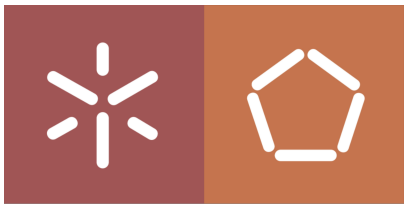

Universidade do Minho
Escola de Engenharia
Departamento de Informática

Repositório de Recursos Didáticos (RRD)

Relatório de Desenvolvimento

Mestrado em Engenharia Informática

Representação e Processamento de Conhecimento na Web

Desenvolvido por:

António Santos **pg47031**
Eduardo Silva **pg47167**
José Magalhães **pg47355**
Rúben Cerqueira **pg47626**

19 de junho de 2022

Conteúdo

Lista de Figuras

1. Introdução

No âmbito da unidade curricular Representação e Processamento de Conhecimento na Web, enquadrada no perfil de Engenharia de Linguagens, foi proposto elaborar uma aplicação que representa um repositório de recursos didáticos. Esta tem como objetivo o armazenamento de ficheiros de conteúdo académico de forma a facilitar a partilha destes e facilitar a aprendizagem dos utilizadores do sistema.

A aplicação desenvolvida segue as orientações do modelo OAIS *(Open Archival Information System)*. Este modelo tem como objetivo generalizar aplicações que se enquadrem no género de armazenamento de ficheiros, estabelecendo uma norma eficiente de processamento de dados.

A solução proposta tem por base a aplicação prática das várias ferramentas lecionadas nas aulas, desde a implementação do servidor ao front-end da aplicação.

Para o desenvolvimento da aplicação foram usadas várias ferramentas *open-source*, tais como W3 e Pug - predominantemente para a componente front-end - e NodeJS, Express, Axios e Multer como ferramentas cruciais importantes na programação do back-end do sistema.

Acrescenta-se que a arquitetura final proposta, bem como todas as decisões tomadas pelo grupo para a construção desta, se encontram devidamente justificadas e documentadas no presente relatório.

2. Arquitetura

De forma a modularizar a solução proposta foi decidido que a aplicação se iria dividir em três partes: **API**, **AUTH** e **APP**, como é possível verificar na figura 2.1.

A **API**, tal como o nome indica, corresponde a uma aplicação API que trabalha maioritariamente com os dados do sistema. Esta fornece e guarda dados que são solicitados ou submetidos por aplicações externas, se tiverem as devidas permissões. Logo, é a camada mais próxima da base de dados em que vai operar.

A **AUTH** constitui a parte de autenticação, tendo como objetivo gerir as contas criadas no sistema, aplicando métodos de autenticação e fornecimento de *tokens* que darão acesso a utilizadores a zonas com a permissão devida, bem como a capacidade de realizar pedidos à interface de API.

A **APP**, por sua vez, será a aplicação geral, ou seja, vai comunicar com as restantes para conseguir responder aos pedidos dos clientes.

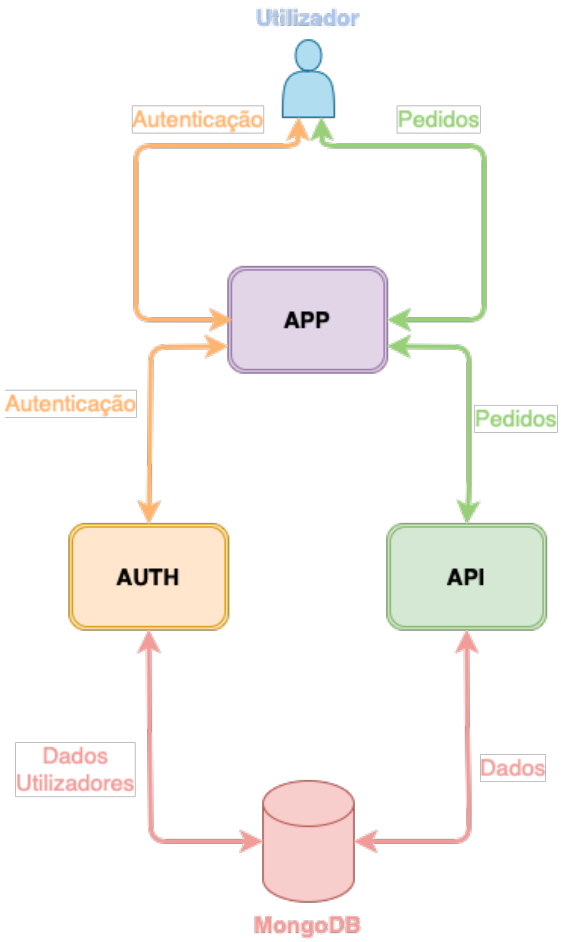

Figura 2.1: Arquitetura da Aplicação

3. Funcionalidades

3.1 Utilizadores

A aplicação construída, tal como foi abordado, segue a norma OAIS. Esta conterá 3 tipos de utilizadores, cada um com o seu respetivo nível de acesso à plataforma, variando este de acordo com as operações autorizadas:

