Miniprojecto 1

Iulia Covalenco nº81719

Tecnologias Dinâmicas para a Internet, 2020/21

Mestrado em Comunicação Multimédia, ramo Multimédia Interativo

Universidade de Aveiro

Índice:

Apresentação do projeto e da temática escolhida

Link repositório: https://github.com/mcmm-mi/tdi-mp1-iulia-covalenco.git

Atualmente, e cada vez mais, se torna urgente a criação de soluções adequadas para o público. Torna-se importante procurar soluções que vão de encontro à necessidade dos diferentes utilizadores, trabalhando produtos e interfaces de plataformas, de forma a permitir a acessibilidade e despertar o interesse de cada um.

Portanto o objetivo principal, no âmbito de desenvolvimento do projeto proposto pelos docentes da cadeira lecionada, é conseguir criar uma plataforma on-line consistente, que tivesse na sua integração tecnologia mais utilizada no desenvolvimento das aplicações de front-end, como o React, Create React App, Redux, API, React Router e Styled Components, permitindo o consumo e visualização da informação proveniente de uma fonte pública de dados.

Assim, foi escolhida a API pública, disponibilizada pela Wizards, a Magic: The Gathering API. Magic, é um jogo de cartas colecionáveis que permite aos jogadores construir baralhos diversificados, que permita derrotar o seu adversário, de acordo com o seu modo individual de jogo.

Design e estrutura da aplicação

Relativamente ao design e as cores implementadas, essas foram escolhidas de acordo com a temática do projeto. Apresenta uma combinação de cores: frias e quentes com a utilização do roxo e azul; utilização do cinzento e branco, com a intenção de criar um maior contraste. Os layouts apresentam uma estrutura simplificada, com o aspeto mais *clean*, sem qualquer ruído visual.

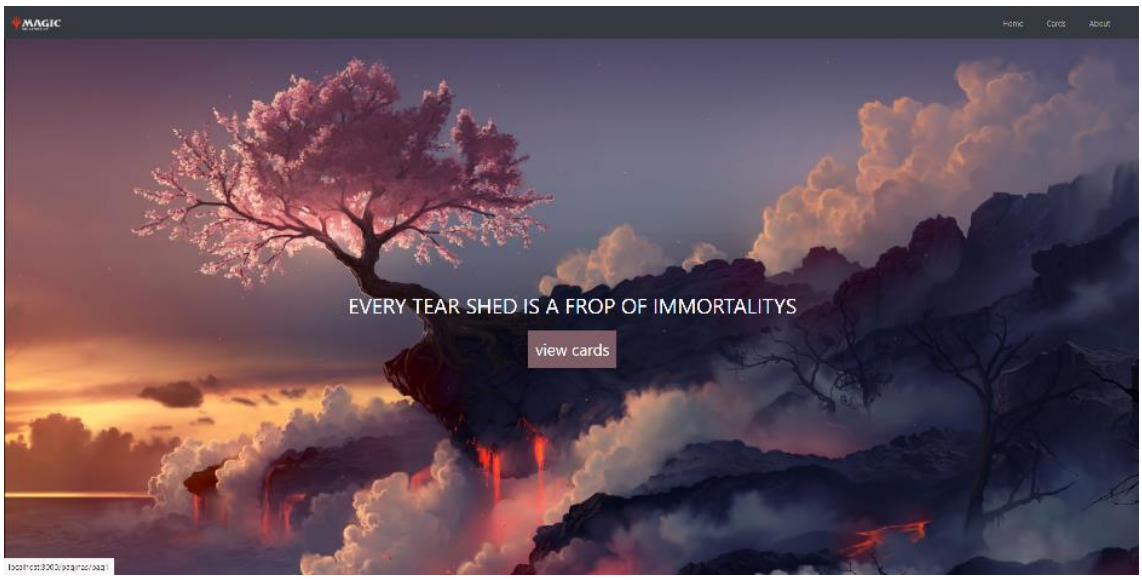

Figura 1 Página Home/Landing Page

A plataforma apresenta uma estrutura simplificada (Figura 2). Dentro da plataforma, podemos encontrar diversos ecrãs, todos com acesso à *NavBar* que permite navegar entre as páginas disponíveis na plataforma.

