Composto de Mudança

Dinis Antunes 102802, Sofia Bermudez 103242

Universidade de Aveiro
(Tecnologias Avançadas para Client-Side)

Abstract

Este relatório descreve o desenvolvimento de uma aplicação web e mobile de monitorização de compostagem comunitária para ser implementada em contextos escolares com recintos de compostagem.

Keywords--- **compostagem, compostagem comunitária, educação ambiental, React, javascript**

1. Introdução

Partimos da necessidade identificada em duas escolas secundárias de Chaves, onde foram nos últimos dois anos instalados recintos de compostagem comunitária, de digitalizar o processo de manutenção e monitorização implicada na gestão de um recinto comunitário que envolve docentes, alunos e funcionários de uma instituição educacional.

Desta forma, de modo a complementar os projetos de educação ambiental existentes nestas escolas e possíveis locais pedagógicos, propusemos o desenvolvimento de uma aplicação que rastreia as deposições de resíduos no compostor ao longo do tempo, assegurando a utilização correta da mesma.

Em resposta ao desafio proposto na unidade curricular de Tecnologias Avançadas para Client-side, de realizar um projeto que apresente uma solução a um problema real, chegamos à aplicação Composto de Mudança

Neste relatório iremos abordar nomeadamente:

- Público-alvo
- Objetivos com a aplicação
- Utilização de Api's
- Autenticação Firebase
- Implementação em código
- Crítica retrospectiva do desenvolvimento
- Planos futuros

2. Relatório

Em contextos como organizações ambientais, escolas, quintas ambientais, esta aplicação viria atender à necessidade de reportar e organizar a participação dos envolvidos na manutenção do compostor, dando maior liberdade aos utilizadores de acederem ao compostor sem consulta prévia, reduzindo os obstáculos que desincentivam a participação colectiva.

2.1. Público-alvo

Trabalhamos principalmente para o público-alvo presente em duas escolas em Chaves: o Agrupamento de Escolas António Granjo e o Agrupamento de Escolas da Fernão de Magalhães, onde com base no projeto fundador FlaviECOmpostor, imaginamos a aplicação de dois utilizadores possíveis a navegar a aplicação: docentes e alunos.

O primeiro utilizador, o docente, poderia criar a conta baseada na escola em que é docente, convidar os seus alunos para uma equipa, adicionar e gerir turnos, e nomeadamente registar entradas no compostor do recinto de compostagem local.

O segundo utilizador idealizado, o aluno, poderia aceder à aplicação através de um código ou convite direto, utilizar o 'Tracker' para registar as entradas no compostor, verificar turnos para a semana e gerir o seu perfil. Para além disso, ainda teria acesso a questionários para testar os conhecimentos de compostagem e tutoriais antes de utilizar o 'Tracker' pela primeira vez.

2.2. Objetivos com a aplicação

O projeto tem como objetivos:

2.2.1. Tracker apresenta-se como uma das principais funcionalidades, porque faz a conexão da nossa solução tecnológica com a realidade destas escolas: o recinto de compostagem.

Este Tracker aplica-se para quando o docente ou aluno estiverem prestes a depositar um resíduo no compostor, possa registá-lo em pouco tempo, podendo

alertar na aplicação que uma entrada foi dada. Desta forma pode-se monitorizar o cumprimento de tarefas de manutenção do processo de compostagem.

2.2.2. Learn More apresenta-se como um componente com potencial para contribuir na digitalização da aprendizagem e apoiar com recursos educativos (nomeadamente questionários) para utilizar em contexto de aula ou para apoio extra-curricular, gerido pelo docente.

2.2.3.Turnos é o nome de um componente que permite aos utilizadores adicionar e gerir turnos num calendário, para saber a que utilizadores estão delegadas as tarefas.

