

UNIVERSIDADE DO MINHO
MIEI

DESENVOLVIMENTO DE APLICAÇÕES WEB

UMbook

Grupo:

Luís Braga - a82088

Luís Martins - a82298

Luís Filipe - a83099

Braga, Portugal
28 de Janeiro de 2020

Conteúdo

1	Arquitectura Geral da Implementação	2
2	Funcionalidade Geral do Sistema	5
2.1	Segurança no registo e autenticação	5
2.2	Segurança dos servidores	5
2.3	Funcionalidades disponibilizadas ao Utilizador	6
2.4	Funcionalidades disponibilizadas ao Administrador	6
3	Documentos MongoDB	7
3.1	Utilizadores	7
3.2	Conversas	8
3.3	Mensagem	8
3.4	Grupos	9
3.5	Publicações	10
3.6	Ficheiros	10
4	Servidores Back-end	12
5	Servidores Front-end	13
6	Conclusão	14

Capítulo 1

Arquitectura Geral da Implementação

Na figura abaixo apresenta-se a arquitectura geral e atual do sistema.

A presente arquitectura categoriza-se como uma arquitectura microsserviços. Este tipo de arquitecturas possui vários apanágios, tais como: manutenibilidade, adquirida pela facilidade como são acrescentadas funcionalidades e alteradas outras já existentes; facilmente testável, por exemplo, com facilidade se rastreia por onde passam os dados e se detectam anormalidades; baixo acoplamento, que se poderá obter, por exemplo, separando os servidores que accedem aos diferentes tipos de dados e capacidade de serem accionados de forma independente, na medida em que todos os servidores podem ser ativados de forma individual.

Tendo em conta as circunstâncias e que o servidor apenas terá correr numa máquina, o grupo de trabalho optou por apenas utilizar apenas um servidor *MongoDB*.

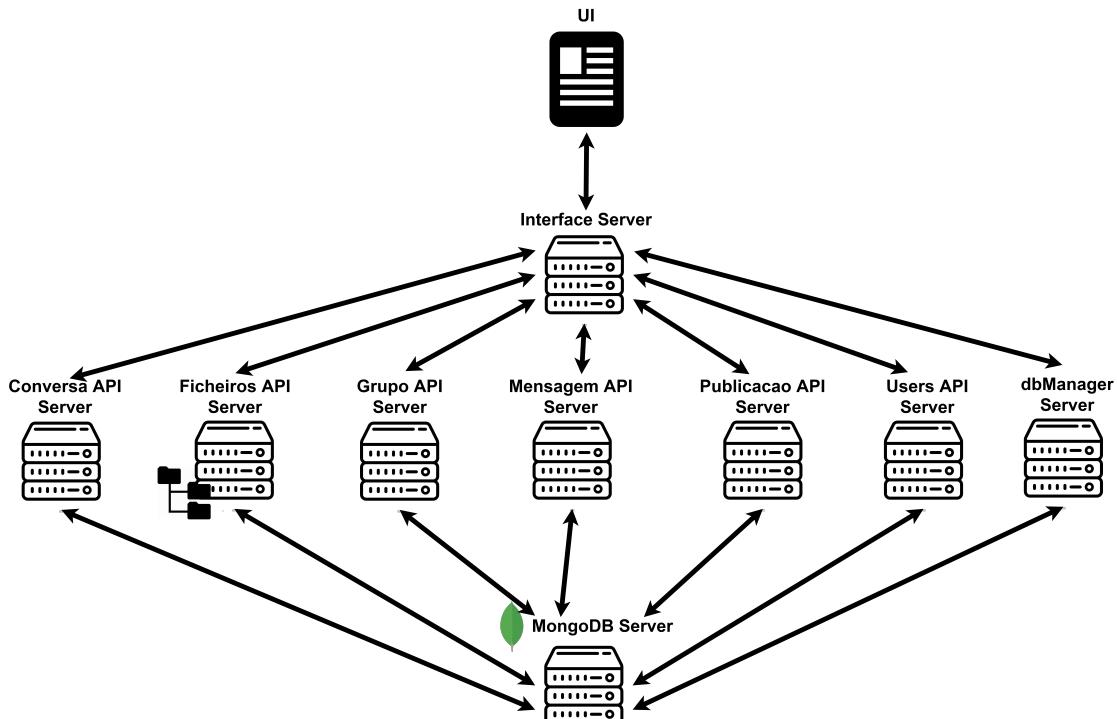


Figura 1.1: Arquitectura geral do sistema.