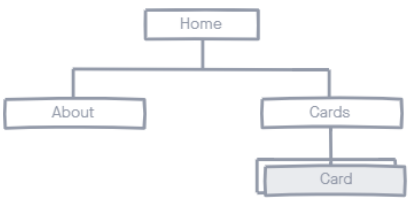

Figura 2 Fluxograma

Da mesma forma, a plataforma permite ao utilizador visualizar, premindo o botão "*view cards*" disponível na *Home (Figura 1),* a *Master List* das cartas importadas da API escolhida. A partir da página *Cards (Figura 4),* o utilizador também pode aceder à página do *Master Ditail*, ou seja, informação especifica de cada carta representada na página *Card (Figura 5),* premindo o botão "*see more*".

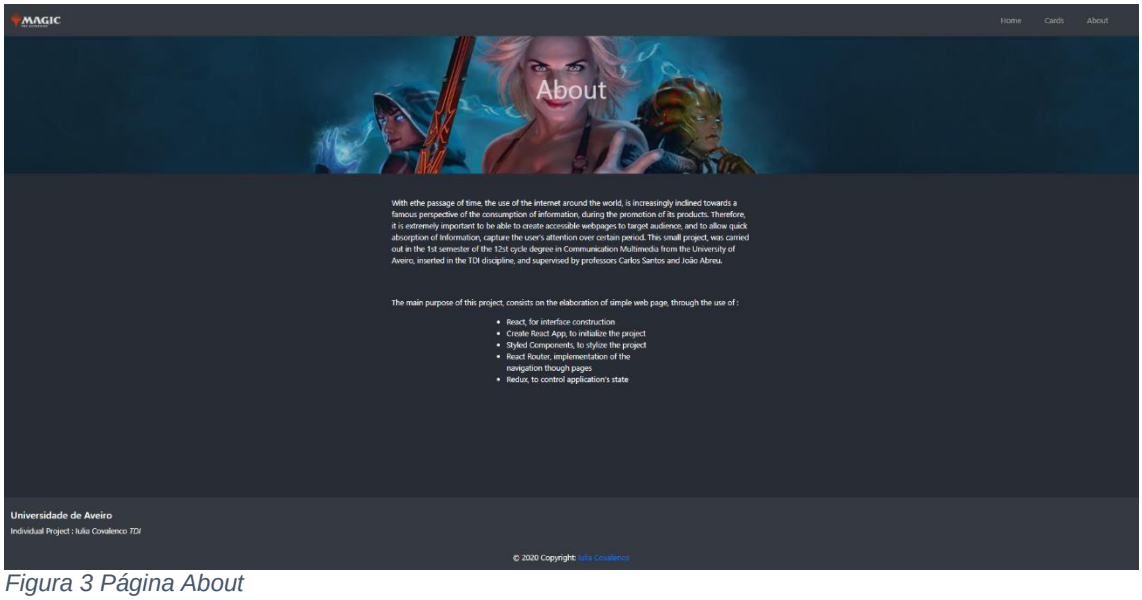

Figura 3 Página About

Relativamente a *NavBar*, anteriormente proferida, essa permite, especificamente aceder à página das cartas, à página *About (Figura 3)*, onde pode ser visualizada pelo utilizador a informação relacionada com o projeto desenvolvido e as tecnologias que foram utilizadas, e voltar para a página *Home*. O logótipo, representado no canto esquerdo da *NavBar* também permite regressar à página *Home*.