2.2.3. Perfil como o nome indica, é uma secção exclusiva a cada utilizador onde podem ver as contribuições dadas ao longo do ano no recinto de compostagem, a atividade do 'Tracker' mais recente e ainda verificar que labels tem associada (de equipa, de papéis de coordenação)

2.2.3. Gestão de Equipas é uma secção ainda por desenvolver que permite aos docentes gerir os seus alunos, os seus papéis para a semana e interagir com os seus alunos.

2.2.3. Utilizadores é conseguir que a aplicação tenha duas interfaces privadas diferentes: uma para o asprofessor e outra para o aluno (com funcionalidades possíveis diferentes para cada)

2.3. Implementação em React

Utilizámos a biblioteca de código React, própria para o lançamento de aplicações rápidas de projetos front-end, nomeadamente a versão 18.2.0. As linguagens utilizadas foram ECmascript 6 e CSS.

Adicionalmente, tivemos o cuidado de assegurar responsividade para uma aplicação em *mobile* e *desktop.*

O objetivo foi abordar o máximo de conteúdos lecionados na componente teórica da unidade curricular Tecnologias Avançadas para Client-Side, sendo que as principais abordadas foram:

2.3.1. Hooks foram utilizados de forma ubíqua, com o *useState* e o *useEffect,* na alteração de botões consoante a interação do utilizador, ao guardar os dados inseridos por utilizadores, e para a autenticação Firebase na leitura dos dados para e-mail e password.

2.3.2. Routers foram utilizados com o módulo respectivo instalado, para a conexão na App.js (ficheiro pai) dos diferentes componentes através do Route e Link também provenientes do módulo Router-DOM. Desta forma, é possível criar uma aplicação de poucas páginas que permite a sua navegação sem ser necessário refrescar, tornando o projeto mais leve e rápido.

2.3.3. Rest Operators foram principalmente utilizados na criação do componente Calendar do módulo React Big Calendar, para quando se adiciona um novo turno criado por parte do utilizador no Calendário.

2.3.4. Class Components foram utilizados na criação do mapa com as localizações das diferentes escolas.

2.3.5. Function Components foram a principal tipologia de componente utilizada pela grande maior das páginas da nossa aplicação.

2.3.6. Event Listeners de ES6 foram utilizados para guardar os dados introduzidos em variados elementos do DOM como na tag <input> e <select>.

2.3.7. Try…Catch foi implementado na deteção de erros de autenticação de utilizadores, no contexto do Firebase Auth.

2.3.8. Arrow Functions foram utilizadas sempre que possível, por exemplo em edição *inline* de componentes e elementos do DOM e na definição dos componentes funcionais.

2.3.9. Exports and Imports foram utilizados de forma ubíqua para realizar a gestão de componentes nas diferentes páginas, de acordo com a necessidade, permitindo o 'nesting' dos componentes num só diretório.

2.3.10. Módulos

Foram adicionais bibliotecas necessárias (módulos) como por exemplo:

- Big React Calendar
- Bootstrap para React
- MapBox,
- DatePicker,
- Firebase
- React-map-gl
- Router Dom
- Date Fns
- Github Contribution Calendar

2.4. Implementação de Api

2.4.1.Firebase foi implementado com a componente de autenticação para o registo de novos utilizadores e também a entrada de utilizadores já existentes.

2.4.1. Mapbox foi utilizado para gerar o Mapa, onde introduzimos o Marker para geo-localizar as escolas no mapa, cruzando dados com um ficheiro .json manualmente introduzido por nós, onde se encontram as diferentes coordenadas das escolas introduzidas no projeto.

Foi também utilizado o *setPopup* do *mapboxgl* para introduzir um *Popup* que, ao clicar no componente Marker, aparece o nome da cidade e a morada da escola em questão, dados estes provenientes do ficheiro.json previamente referido.

.

2.4.1. Dicebear é uma API open-source que permite a utilização gratuita de avatares de diferentes estilos e foi utilizada para gerar fotos de perfil para os utilizadores na nossa aplicação.

2.5. Crítica retrospectiva

Após a entrega do projeto, podemos tirar algumas conclusões do desempenho da implementação do projeto.

