Relatório Projeto Web

Relatório do projeto do PetShop da disciplina de Introdução ao Desenvolvimento Web

Aluno: Luiz Henrique Lourenção
Nusp: 10284862

1 - Requisitos

Os requisitos do sistema web estão descritos na especificação do projeto.
Aqui estão alguns dos requisitos:
Usuário:

- Login
- Cadastrar
- Cadastrar/Remover Pet
- Visualizar Produtos e Serviços
- Comprar Produtos
- Remover Produtos do Carrinho
- Agendar Serviços
- Deletar a própria conta

Administrador:

- Login
- Cadastrar/Remover Funcionários
- Cadastrar/Remover/Alterar Produtos e Serviços
- Visualizar Produtos e Serviços

2 - Descrição do Projeto

Meu projeto cobre todos ou quase todos os requisitos tanto na parte do cliente quanto na parte do servidor. Para isso, quando um usuário faz o login, uma token fica “presa” a ele na parte do cliente, dessa forma é possível diferenciar um usuário sem cadastro, um usuário comum cadastrado e um usuário administrador cadastrado. O usuário sem cadastro só tem acesso à funcionalidade de visualizar produtos e serviços. O usuário cadastrado comum tem acesso às funcionalidades de comprar produtos, agendar serviços, cadastrar pets e etc. E o usuário administrador tem acesso às funcionalidades de cadastrar/remover funcionários, produtos e serviços.

3 - Comentários sobre o código

No meu projeto é possível realizar todas as operações requeridas na descrição do projeto. Porém não consegui utilizar imagens no banco de dados. Para essa parte, eu carrego o nome das imagens no banco de dados e tenho elas estáticas na pasta “@/assets”, porém isso implica no fato de não ser possível adicionar novas imagens, o que é uma pena. Além disso, acredito que a lógica que eu utilizei na agenda não foi muito boa. Do mais, o resto do projeto está funcionando bem.

Também não consegui “acertar” a estilização. Eu deixei a estilização igual estava na parte 1, só css puro sem framework, pois não consegui ter tempo de aprender algum framework de css e aplicar neste projeto. Até porque acho que se eu tivesse mais tempo eu melhoraria outras partes do projeto e não a estilização (que está satisfatória).

4 - Plano de Testes

Pensando em um(a) avaliador(a) que vai iniciar um banco de dados do zero:
O(A) avaliador(a) deve acessar a página “RegisterAdm”->http://localhost:8080/#/registeradm para se cadastrar como ADMINISTRADOR do site. Esta página não é acessível por links diretos do site pois ninguém deve ser capaz de se cadastrar como administrador.
Após se cadastrar como administrador, o(a) avaliador(a) poderá realizar todas as operações referentes ao administrador, como: cadastrar/remover/alterar produtos e serviços, cadastrar/remover funcionários e etc..

Um problema é que o(a) avaliador(a) tem que, obrigatoriamente, colocar o nome de alguma imagem presente na pasta “@/assets” para cadastrar produtos e/ou serviços (apenas produtos e serviços). Ex: no campo imagem colocar “banhopetimg.jpg”.

Após cadastrar quantos produtos/serviços/funcionários quiser, o(a) avaliador(a) pode clicar em LogOut e se registrar normalmente como um usuário comum. Nesse ponto, os produtos e serviços ja existirão para serem mostrados para o usuário. Caso você não se cadastre anteriormente como administrador para criar produtos e serviços, as páginas referentes a eles não terão conteúdo.

Por fim, depois de testar todas as funcionalidades do usuário comum, como o carrinho e a agenda, o(a) avaliador(a) pode também deslogar e ver que o usuário sem cadastro não consegue realizar as operações do usuário com cadastro.

5 - Informações para execução

Primeiramente o(a) avaliador(a) deverá entrar na pasta “nodestudy” e executar “npm install” para instalar as dependências do backend. Depois, repetir o processo na pasta “frontend” para instalar as dependências do frontend.

Caso necessário, o(a) avaliador(a) deverá alterar as informações da linha 14 do arquivo “app.js” da pasta “nodestudy” referentes ao banco de dados. Em especial o(a) avaliador(a) pode mudar o nome do banco que é a parte que se encontra antes de “?retryWrites=true…” e depois de “mongodb.net/”. Desta forma, um novo banco de dados será criado para que o(a) avaliador(a) consiga testar todas as funcionalidades do início e sem informações anteriores presentes no banco de dados.

Depois, o(a) avaliador(a) deverá entrar na pasta “nodestudy” e executar “npm run dev” para rodar o backend na máquina e conferir se está tudo funcionando. As seguintes mensagens devem aparecer no prompt de comando: “API rodando na porta 3000” e “Mongo connection OK”. Certifique-se de que o mongoDB está conectado corretamente e que o server está rodando na porta 3000, pois as requisições do frontend são feitas nesta porta.

Depois disso, o(a) avaliador(a) deverá entrar na pasta “frontend” e executar o comando “npm run serve” para rodar o frontend na máquina. Este comando vai disponibilizar um link para http://localhost:8080/ , onde o frontend estará funcionando.

Depois disso é só utilizar o sistema da maneira descrita no tópico 4 deste documento.

7 - Problemas

Enfrentei vários problemas durante o desenvolvimento deste projeto. Primeiramente eu não tinha um conhecimento tão profundo em desenvolvimento web e tive que estudar todas as partes tanto do backend quanto do frontend e também da integração entre os dois para desenvolvê-lo. Foi bem difícil porém gostei do resultado e sinto que aprendi.

Além disso tive problemas com as imagens no banco de dados e acabei não conseguindo fazer essa parte.

Também tive problemas com a parte da Agenda. Não estava conseguindo pensar em algo para torná-la prática e usável, porém consegui deixá-la de um jeito aceitável.

Fora isso, todos os problemas que eu encontrei durante o desenvolvimento, foram resolvidos com a ajuda de alguns colegas, aulas ou informações na internet, além de estudo da minha parte.

8 - Comentários

Tenho que comentar que existe um texto na descrição do projeto “Pet Shop Final Version” que eu não consegui entender e não sei o que deveria ser feito.

O texto é:

- The source code properly formatted and commented on. Html, CSS, and javascript files with the application code. The code should be compiled in a distributable form. If using TypeScript, include fonts and compile with SourceMap (`tsc -sourcemap file.ts` OR in `tsconfig.json`). There must be an index.html file.

Provavelmente o meu projeto não responderá a esse requisito e peço desculpas por isso, mas não sei do que se trata.