Sistema de Cadastro de Animais com Autenticação de Usuário

"O teste consiste no seguinte:

O Projeto deve consistir em um cadastro de animais com a raça respectiva e idade, deve conter autenticação pra entrar no sistema, será avaliado a lógica e o desempenho da aplicação, visual não será um critério, mas pode ajudar bastante.
O projeto deve ser postado no GitHub e enviado o link pra Bruna, será bacana usar um MySQL separado e enviar o dump do banco junto no projeto!
1- Deve ter autenticação de usuário
2- Criação do animal
3- Raça do animal
4- Idade do animal
5- Apenas criar, editar, não colocar excluir"

A linguagem e framework escolhidos para desenvolver o sistema foi Ruby on Rails.

```
pedro@minare:~$ sudo ruby -v
ruby 2.7.0p0 (2019-12-25 revision 647ee6f091) [x86_64-linux-gnu]
pedro@minare:~$ sudo rails -v
Rails 5.2.0
pedro@minare:~$
```

Ruby na versão 2.7.0p0, Rails na versão 5.2.0 instalados num ambiente Linux Ubuntu 20.04.4 LTS.

```
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.4 LTS
Release:        20.04
```

A versão do MySQL instalada é 8.0.30-0ubuntu0.20.04.2.

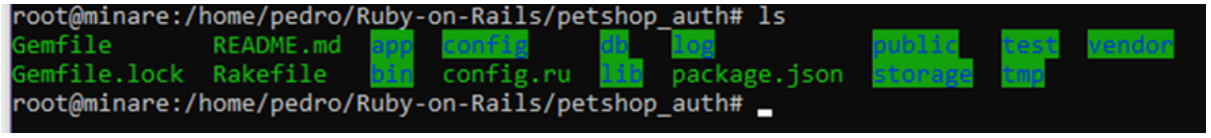
```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.30-0ubuntu0.20.04.2 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Reading history-file /root/.mysql_history
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

Uma vez que o ambiente já possui o Rails, Ruby e MySQL instalado, o próximo passo é construir a aplicação através do Rails. Em ambientes de produção não é uma boa prática fazer uso direto do usuário “root” do Linux Ubuntu (normalmente cria-se um usuário e atribui privilégios utilizando “sudo” antes de executar comandos). Apenas fiz uso dos comandos pelo root por se tratar de um ambiente provisório de testes.

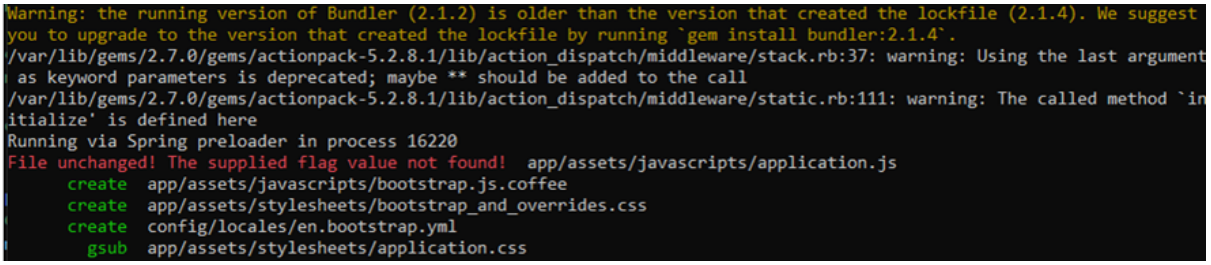
O Rails consegue facilmente iniciar uma aplicação (dei o nome de petshop_auth) através do comando:

```
rails _5.2.0_ new petshop_auth -d=mysql
```

Assim é possível selecionar a versão desejada do Rails e informar que o banco de dados a ser usado será o MySQL.



Até o momento não foi necessário adicionar gems no Gemfile. No entanto, tomei a liberdade de inserir a gem do bootstrap, responsável pelo layout responsivo e mais agradável, sendo necessário o comando `bundle-install` e `rails generate bootstrap:install static`.



Quando uma aplicação Rails é criada e o banco de dados utilizado é o MySQL há a necessidade de se criar as tabelas do banco de dados de forma antecipada. Para isso utilizei o comando `rake db:create` antes mesmo de iniciar o servidor.

De acordo com o que foi proposto no teste, o cadastro deve conter a raça do animal, o nome e a idade. Para isso, foi necessário que o modelo de cadastro fosse gerado através do comando:

```
rails generate scaffold animal raca:string nome:string idade:integer
```

O modelo foi criado com o nome `animal`. Assim que um modelo é criado, é necessário que o db seja migrado através do comando `rails db:migrate`.

É possível visualizar as tabelas criadas por meio do `rails dbconsole`. Utilizando o MySQL, basta utilizar o comando `show databases;` para exibir os bancos de dados existentes, assim como `show tables;` e `describe tablename;` para mais informações.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| petshop_development |
| petshop_test       |
| sys                |
+--------------------+
6 rows in set (0.00 sec)
```

```
mysql> show tables;
+-------------------------------+
| Tables_in_petshop_development |
+-------------------------------+
| animals                       |
| ar_internal_metadata          |
| schema_migrations             |
| users                         |
+-------------------------------+
4 rows in set (0.00 sec)
```

```
mysql> describe animals;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| id         | bigint       | NO   | PRI | NULL    | auto_increment |
| raca       | varchar(255) | YES  |     | NULL    |                |
| nome       | varchar(255) | YES  |     | NULL    |                |
| idade      | int          | YES  |     | NULL    |                |
| created_at | datetime     | NO   |     | NULL    |                |
| updated_at | datetime     | NO   |     | NULL    |                |
| user_id    | int          | YES  |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
7 rows in set (0.00 sec)
```

```
mysql> describe users;
+------------------------+--------------+------+-----+---------+----------------+
| Field                  | Type         | Null | Key | Default | Extra          |
+------------------------+--------------+------+-----+---------+----------------+
| id                     | bigint       | NO   | PRI | NULL    | auto_increment |
| email                  | varchar(255) | NO   | UNI |         |                |
| encrypted_password     | varchar(255) | NO   |     |         |                |
| reset_password_token   | varchar(255) | YES  | UNI | NULL    |                |
| reset_password_sent_at | datetime     | YES  |     | NULL    |                |
| remember_created_at    | datetime     | YES  |     | NULL    |                |
| created_at             | datetime     | NO   |     | NULL    |                |
| updated_at             | datetime     | NO   |     | NULL    |                |
+------------------------+--------------+------+-----+---------+----------------+
8 rows in set (0.00 sec)
```

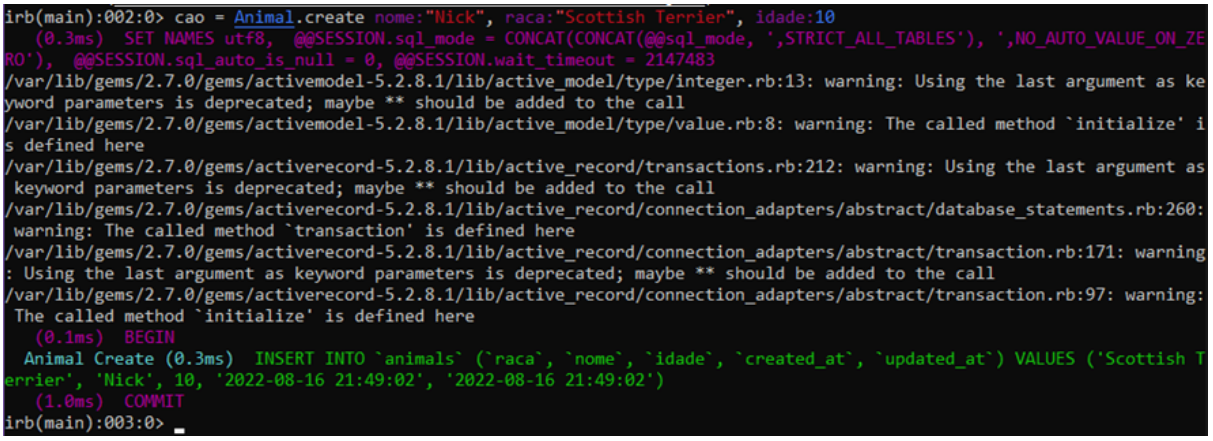
O teste inclui autenticação de usuário. A comunidade do Rails desenvolveu a gem **Devise**, responsável por gerar telas de **login, cadastro, logout, criptografia de passwords** e demais features responsáveis pela autenticação segura de usuários. Assim, como parte do que ensina a prática do **DRY (Don't Repeat Yourself**), decidi fazer uso desse recurso no projeto:

Adicionei a gem Devise no Gemfile:

Executei o `bundle install` e `rails g devise:install`

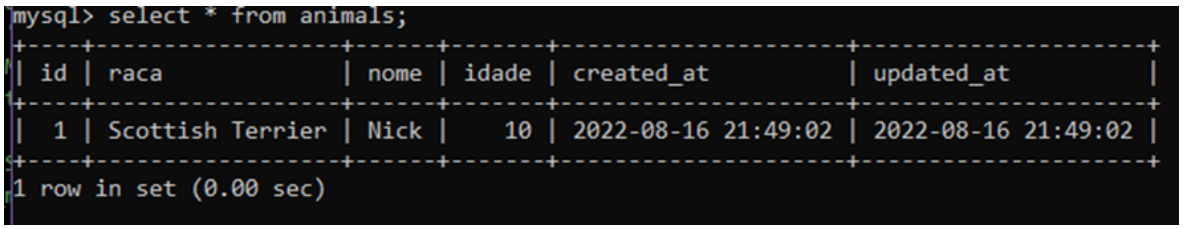
O Rails, como boa prática, considera criar campos automaticamente como `"id"`, `"created_at"` e `"updated_at"` como forma de registrar informações importantes que podem não ter sido declaradas durante a modelagem da tabela do banco de dados.

Apenas com a finalidade de inserir o primeiro cadastro teste, inseri dados na tabela criada utilizando o rails console:



Utilizei a variável `cao` para cadastrar o primeiro animal no banco de dados.

```
cao = Animal.create nome:"Nick", raca:"Scottish Terrier", idade:10
```



Uma vez que o banco de dados do projeto já é capaz de ser alimentado, o próximo passo é melhorar a experiência visual com o bootstrap, configurar o Devise e testar as funcionalidades.

A tabela dos usuários é `users` e a tabela de cadastro de animais é `animals`. As tabelas ainda não possuem associações (relacionamento). Assim, criei uma **migration** para incluir uma coluna `integer` chamada `user_id` na tabela animals. Em seguida, executei `rake db:migrate` e adicionei o comando `has_many :animals` em app/models/user.rb e `belongs_to :user` em `app/models/animal.rb`. Assim, uma instância de **user** terá automaticamente um método que irá referenciar todos os animais com um `user_id`.

Agora é necessário que seja exigida uma autenticação de usuário para executar as ações propostas no teste: **Criar e Editar.** Adicionei o comando `before_action :authenticate_user!` em `app/controllers/animals_controller.rb` e também foi necessário inserir em `def create` que todos os usuários cadastrados possam executar

ações em qualquer registro de animais, relacionando o usuário com cada animal cadastrado pelo mesmo, inserindo `@animal.user_id = current_user.id` . Considero uma boa prática para um sistema grande, onde vários usuários podem ser cadastrados, assim como cadastrar vários animais.

Agora é só rodar a aplicação com o comando `rails s` no ambiente Linux Ubuntu e alimentar o banco de dados acessando `http://localhost:3000`. Tomei a liberdade de ordenar os cadastros por nome do animal.

Testes realizados no projeto:

- É possível cadastrar um novo usuário por e-mail (Devise).
- A senha deve conter pelo menos 6 caracteres (Devise).
- É possível fazer login/logout (Devise).
- É possível cadastrar mais de um animal de mesmo nome, raça e idade, haja visto que podem haver coincidências e que um registro único de animal (como um Pedigree, por exemplo), não faz parte do teste proposto.
- O campo `idade` do animal pode conter valores entre 1 e 30 e apenas valores inteiros.
- Não é possível cadastrar um animal com qualquer dos campos em branco. Todos os campos possuem validação e devem ser preenchidos com letras ou números.
- Além da página inicial de login, não é possível acessar demais páginas do projeto sem estar logado.

O Rails é um excelente framework para se trabalhar, uma vez que a comunidade desenvolvedora já criou diversos recursos disponíveis para atender às mais diversas necessidades de mercado.

O banco de dados foi exportado por meio do comando:

```
mysqldump -u root petshop_development > petshop.sql
```

Link do projeto no GitHub:

https://github.com/pedrominare/Ruby-on-Rails/tree/main/PetShop

Foi um prazer realizar o teste proposto!

Pedro Minaré