Como alternativa, o sistema poderia funcionar da seguinte forma, isto é, com vários servidores *MongoDB*. Em que cada um conteria uma *collections*, uma vez que são todas independentes,

apesar de existir referenciabilidade entre as mesmas.

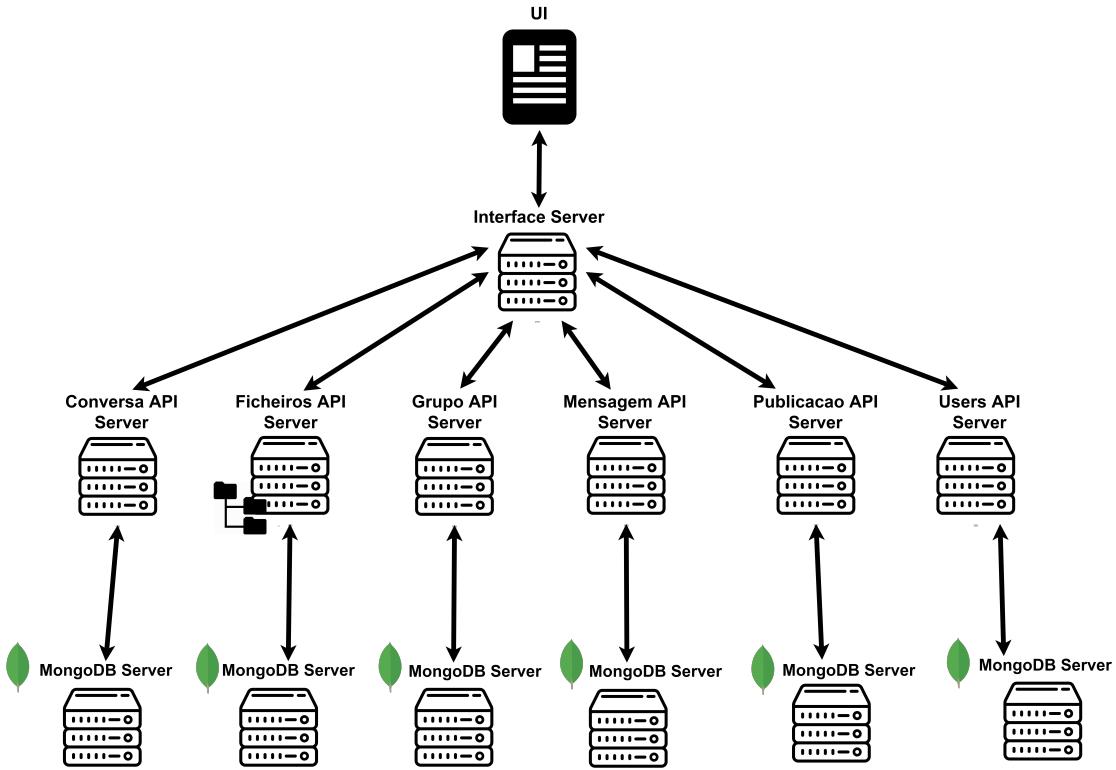


Figura 1.2: Arquitectura geral do sistema.

O sistema foi ainda pensado para evoluir no sentido apresentado na figura abaixo. Se a API fosse de dados fosse disponibilizada a outros tipos de máquina ou até utilizada para outros propósitos tal seria possível.