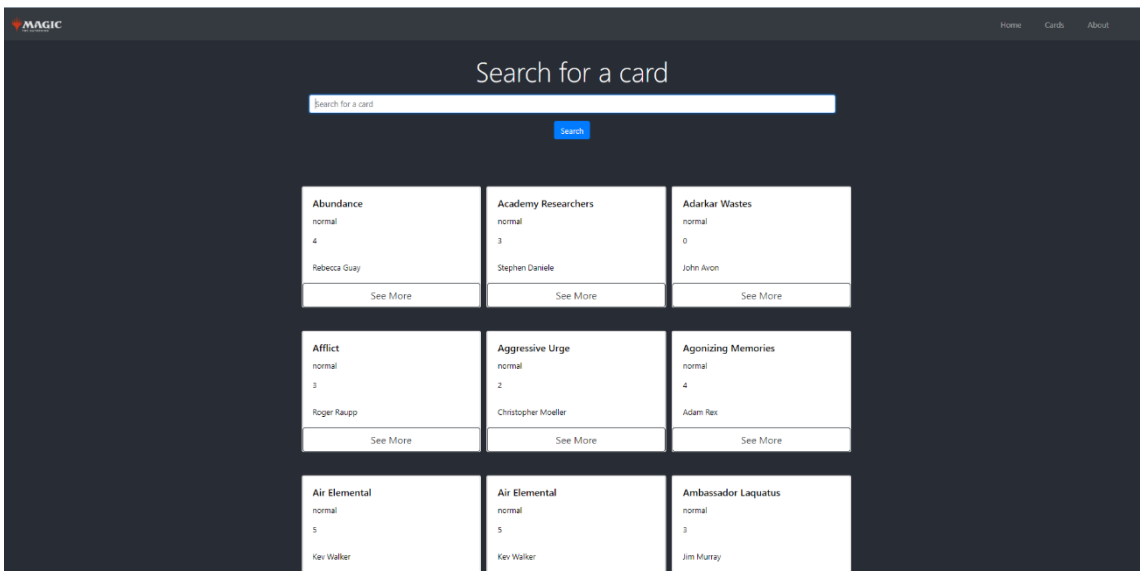

Figura 4 Página Cards

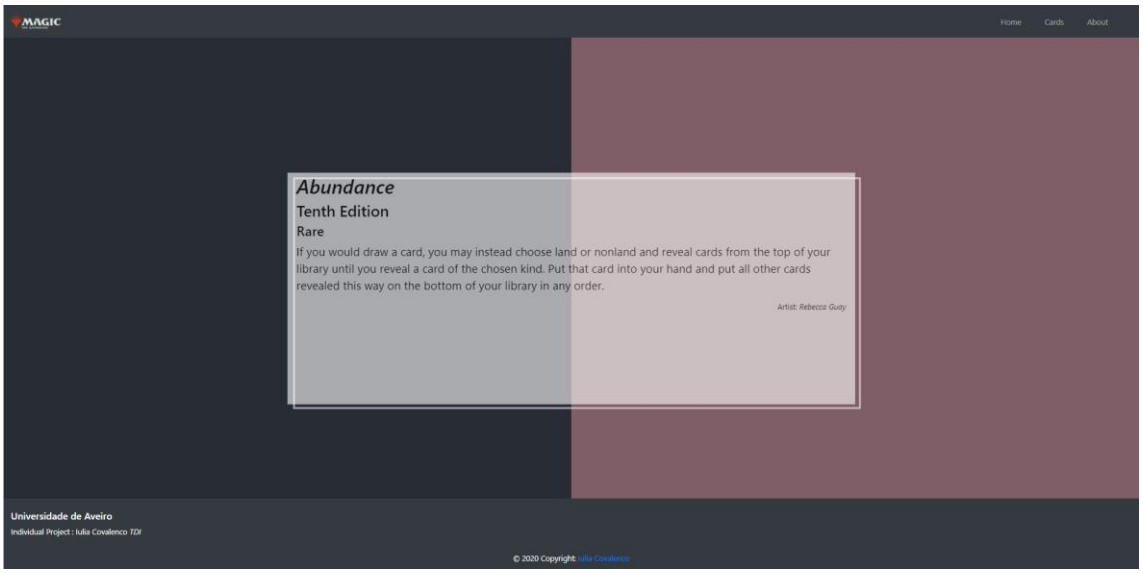

Figura 5 Página Card - permite visualizar a informação detalhada sobre a carta escolhida

