Eder Rosati Ribeiro — 8122585,
Leonardo de Almeida Lima Zanguetin — 8531866,
Victor Luiz da Silva Mariano Pereira — 8602444

Projeto 3
CouchDB PetShop

Brasil

2018

Eder Rosati Ribeiro — 8122585,
Leonardo de Almeida Lima Zanguetin — 8531866,
Victor Luiz da Silva Mariano Pereira — 8602444

Projeto 3
CouchDB PetShop

Universidade de São Paulo – USP
Instituto de Ciências Matemáticas e de Computação – ICMC
Introdução ao Desenvolvimento Web – SCC0219

Brasil
2018

Introdução

Nesta terceira fase, a final, da aplicação PetShop, nos foi solicitado que concluíssemos o desenvolvimento de uma página *web* utilizando as linguagens HTML5, CSS3, as quais havíamos utilizado para desenvolver a primeira parte do projeto, JavaScript e o uso de *frameworks*, vistos na segunda parte, e agora a utilização de um banco de dados.

O objetivo desta parte é o refinamento de uma página no estilo *Single-Page Application* de um PetShop. Por termos feito já o *mockup* na primeira parte do trabalho e a utilização de JavaScrip com *frameworks* da segunda parte, acreditamos que será mais simples de fazer essa terceira, e última, parte.

Na segunda parte do trabalho, utilizamos uma API para simular um banco de dados, pois pensamos que seria mais simples a migração para um banco que utiliza a arquitetura RESTful. Escolhemos para essa parte o *CouchDB*, um banco de dados noSQL que utiliza como linguagem JavaScrip e protocolo de comunicação o HTTP.

1 Funcionamento

Como foi dito anteriormente, todo o *site* foi desenvolvido utilizando o *React*, uma biblioteca responsável apenas pelo desenvolvimento do *front-end* em *Single Page Application* (SPA). Tudo que julgamos conveniente à reutilização foi componentizado para reaproveitamento, tanto nos itens que precisavam ser reutilizáveis quanto nas partes que tinham papéis diferentes dentro do *site*.

Essa biblioteca JavaScript roda através do Node, portanto é necessário a instalação do Node e do NPM (do inglês, *Node Package Manager*) para iniciar um projeto. Com o NPM é possível baixar e instalar o *Create React App*, um pacote responsável por configurar várias tecnologias e trazer um projeto pronto para o desenvolvimento com todas as ferramentas iniciais mais utilizadas no *React*. Uma dessas tecnologias é o JSX, capaz de transformar os códigos JavaScript em códigos parecidos com HTML5.

Pode haver estranhamento com o código desenvolvido por aparentar utilizar HTML misturado com JavaScript, porém, é tudo JavaScript e o Babel nos auxilia nessa etapa. Todo o código pode ser realizado em ECMAScrip 6 pois é transpilado e convertido em ECMAScript 5 para funcionar nos navegadores atuais que ainda não o suportam.

Juntamente com a parte estrutural do *site*, também adicionamos uma base de dados, a qual alimenta a aplicação com o os dados previamente gerados pelo grupo. Em outras palavras, o banco de dados disponibiliza os dados por meio de uma API e o projeto mostra essas informações na tela para o usuário.

2 Estrutura do *site*

Como o nosso *site* é baseado praticamente em componentização, neste capítulo passaremos uma noção de como o estruturamos.

2.1 APP

No começo de tudo, temos a nossa aplicação, mais conhecida como APP, componente a qual consiste o nosso *site* inteiro e que será responsável por formar nosso único arquivo `.html`, chamado `index.html`. A APP está separada em 4 partes, as quais são o **Topo**, ***Navbar***, **Conteúdo** e **Rodapé**;

2.1.1 Topo

Neste componente nós temos três itens basicamente, o logo da aplicação, o qual também é uma ligação para a página inicial, uma barra de busca no *site* e uma área de acesso ao usuário. O **Topo** é o único componente, dos quatro principais, que tem uma parte dinâmica, além do componente **Conteúdo**, pois ela contém a parte do *login*, onde o usuário pode entrar com o seu *e-mail* e senha ou, se já tiver entrado, acessar sua conta.

2.1.1.1 Barra de busca

Área de pesquisa que serve para buscar algum produto que tenha na loja. A parte da busca ainda não está funcionando completamente, pois busca só palavras completas na parte de produtos com *case sensitive*, ou seja, se tiver um produto com o nome "Produto 1" e procurarmos com as palavras "prod", "produto 1", "Prod" ou "Produto" não haverá retorno, só teremos sucesso se procurarmos utilizando a palavra "Produto 1".