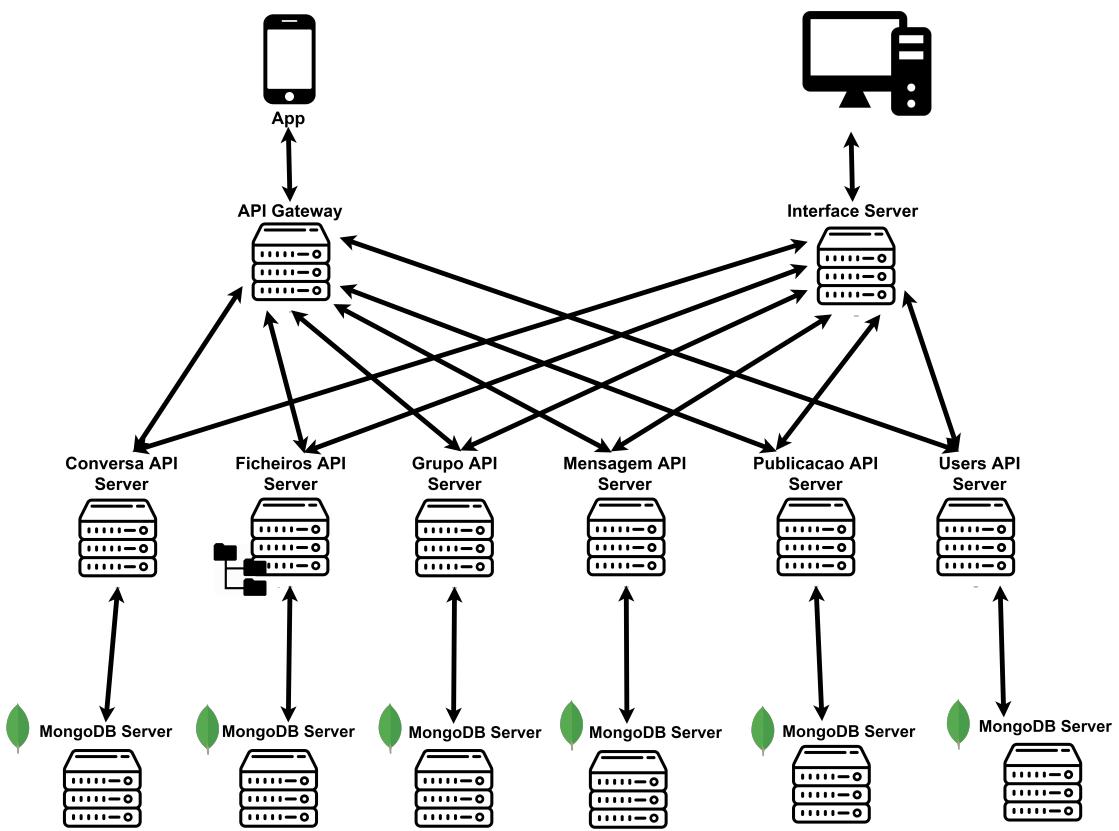


Figura 1.3: Arquitectura geral do sistema.

Capítulo 2

Funcionalidade Geral do Sistema

2.1 Segurança no registo e autenticação

De seguida serão apresentados tópicos relacionados com a segurança e a normal utilização do sistema por parte de um utilizador:

- O utilizador começa por se registar;
- Quando o utilizador submete os dados é enviada uma mensagem para o email fornecido no registo, com a finalidade de validar o email provido;
- Quando do registo é gerado um hash associado ao email;
- Só o utilizador que recebe o email tem acesso ao hash e, portanto, só ele consegue validar o email;
- Quando o utilizador acede ao link presente no email enviado, é verificado o hash associado com o endereço electrónico do utilizador;
- Caso o hash seja válido a conta é ativada, dando-lhe nível de acesso para aceder;
- Depois de validado o email o utilizador pode autenticar-se;
- Caso se autentique é gerada uma sessão associada ao utilizador;
- Essa sessão é enviada por *cookie* para evitar que o utilizador se tenha de autenticar em cada pedido que realiza;
- Em cada pedido do utilizador ao *Interface Server* são verificados os *cookies*.

2.2 Segurança dos servidores

- Para se conseguir comunicar com as APIs é necessário gerar um *JSON Web Token* no *Interface Server*;
- Em cada pedido é preciso verificar o *token* no servidor da *API*;
- Para além de se verificar o *token* ainda é verificado, no servidor da *API*, se o pedido possui nível de acesso suficiente para o pedido ser realizado.

2.3 Funcionalidades disponibilizadas ao Utilizador

Abaixo apresentam-se as funcionalidades disponibilizadas a um utilizador autenticado.

- O utilizador é capaz de pedir acesso a um grupo;
- Permitido o acesso, o utilizador poderá fazer publicações no grupo;
- Uma publicação pode ser apenas texto, apenas ficheiros ou poderá conter ambos;
- O utilizador pode apagar as suas publicações;
- O utilizador pode fazer download de ficheiros contidos em publicações;
- O utilizador pode conversar com outros utilizadores através de um chat;
- O utilizador pode alterar todos os seus dados com a excepção do curso e do email;
- Um utilizador consegue terminar a sua sessão.