- **Administrador** - Poderá realizar todas as operações do sistema, desde consumir dados, produzir dados até operações de administração como remoção de contas ou conteúdo. Possui o nível de permissões mais elevado.
- **Produtor** - Exige o registo e autenticação na plataforma, podendo então editar o seu perfil, adicionar recursos e editar ou remover os seus recursos previamente adicionados. Para além disso, pode visualizar, fazer *download*, comentar e classificar recursos. Possui um nível intermédio de permissões.
- **Consumidor** - Não lhe é permitido editar, remover ou adicionar recursos, nem comentar as publicações envolvendo os mesmos, sendo apenas possível visualizar ou fazer *download* de recursos. Não necessita de registo e possui o nível de permissões mais baixo.

3.2 Autenticação

De forma a garantir um mecanismo de autenticação seguro utilizou-se a ferramenta *open-source* **JSONWebToken**, crucial para uma partilha eficiente de informações de segurança entre duas partes — um cliente e um servidor pois, cada JWT contém objetos JSON codificados, incluindo um conjunto de declarações manipuláveis.

Quando um utilizador se autentica na plataforma, é gerado o respetivo **JSONWebToken** (JWT) e é adicionado aos seus *cookies* um campo ***token*** que, para além do JWT, possui informação relativa ao utilizador em questão, como o seu nome, nível e identificador. Esta informação é utilizada ao longo de toda a navegação pela aplicação.

Já o nível de cada utilizador varia consoante o tipo do mesmo: se for um administrador, é-lhe atribuído o nível de **administrador**. Se este utilizador, ao autenticar, não for administrador, é-lhe atribuído automaticamente o nível de **Produtor**.

Caso um utilizador da aplicação **não** efetue *login*, é-lhe atribuído um *token* com o nível de **Consumidor** e, neste caso, o *token* não possui qualquer informação extra sobre o utilizador, uma vez que este não está autenticado.

3.3 Ingestão de Dados

O processo de ingestão de dados é uma das partes mais importantes do programa uma vez que permite introduzir dados para que o sistema possa operar e processar. O método consiste na apresentação de um formulário a preencher pelo utilizador, passando depois todo o processo pela interpretação do mesmo.

Ao utilizador, é-lhe exigida a autenticação no sistema, para conseguir proceder à introdução de informação na aplicação pois, desta forma, garante-se sempre que este terá um dos seguintes níveis: administrador ou produtor. No formulário em questão são pedidos os metadados necessários para a criação de uma publicação, bem como os ficheiros que o utilizador pretende introduzir no sistema pela publicação que deseja criar.

Tendo então o formulário preenchido, com os diversos ficheiros carregados, esta informação é pré-processada no *front-end* antes de ser enviada para a aplicação. Este pré-processamento segue uma norma específica, conforme sugerido pelo enunciado - **BagIt**, que consiste no empacotamento dos ficheiros num ficheiro comprimido com uma estrutura específica, envolvendo também a gravação da *checksum* de cada ficheiro para verificações de integridade dos dados que são enviados.

Cada SIP *(Submission Information Package)*, correspondendo este às estruturas BagIt que adicionam informação ao sistema, é composto por um ficheiro *zip* com a seguinte estrutura:

- Um ficheiro de versão do BagIt com metainformação, como o *encoding*;
- Um ficheiro de metadados em que em cada linha constará um *checksum* e o ficheiro associado a esse mesmo *checksum*, estando estas informações separadas por um espaço;
- Uma diretoria chamada **data** que conterá todos os ficheiros envolvidos.