Implementação técnica

Relativamente à codificação da página web criada, foi utilizada a linguagem de programação React, com a integração de HTML e CSS. O seu uso consistiu na tradução do design gráfico definido, em linguagem visual consistente, que pudesse vir a ser visualizada e entendida pelos utilizadores via Internet, utilizando o editor de código Visual Studio Code.

Na fase inicial da criação da plataforma, foi utilizado o comando *npm create-react-app*, que permitiu a inicialização do projeto com a estrutura organizacional e logica das pastas e dos ficheiros correta.

Para a criação e a organização dos respetivos elementos usados na implementação da proposta, como os botões, imagens, posicionamento das caixas de texto, recorreu-se à utilização do *react-bootstrap*, com a instalação do ***npm install react-bootsrap*** e a importação das componentes, como o *Container, Footer, Col, Row, Button e InPut*. Quanto à personalização de alguns dos detalhes apresentados nas páginas da plataforma web criada, recorreu-se à utilização de *styled-components*. Foram utilizadas as duas tipologias que permitem formar o layout gráfico da aplicação, pois considerou-se interessante ver a distinção entra as duas componentes, e entender qual dos modelos é o mais eficiente. Sendo assim, concluiu-se que os dois formatos parecem ser adequados, e cada um tem as suas valências, porém a utilização de *styled-components* permite criar layouts mais personalizados.

A navegação entre páginas foi implementada com o recurso à *react-router-dom*, com a importação das componentes *BrowserRouter* as *Routet*, *Route*, *Switch*, *useLocation* e *Link*. Porém, na *NavBar*, nos itens apresentados do lado direito, que permitem ir diretamente para as páginas *“About”, “Cards”, Home”,* recorreu-se a utilização do *tag <a href=”…”>”.* Comprando os dois métodos utilizados, concluiu-se que a utilização de componentes de react-router-dom, é o mais apropriado, pois permite uma *single-page web application*, que não obriga a fazer *refresh*.

Por último, a separação do estado global e local da aplicação, foi feita com a utilização do Redux, permitindo também deste modo manter o estado da plataforma

inteira numa arvore de estado imutável, que não pode ser alterada diretamente. Quando algo muda, um novo objeto é criado, utilizando *actions* e *reducers. O fetch* e o *dispatch* da informação da API escolhida também foi feito com a utilização do Redux. O *dispatch*, método que é usado para fazer o *dispatch* das *actions* e desencadeia alterações na store.

Relativamente à seleção da informação que iria ser disponibilizada para o utilizador sobre as cartas, a partir da API escolhida, foram considerados os valores de: na página cards: i*d, name, layout, cmc, artist, gameFormat; na página card: name, setName, rarity, artist, text.*

Relativamente à organização das pastas, as mesmas ficaram organizadas de modo a que fosse feita a separação e distribuição dos ficheiros das componentes, páginas, imagens e material relacionado com o redux (pasta store que contem actions e reducers, pelas pastas destintas.