2.4 Funcionalidades disponibilizadas ao Administrador

Abaixo apresentam-se as funcionalidades disponibilizadas a um administrador.

- Um administrador pode adicionar pessoas a grupos;
- Tem nível de acesso para apagar dados;
- Tem nível de acesso para criar grupos;
- Tem nível de acesso para editar grupos;
- Pode fazer *drop* das coleções;
- Pode exportar os dados das coleções;
- Pode importar os dados das coleções.

Capítulo 3

Documentos MongoDB

No presente capítulo apresentam-se os formatos dos documentos *Mongo* usados para assegurar a persistência dos dados do sistema. De forma sucinta, explica-se a que correspondem os campos de cada documento.

3.1 Utilizadores

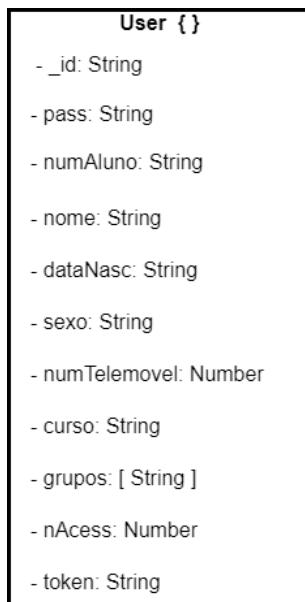


Figura 3.1: Formato do documento que guarda a informação relativa aos utilizadores do sistema.

Os utilizadores do sistema serão quem utilizará o sistema para usufruir das suas funcionalidades. Para tal, o sistema guardará os dados discriminados abaixo:

- **_id**: identificador do documento, que é igual ao email do utilizador;
- **pass**: password do utilizador;
- **numAluno**: visto que o sistema servirá como um sistema de partilhas de ficheiro entre alunos, guarda-se o número de aluno;

- **nome**: Nome do utilizador;
- **dataNasc**: Data de nascimento do utilizador;
- **sexo**: Sexo do utilizador;
- **numTelemovel**: Número de telemóvel do utilizador;
- **curso**: Curso em que o utilizador ingressou;
- **grupos**: Lista com os identificadores dos grupos a que o utilizador tem acesso;
- **nAcess**: nível de acesso do utilizador;
- **token**: hash inicial de autenticação.

3.2 Conversas

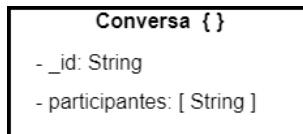


Figura 3.2: Formato do documento que guarda a informação relativa às conversas entre utilizadores.

Os utilizadores da aplicação poderão ter conversas entre si e, portanto, é necessário guardar esta informação. As conversas apenas possuem dois campos que são necessários guardar:

- **_id**: identificador da conversa;
- **participantes**: Lista com os identificadores do intervenientes da conversa.

3.3 Mensagem

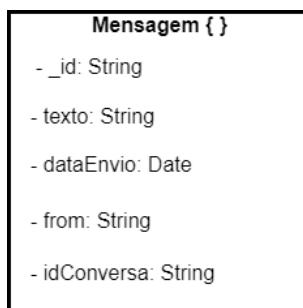


Figura 3.3: Formato do documento que guarda a informação relativa às conversas entre utilizadores.

Para se realizar uma conversa é necessário enviar mensagens. Para tal existe a seguinte estrutura em esses dados são guardados:

- **_id**: identificador único de uma mensagem;
- **texto**: conteúdo da mensagem;
- **texto**: texto da mensagem;
- **dataEnvio**: data de envio da mensagem;
- **from**: identificador de quem enviou a mensagem em questão;
- **idConversa**: identificador da conversa à qual a mensagem pertence.

3.4 Grupos

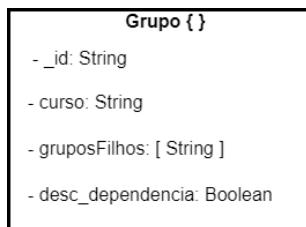


Figura 3.4: Formato do documento que guarda a informação relativa aos grupos do sistema.