Primeiramente, compreendemos o que se houvesse um maior investimento em estudos de interface design em Figma, poderíamos ter obtido um melhor resultado estético quando implementado em código tendo em conta também especificidades de design aplicado ao código (estilização pré-definida a cada elemento no DOM), prevenindo assim uma incrementação de escolhas estéticas a meio da fase de desenvolvimento.

Já na fase de desenvolvimento, deparámo-nos com dificuldades em passar dados de componente a componente, não tendo optado por utilizar nenhuma solução back-end (devido ao perfil do grupo) ou solução tecnológica como o Redux, devido à baixa dimensão do nosso projeto.

Dada estas últimas decisões referidas, encontramos algumas limitações em guardar estados entre componentes que não partilhavam relação pai-filho. Quando havia essa relação conseguimos passar valores, como é o exemplo dos dados inseridos no Tracker, para o Feed, na parte da 'Atividade Recente'.

Adicionalmente existe matéria leccionada que poderíamos ter testado no nosso projeto, como por exemplo Promises.

Para além destas adaptações tecnológicas, também não cumprimos com um dos objetivos autopropostos de desenvolver uma secção de gestão de equipas, que também implicaria por natureza a passagem de dados de que equipa pertence cada utilizador, de forma a estar presente em virtualmente todos os componentes de forma ou outra.

Contudo, compreendemos que isto também só seria verdadeiramente útil com a implementação de utilizadores com direitos de acesso diferentes (utilizador registado como professor e utilizador registado como aluno), o que nos traz à nossa última crítica construtiva: a pendente implementação de 'role-based routing'.

2.6. Planos futuros

Com o conhecimento e experiência ganha no desenvolvimento deste projeto, ao qual pretendemos incluir em futuros projetos, compreendemos que os próximos passos que acrescentariam valor ao nosso projeto seriam:

- Confirmar que a aplicação serve as necessidades do público-alvo, fazer focus groups e testes no terreno, de forma a sondar as sugestões e críticas para novas ideias de implementação.
- Reformular a abordagem feita ao design de interface, implementando escolhas estéticas que estejam a par do estado da arte e também que apelem adequadamente ao público-alvo. Para isto, também incrementar design de interação, com mais animações.
- Continuar o uso de Firebase para implementar um sistema de notificações que, por exemplo, pode servir para alertar os alunos quando lhes é atribuído um turno para a semana.
- Na mesma ótica, adicionar mais formas de interação entre aluno-docente através de comentários no registo de entradas no compostor, ou também através da utilização de Firebase para criar um sistema de chat direto entre docente e aluno.
- À medida que a aplicação fica mais robusta é necessário pensar numa solução de preservação dos dados, forçando a exploração de soluções como Redux ou inclusive uma solução back-end com a criação de uma base de dados.
- Implementar routing privado através da tipologia de utilizador registado, para que o professor possa ter acesso a diferentes funcionalidades de gestão de equipa e de administração que devem estar fora do alcance de qualquer aluno.

Conclusões

Em retrospectiva, o produto final aproxima-se bastante aos objetivos propostos inicialmente.

Porém, compreendemos que alguns passos são necessários para a aplicação estar preparada para lançamento: para além de torná-la mais robusta com a contínua implementação de mais componentes, é necessário também criar novas formas de interação entre os alunos, os docentes e o recinto de compostagem.

Enquanto que se cumpre o objetivo de apoiar estas comunidades que tentam alcançar a mudança na gestão de resíduos em Portugal, mantém-se a necessidade de manter o desenvolvimento da aplicação viva.

Referências

[1] Mapbox GL - (Version 2.12.0). https://docs.mapbox.com/mapbox-gl-js/guides/

[2] Big React Calendar- módulo React (Version 2.0) http://jquense.github.io/react-big-calendar/examples/

[3] Dicebear - Avatar library (Version v5.1.0) .https://github.com/dicebear/dicebear

[4] Firebase - Autenticação console (Version 9.15.0) https://console.firebase.google.com/u/0/project/composto-de-mudanca/authentication/users