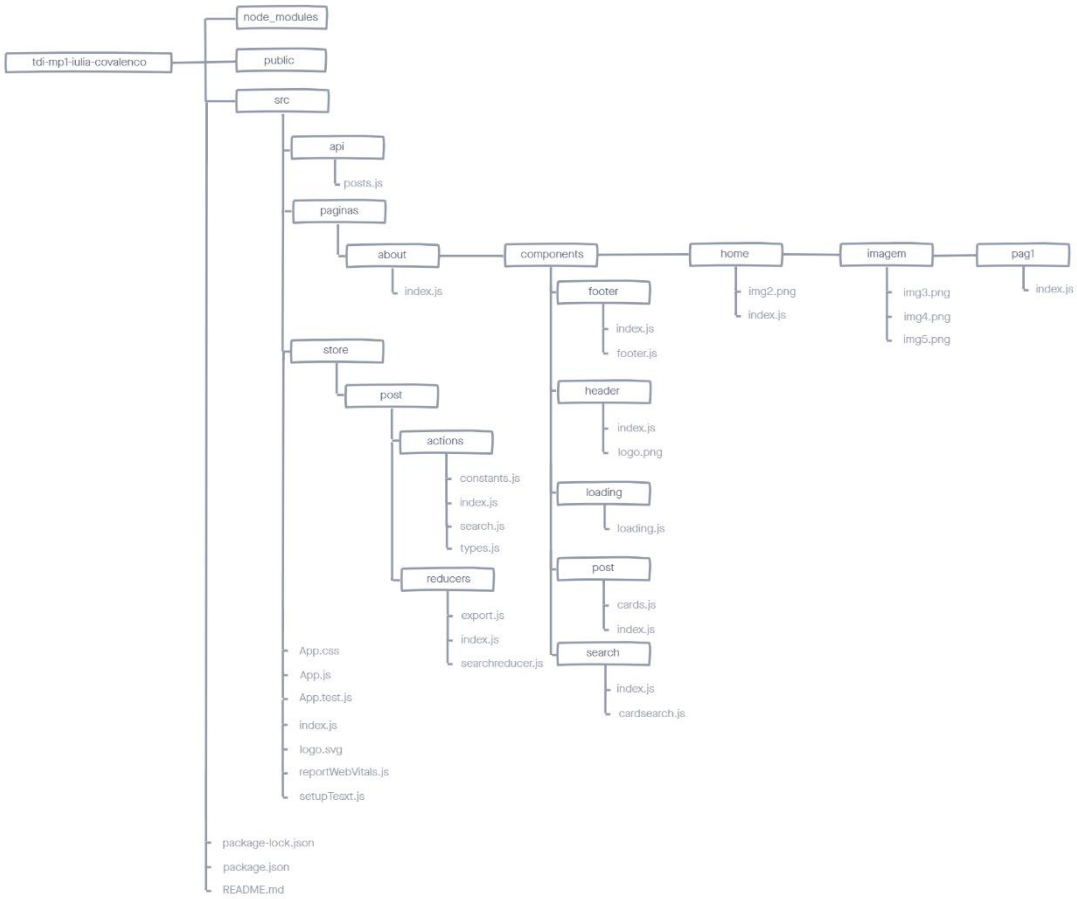

Figura 6 Organização das pastas

Principais desafios

Relativamente aos desafios propostos, estes consistiam na capacidade de o aluno conseguir comprimir todos os requisitos pedidos pelos mesmo, e na integração das tecnologias aconselhadas.

No decorrer da implementação do miniprojecto proposto pelos docentes da cadeira, foram identificados alguns problemas técnicos, que felizmente foram

solucionados, com recurso a ajudas tanto do docente, como de alguma informação disponibilizada nas plataformas on-line como por exemplo, Youtube, StackFlow, Reddit e GitHub. Portanto, como o desafio principal pode ser considerada a compreensão da linguagem utilizada e a sua lógica estrutural.

O primeiro problema encontrado diz respeito à utilização do *react-bootsrap*, mais relacionada com a falta de prática. Essa foi solucionada com recurso à informação adicional na página oficial do *react-bootsrap,* que descrevia as componentes disponíveis e as respetivas classes, e com a observação da documentação que foi posteriormente integrada no projeto, com a instalação do *npm install react-bootsrap*.