O sistema possui Grupos a que um utilizador poderá ou não ter acesso. São nestes grupos que se fazem publicações e partilham ficheiros. Para isto ser possível concebeu-se a seguinte estrutura:

- **_id**: identificador de um grupo;
- **curso**: curso a que pertence determinado grupo;
- **gruposFilhos**: lista de identificadores dos grupos que este grupo contém;
- **desc_dependencia**: indica se existem outros grupos que dependem do grupo em questão.

3.5 Publicações

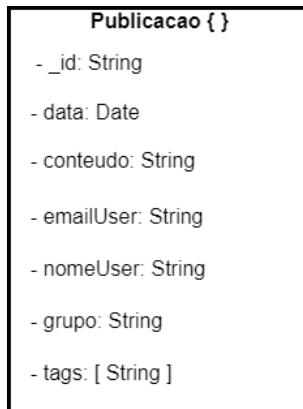


Figura 3.5: Formato do documento que guarda a informação relativa às publicações dos grupos.

De forma a partilhar os ficheiros, os utilizadores terão de criar publicações. Uma publicação tem o seguinte de formato na base de dados:

- **_id**: identificador de uma publicação;
- **data**: data em que foi feita a publicação;
- **conteudo**: conteúdo de texto da publicação;
- **emailUser**: identificador do utilizador que realizou a publicação;
- **nomeUser**: nome do utilizador que realizou a publicação;
- **grupo**: identificador do grupo ao qual a publicação pertence;
- **tags**: tags relacionadas com a publicação.

3.6 Ficheiros

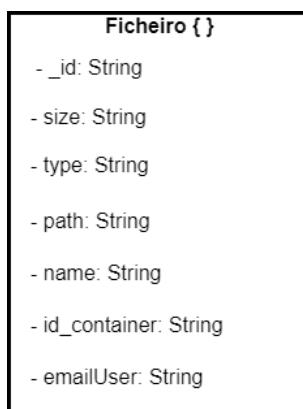


Figura 3.6: Formato do documento que guarda a informação relativa aos ficheiros partilhados em publicações.

Toda a aplicação perderia o propósito sem o seu ponto principal, isto é, a partilha de ficheiros entre alunos. Neste caso, implementou-se um sistema de ficheiros baseado em *md5* e uma coleção *Mongo* que auxilia na procura destes ficheiros. A estrutura dos documentos dessa coleção é a seguinte:

- **_id**: identificador de um ficheiro;
- **size**: tamanho do ficheiro em questão;
- **type**: tipo do ficheiro;
- **path**: caminho, no sistema de ficheiros, até ao ficheiro em questão;
- **name**: nome do ficheiro em questão;
- **id_container**: identificador da publicação onde o ficheiro foi inserido;
- **emailUser**: identificador do utilizador que inseriu o ficheiro.