2.1.1.2 *Login*

Este componente, como foi falado anteriormente, serve para o usuário entrar na sua conta ou se cadastrar, caso não tiver uma conta cadastrada. Caso "*logado*", o usuário pode acessar a área de usuário, sair da sessão e acessar o carrinho de compras. Um ponto importante de se falar é que na parte do *login* utilizamos um padrão de projeto (*pattern design*) chamado *flux/redux*, responsável por criar uma store com as variáveis que precisamos acessar por outras componentes.

2.1.2 *Navbar*

Um menu que serve para acessar de forma mais intuitiva algumas partes do *site* como **Início**, o qual você consegue acessar de qualquer lugar do *site* a página inicial, **Serviços**, onde você consegue de qualquer lugar da aplicação ir para a página de serviços que estão disponíveis, e alguns tipos de animais, como **Cachorros**, **Gatos**, entre outros, que são um atalho para os produtos da respectiva espécie.

2.1.3 Conteúdo

Além da parte do **Topo**, a qual tem a área de *login*, essa é a única parte do nosso *site* que se muda, nela terá todas as informações não estáticas do site. Um exemplo é na tela incial, no componente **Conteúdo** temos uma área que fica um *slide* de imagens mostrando os destaques que o dono da aplicação pode colocar e, logo abaixo, temos listado alguns produtos de cada espécie, esses produtos, na nossa aplicação está pegando os quatro primeiros de cada espécie que estão cadastrados. Nada impede que coloquemos os produtos de cada espécie que estão em promoção.

Nesta parte do site, será mostrada todas as informações de usuário, de serviços, listagem de produtos, resultados de pesquisa, cadastros em geral, entre outros, ou seja, este componente é responsável por chamar outros componentes.

2.1.4 Rodapé

No rodapé, há o nome completo de todos os integrantes do grupo, onde cada nome é um *link* que redireciona o usuário para o a conta do GitHub de cada um.

3 Instruções

Neste capítulo vamos mostrar como utilizar a nossa aplicação, os comandos estão listados em ordem, portanto a execução deles resulta no funcionamento do projeto. Na sequência, teremos a instalação das ferramentas necessárias no Linux e no Windows e posteriormente como será o procedimento para rodar a aplicação.

3.1 Linux

3.1.1 Instalação do Node

```
1 sudo apt-get install nodejs #Distros baseadas em Debian ou
2 sudo pacman -S nodejs       #Distros baseadas em Arch Linux
```

3.1.2 Instalação do NPM

```
1 sudo apt-get install npm #Distros baseadas em Debian ou
2 sudo pacman -S npm       #Distros baseadas em Arch Linux
```

3.1.3 Instalação do *Create React App*

```
1 sudo npm install -g create-react-app
```

3.2 Windows

3.2.1 Instalação do Node

Pode ser baixado através do *site* oficial: https://nodejs.org/en/download/

3.2.2 Instalação do NPM

Como o NPM vem do Node e é instalado por padrão, não é preciso instalá-lo.

3.2.3 Instalação do *Create React App*

```
1 npm install -g create-react-app
```

3.3 Instalação do *CouchDB*

Cada distribuição linux tem a instalação do *CouchDB* de uma forma, podendo também baixar direto do site, assim como feito no Windows. No GNU/Linux é importante verificar se o serviço/*daemon* do banco de dados está funcionando, não colocaremos código para exemplificar, pois pode mudar de distribuição para distribuição.

Para verificar se o banco de dados está funcionando, basta entrar no endereço http://localhost:5984/.

Para o funcionamento da aplicação, precisamos ativar as permossões de acesso de domínios, para acessar a API do banco basta acessar http://localhost:5984/_utils, com isso basta ir nas configurações do *CouchDB*, na opção CORS e clicar em *Enable* CORS e selecionar a opção *All domains*.

3.3.1 Alimentando o banco de dados

Para colocar os conteúdos que temos no banco de dados, temos que utilizar os arquivos contidos na pasta `database/` da raiz do projeto, então vamos para a pasta:

```
1 cd database
```

Antes de adicionar o conteúdo dos `.json`, precisamos criar o banco de dados para depois inserir as tuplas, então precisamos rodar os seguintes comandos:

```
1 curl -X PUT http://localhost:5984/petshop
2 curl -X PUT http://localhost:5984/pets
3 curl -X PUT http://localhost:5984/usuarios
4 curl -X PUT http://localhost:5984/servicos
```

E, agora para inserir os dados no banco de dados:

```
1 curl -d @petshop.json -H "Content-type: application/json" -X POST
  ↪ http://localhost:5984/petshop/_bulk_docs
2 curl -d @pets.json -H "Content-type: application/json" -X POST
  ↪ http://localhost:5984/pets/_bulk_docs
3 curl -d @usuarios.json -H "Content-type: application/json" -X POST
  ↪ http://localhost:5984/usuarios/_bulk_docs
4 curl -d @servicos.json -H "Content-type: application/json" -X POST
  ↪ http://localhost:5984/servicos/_bulk_docs
```

Com estes comandos temos todas as ferramentas necesssárias para o funcionamento da nossa aplicação, tanto no Windows como no Linux.