Outro problema encontrado estava relacionado com a utilização do *react-redux*. Durante a implementação do mesmo no projeto, surgiram erros relacionados com as funções e as constantes que não foram corretamente definidas na integração da *Api* escolhida *(Figura 7).*

Figura 7 Dispatch

Figura 8 A utilização do Provider

Surgiu também um problema relacionado com a incorreta utilização da *store* e da componente { *Provider }* do *react-redux (Figura 8).* Posteriormente, o problema foi solucionado com a definição correta das constantes e com a restruturação das pastas, pondo a pasta da *store* fora da pasta pag1 e com a inserção da componente *{ Provider }* na página App.js, integrando toda a estrutura lógica da aplicação dentro da componente.

Obstáculos não ultrapassados

Infelizmente, por falta de conhecimentos básicos da linguagem de programação pedida, que nunca foi lecionada e praticada anteriormente, o resultado final, não representa na totalidade a visão inicialmente definida, tanto a nível técnico como a nível visual. Seria necessário redesenhar a plataforma e criar detalhes visuais mais relacionados com a tipologia e a temática em si.

Por último, o maior desafio foi conseguir implementar a opção *search* na página *Cards.* A ideia inicial consistia na atribuição da possibilidade ao utilizador pesquisar a carta pelo seu nome. Para isso, foi procurado um exemplo relacionado com esse modelo de pesquisa e encontrada uma solução mais aproximada. Porém, durante a integração da mesma, surgiram alguns erros relacionados com as funções definidas.

Posteriormente, esse problema foi resolvido, mas o *search* em vez de ir buscar apenas o que foi pedido pelo utilizador, fazia o *fetch* do *Array* completo *(Figura 9).*

Figura 9 Fetch do Array

Entretanto foram feitas alterações mais drásticas, relativas à separação dos *reducers,* e adição de novas componentes. Porém, surgiu um outro problema relacionado com a busca do objeto escolhido e o LOADING *(Figura 10).*

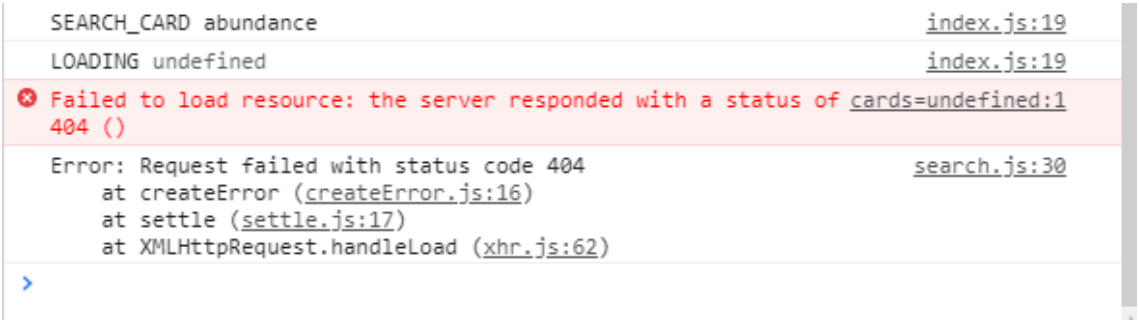

Figura 10 Erro do LOADING undefined

Conclusões

Apesar das dificuldades encontradas durante a construção da plataforma, problemas que não obtiveram uma resolução fiável e a obvia falta dos detalhes visuais mais apelativos, considera-se que a implementação do projeto trouxe bastantes vantagens, como o alargamento das capacidades de compreensão da sintaxe e da lógica da linguagem de programação pedida.

Em suma, tendo em conta que o desenvolvimento da plataforma web em React nunca foi praticado antes, pode-se considerar que o miniprojecto respeitou quase todos os objetivos propostos pelos docentes, conseguindo obter um produto final relativamente funcional a nível de: navegação entre as páginas com a alteração de URL; consumo de informação dinâmica de uma fonte externa; apesenta uma página de listagem de conteúdos; tem na sua estrutura duas páginas de informação estática. Porém seria necessária, tal como já foi referido anteriormente, a implementação de alterações significativas na página *Cards* e tornar a opção *Search* funcional.