Desenvolvimento de Aplicações Web

Luís Duarte a81931, Ricardo Silva a84123, Ulisses Araújo a84010

Universidade do Minho, Dezembro 2020

1 Introdução

O presente relatório descreve o desenvolvimento do trabalho pertencente à componente prática da Unidade Curricular Desenvolvimento de Aplicações Web, tendo este como objetivo aplicar os conhecimentos adquiridos durante o decorrer do semestre. Para o sistema de base de dados utilizamos o MongoDB, que utiliza documentos **JSON** para armazenar a informação, para as interfaces foram utilizados principalmente o **Pug** e o **Bootstrap** e finalmente o controlo foi implementado através de um servidor **HTTP** definido em **NodeJS**.

2 Aplicação

A aplicação que nos foi pedida para desenvolver consiste num sistema de gestão e disponibilização de recursos educativos, sejam estes imagens, slides, ou de qualquer outro tipo. Sobre estes recursos é também possível aos utilizadores criar posts. Finalmente, existe também um sistema de ranking para estes posts e recursos.

2.1 Funcionalidades

A aplicação oferece uma variedade de funcionalidades. É possivel aos utilizadores criarem contas contendo algumas informações sobre os mesmos, podendo posteriormente manusear então os recursos e proceder à criação de posts. Uma vez criada a conta, um utilizador é capaz de:

- Editar o seu perfil, sendo-lhe possível trocar a sua foto de perfil, password, posição e curso.
- Carregar recursos para o sistema, atribuindo-lhes título, subtítulo, tipo e uma data de criação. Podendo este recurso ser disponibilizado publicamente ou não. Posteriormente pode também editar esse recurso, removê-lo do sistema ou mudar o seu nivel de privacidade.
- Criar posts relativos a recursos presentes no sistema, sejam estes recursos seus ou de outrém, podendo estes ser editados posteriormente.
- Adicionar um recurso ou post aos favoritos, afetando estes os sistemas de ranking. Ficam também disponibilizados numa lista de favoritos para o utilizador aceder no seu perfil.
- É também possivel ao utilizador consultar e filtrar as listas de todos os recursos, posts e utilizadores no sistema.
- Nas páginas dos recursos, é possível pré-visualizar alguns dos tipos de ficheiros podendo estes ser descarregados individualmente, ou simplesmente descarregar os ficheiros pertencentes ao recurso todos de uma vez.
- Fazer e apagar comentários nos posts.
- Existe também um administrador que possui a habilidade de remover qualquer recurso, post e comentário disponível no sistema.

3 Modelos

Para interagir com a nossa base de dados implementada em **MongoDB** utilizámos a biblioteca *Mongoose*, que permite definir schemas e efetuar pedidos à base de dados. Visto a nossa base de dados ser composta por três coleções: **users**,**posts** e **resources**, definimos então os seguintes schemas:

- **Resource:**

```
const resourceSchema = new mongoose.Schema({
    title: String,
    subtitle: String,
    type: String,
    producer: {_id: ObjectId, name: String},
    files: [Object],
    createdAt: Date,
    registeredAt: Date,
    downloads: Number,
    favs: Number,
    public: Boolean
},{versionKey: false})
```

- **Post:**

```
const postSchema = new mongoose.Schema({
    title: String,
    subtitle: String,
    themes: [String],
    content: String,
    views: Number,
    favs: Number,
    producer: {_id: ObjectId, name: String},
    resources:[{_id: ObjectId, title: String}],
    comments:[{user: {_id: ObjectId, name: String}, text: String, createdAt: Date}],
    createdAt: Date,
},{versionKey: false})
```

- **User:**

```
const userSchema = new mongoose.Schema({
    name: String,
    email: String,
    password: String,
    position: String,
    course: String,
    level: String,
    dateReg: Date,
    lastOnline: Date,
    favs: Number,
    favouriteResources: [{_id: ObjectId, title: String}],
    favouritePosts: [{_id: ObjectId, title: String}]
},{versionKey: false})
```

4 Controlo

Para a nossa aplicação foram desenvolvidos vários *routers* sendo estes:

Router Index (GET):

- / - página inicial da aplicação onde é possível ao utilizador iniciar sessão ou registar-se.