3.4 Rodando a aplicação

Agora precisamos que entre na pasta do projeto pelo terminal.

```
1 cd ReactPetShop #Tanto Windows quanto Linux
```

Rode o projeto instalando suas dependências:

```
1 npm install && npm start      #Windows
2 npm install && npm start      #Linux
```

Observação: caso, não consiga rodar o comando `npm install`, exclua a pasta `node_modules` da raiz do projeto e tente novamente os comandos anteriores.

O projeto é aberto no navegador padrão assim que tudo estiver pronto, caso isso não ocorra, é possível acessá-lo através do *link*: http://localhost:3000/

4 Funcionalidades

4.1 Página de *index*

Na página inicial o logo da aplicação sendo exibido corretamente e, de forma global, redireciona o usuário para o *index* ao clicar.

A *navbar* está funcional funcional, sendo possível acessar qualquer categoria desse menu. Na área de usuário, quando o usuário está "logado", as opções de perfil de usuário, carrinho e sair são acessíveis. Já na barra de busca, temos uma busca funcional para buscas exatas, conforme descrito no item 2.1.1.1.

Todos produtos, de todas as categorias, estão sendo exibidos com suas respectivas informações e o produto listado é acessível para área do produto, mostrando suas devidas informações.

4.2 Página de produto

Exibe o produto selecionado corretamente, permite a mudança das imagens armazenadas no banco e exibe suas informações contidas no banco.

O botão comprar adiciona o produto no carrinho e já redireciona o usuário para a página do carrinho, se tiver feito o *login*, caso não tiver com a sua conta "logada", ele redireciona para o carrinho, mas avisa que precisa estar "logado" para vizualizá-lo. Obs: ao logar, seu carrinho será exibido com o produto selecionado previamente.

4.3 Página do carrinho

A página exibe os produtos que estão no carrinho, se caso tiver produtos. Permite mudar a quantidade de cada produto, permite exclusão dos produtos do carrinho, porém quando uma exclusão é feita, não é feito a atualização automática da área do carrinho.

O botão fechar carrinho, finaliza uma compra, removendo todos os produtos do carrinho e redireciona o usuário para a página principal. No entanto a compra não é salva em banco.

4.4 Página de serviço

Todos os serviços são exibidos corretamente com todas as suas informações trazidas do banco e cada serviço tem o seu botão agendar funcionando que, ao clicar, redireciona o

usuário para a página de agendamento do serviço selecionado, se caso o usuário estiver "logado".

4.5 Página de agendamento

O serviço que foi selecionado previamente é exibido com todas as suas informações. O usuário "logado" é reconhecido e seus *pets* são listados, permitindo seleção do *pet* para agendamento.

A escolha de data e hora está funcional, porém sem restrições. Já o valor do serviço é exibido, porém o agendamento não é registrado em banco ao clicar em agendar e o usuário é redirecionado para a página inicial.

4.6 Página de busca

A página de busca é acessada através de uma busca realizada na barra de busca, mostrando os produtos que foram encontrados pela ação realizada. Os produtos são exibidos corretamente.

Um subtítulo da página exbibe corretamente o que foi buscado e todos os produtos com este nome são exibidos e podendo ser acessados clicando em sua miniatura.

4.7 Página de usuário

Ao entrar com seu usuário e senha, a área de *login* é modificada, permitindo o acesso ao perfil do usuário. As informações de usuário são buscadas e exibidas corretamente, com todos os seus *pets*, que são listados, podendo ser selecionado para acessar suas informações no perfil do *pet* escolhido.

Caso este usuário seja um administrador, uma área do administrador é exibida, caso contrário apenas são listadas áreas de compras e serviços realizados. Se o usuário for um administrador, ele pode acessar as áreas de inserção de produtos, serviços e usuários.

Compras e serviços realizados não são exibidos, assim como a área do administrador não é funcional. Ao clicar no botão de adicionar *pet*, o usuário é redirecionado para um formulário de cadastro do *pet*.

4.8 Páginas de cadastro

As páginas de cadastro de produtos, usuários e serviços são acessadas somente por um administrador, já o cadastro de *pets* pode ser feito por qualquer tipo de usuário.

As páginas de cadastro estão com todos os campos funcionais, porém o botão cadastrar não tem funcionalidade.