Exemplo de Response:

```
{
  "id": 2,
  "name": "Vacina",
  "value": 20.5,
  "duration": "0h10m",
  "employee": "Dan",
  "description": "Vacina",
  "vet_id": 2,
  "created_at": "2020-10-16 21:07:56",
  "updated_at": "2020-10-16 21:46:35"
}
```

Destroy

- **DELETE**
- **Route:** `url/vet/services/:id`

Request:

A request do método delete de VetService não recebe nenhum parâmetro a não ser o id do serviço, enviado pela url. Portanto, para realizar essa request, basta apenas especificar o id do serviço e estar autenticado.

Response:

A resposta do método delete não retorna nada a não ser o código HTTP indicando o sucesso da requisição.