Router Auth (GET):

- /login - Página onde o utilizador pode inserir as suas informações e fazer login
- /logout - Logout de um utilizador
- /signup - Página onde o utilizador pode inserir as suas informações e registar-se na aplicação

Router Auth (POST):

- /login - Login de um utilizador
- /signup - Registo de um utilizador

Router Posts (GET):

- / - Página onde o utilizador pode visualizar e filtrar todos os posts disponíveis no sistema
- /upload - Página onde está disponibilizado o formulário para a criação de um novo post
- /:id - Página do post com o respetivo id onde estão dispostas todas as suas informações e onde é possível comentá-lo
- /:id/resources - Devolve os recursos do post com o id dado
- /edit/:id - Disponibiliza a página de edição do post

Router Posts (POST):

- /upload - Faz o carregamento de um post novo
- /edit/:id - Edita o post com o dado id
- /:id/comment - Publica um comentário no post com o id dado

Router Posts (DELETE):

- /:id - Apaga o post com o id dado.
- /:id/:idcomment - Apaga o comentário com id == idcomment do post dado por id.

Router Resources (GET):

- / - Página onde o utilizador pode visualizar e filtrar todos os recursos disponíveis no sistema
- /upload - Página onde está disponibilizado o formulário para o carregamento de um novo recurso
- /:id - Página do recurso com o respetivo id onde estão dispostas todas as suas informações
- /:id/files/* - Realizar download de um ficheiro especifico contido dentro do recurso com o id dado, e caminho igual a *.
- /edit/:id - Disponibiliza a página de edição do recurso
- /:id/download - Download do recurso completo incluindo o manifesto

Router Resources (POST):

- /upload - Faz o carregamento de um recurso novo
- /edit/:id - Edita o recurso com o id dado

Router Resources (DELETE):

- /:id - Apaga o recurso com o id dado

Router Resources (PATCH):

- /:id - Atuliza a informação do recurso com o id dado, nomeadamente é utilizado para mudar a visibilidade do recurso.

Router Users (GET):

- / - Página com todos os users registados no sistema
- /profile - Perfil do utilizador logado, permitindo ver os posts e recursos que este criou, permitindo-lhe editá-los/apagá-los, editar o seu perfil e ver as suas listas de favoritos
- /admin - Página exclusiva ao administrador onde pode remover qualquer utilizador, post ou recurso do sistema
- /edit/:id - Página onde apresenta o formulário para editar o utilizador
- /:id - Página de um dado utilizador
- /:id/picture - Obtém a foto do utilizador

Router Users (POST):

- /:id/favouritesPosts - Adiciona um post aos favoritos do utilizador
- /:id/favouritesResources - Adiciona um recurso aos favoritos do utilizador

Router Users (DELETE):

- /:id/favouritesPosts/:postid - Remove um post dos favoritos do utilizador
- /:id/favouritesResources/:resid - Remove um recurso dos favoritos do utilizador
- /:id - Remove um utilizador do sistema

5 Interface

Para a implementação da interface desta aplicação utilizamos a linguagem **Pug** e o *Framework* **Bootstrap** juntamente com **JQuery** para proporcionar ao utilizador uma interface amigável e interativa.

6 Povoamento inicial

De forma a que o grupo conseguisse testar de forma confiável e realista a aplicação desenvolvida foi criado, recorrendo á linguagem **Python**, um programa capaz de criar ficheiros com formato **JSON** para povoar a base dados com os metadados de centenas de utilizadores, recursos e posts. O programa cria ainda, de forma automática, os recursos a serem usados pelo sistema, tendo estes uma carga considerável, de cerca de 5GB.

7 Conclusão

Com a realização deste projeto foi possível criar uma aplicação na qual é disponibilizado um sistema de gestão e partilha de recursos educativos. Neste sistema disponibilizamos também um sistema de posts e comentários sobre os recursos carregados existindo sobre estes um sistema de ranking definido através do número de favoritos ou visualizações. De um modo geral, encontramo-nos agradados com o trabalho que desenvolvemos e o resultado que obtemos. Porém como em tudo, existem aspetos a melhorar, seja a adição de novas funcionalidades ou a melhoria das que já estão presentes na aplicação, como por exemplo, optar por uma arquitetura de microserviços ou apostar em mais formas de autenticação.