Este ficheiro *zip*, no fim da sua criação, é enviado, então para o servidor. Já o servidor, ao receber o ficheiro, vai desempacotá-lo, procedendo para a verificação dos *checksums* de cada um dos ficheiros contidos na diretoria **data**. Caso a verificação falhe, o processo é cancelado e os dados são removidos, caso contrário, os ficheiros são armazenados no *FileSystem* e os seus metadados são armazenados num *Schema* apropriado, elemento constituinte da base de dados **MongoDB**, tornando-se objetos AIP *(Archival Information Package)* do modelo.

3.4 Recursos

Todo o objetivo do sistema passa não só por estes componentes, mas também pela respetiva gestão e manipulação. Os recursos passam por um longo processo: desde a sua submissão e devida configuração na plataforma, até a uma possível edição das respetivas informações, por parte do Produtor, ou até mesmo a uma visualização e *download* desta unidade elementar, por parte de um Consumidor.

Cada recurso pode ser marcado como público ou privado, pelo que apenas os recursos públicos são exibidos na listagem dos recursos disponíveis para outros utilizadores poderem ver e/ou descarregar.

Quando um recurso é submetido, o produtor responsável por essa operação pode, futuramente, e a qualquer momento, editá-lo (como por exemplo torná-lo público ou privado, consoante o caso), ou até mesmo removê-lo da plataforma. Quanto aos restantes utilizadores estes podem consultar o conteúdo dos ficheiros contidos no recurso em questão, com a possibilidade de os descarregar, desde que este seja público, bem como classificar os mesmos (no caso de serem outros produtores).

3.5 Publicações

Os produtores podem fazer publicações sobre um determinado recurso, onde outros produtores podem inserir comentários (caso esse recurso seja público). Cada publicação possui uma descrição e um título, bem como todos os comentários feitos na mesma, com identificação do respetivo utilizador (produtor ou administrador).

Desta forma, as publicações, para além de possibilitarem que um utilizador exprima uma opinião sobre um recurso, representam também um local de discussão sobre este.

Dado que cada publicação é relativa a um recurso, a plataforma disponibiliza filtros sobre os mesmos:

1. Título, para obter recursos que possuam um dada palavra no título;
2. Tipo, para obter os recursos de um dado tipo.

3.6 Notícias

À medida que os utilizadores vão interagindo na plataforma, são geradas notícias sobre as suas ações, que são apresentadas não só no *feed* de notícias principal, contendo sempre as mais recentes, mas também é possível a sua consulta na totalidade numa página criada para o efeito. São geradas notícias nas seguintes situações:

- Quando é feito o *upload* de um recurso **público**;
- Quando é feito um comentário numa publicação;
- Quando um recurso é classificado;
- Quando um determinado recurso é editado;
- Quando é feito o *download* de um recurso público.

De forma a aproximar a nossa solução de outras alternativas reais, decidiu-se que as publicações e as notícias estão presentes na página inicial da plataforma, visíveis a qualquer utilizador, sendo estas apresentadas de forma cronológica, dando privilégio às mais recentes.

4. Conclusão

De um modo geral, o sentimento que passa é que este trabalho prático, além de extremamente desafiante, ajudou bastante a consolidar a matéria lecionada. Foi reconhecido que a experiência se manifestou muito enriquecedora para o nosso coletivo, pois permitiu que cimentássemos todas as competências não só de desenvolvimento de aplicações Web, mas também da parte relativa ao trabalho em grupo e organização coletiva.

Assumiu-se que houve alguma dificuldade na ligação de todos os conceitos, pois, além de ser uma solução que envolve um grande número de componentes, devido aos extensos requisitos, sentiu-se o sentimento que à medida que o desenvolvimento foi avançado, novos requisitos e problemas apareceriam, aumentando a complexidade do problema elevando a dificuldade de gerir o trabalho, tanto conjunto como individual.

Todas as decisões tomadas pelo grupo, desde as estratégias escolhidas até traços fundamentais da implementação, foram sempre discutidas entre todos, pelo que o grupo acha que a solução final pode ser considerada bastante válida.

Devido ao aparecimento de vários obstáculos como *bugs* e a complexidade do trabalho, não foram concretizados todos os requisitos requeridos no enunciado do projeto, sendo este o foco do grupo como trabalho futuro implementar a totalidade da aplicação pedida, bem como o posterior melhoramento desta, uma vez que é uma potencial ferramenta de trabalho e que poderia ser útil para grupos de utilizadores que pretendam partilhar documentos entre si.

Em suma, o esforço foi grande com o intuito de garantir boas soluções para o enunciado proposto, deixando no grupo um sentimento de satisfação por ter cumprido grande parte dos objetivos.