Capítulo 4

Servidores Back-end

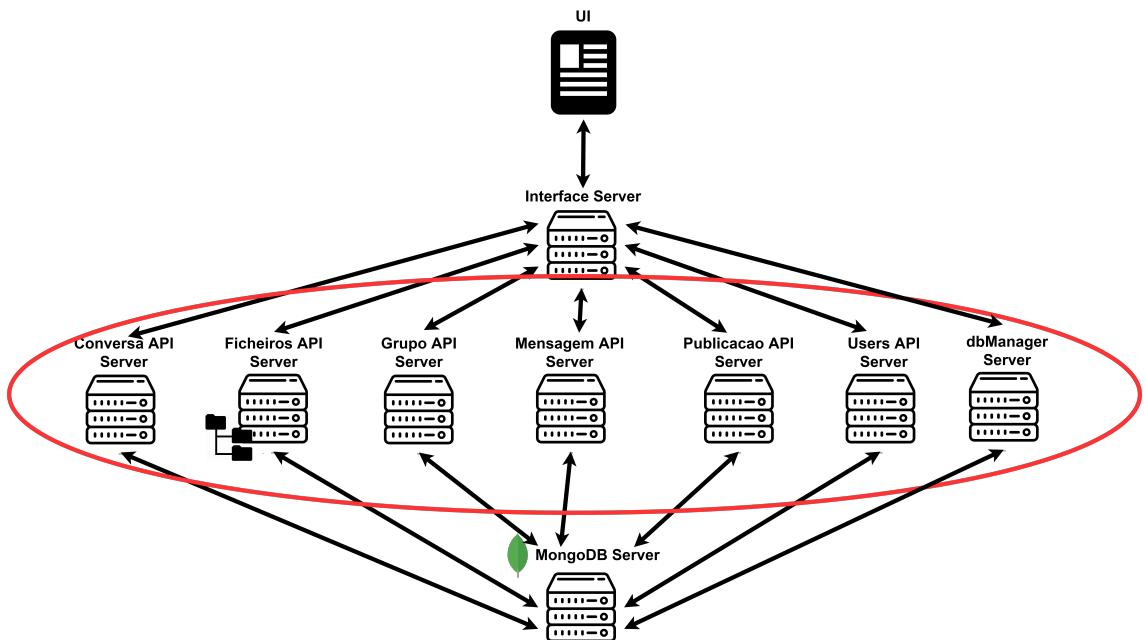


Figura 4.1: Servidores Back-end.

Os servidores *back-end* (rodeados a vermelho na arquitectura) são responsáveis pela comunicação com o servidor *MongoDB*. Por eles passam todos os dados usados pelo sistema, quer os dados requisitados quer os dados que serão guardados na base de dados. Estes servidores também estão preparados para serem futuramente disponibilizados a outros servidores sem ser apenas o servidor *Interface Server*.

Capítulo 5

Servidores Front-end

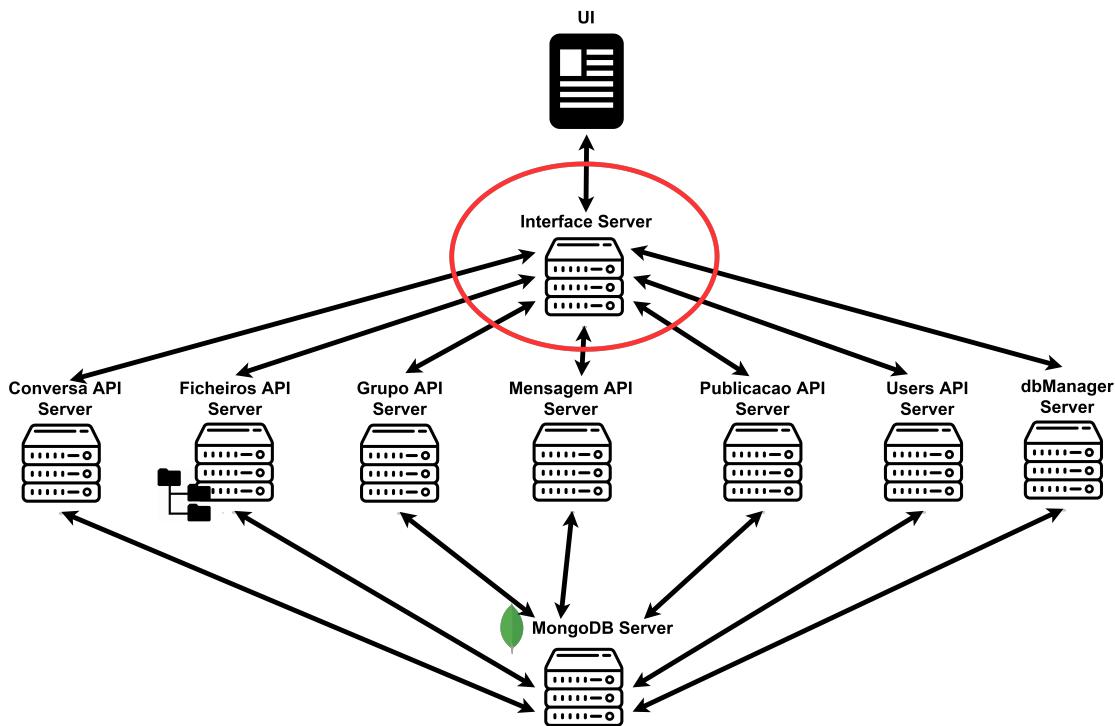


Figura 5.1: Servidores Front-end.

O servidor *front-end* (rodeado a vermelho na imagem acima) ocupa-se de fazer de a ligação entre os servidores de *back-end* e os utilizadores finais da aplicação.

Neste servidor encontram-se as rotas disponibilizadas aos utilizadores, através da interface, para realizarem ações no sistema. Essas mesmas rotas encontram-se protegidas e o utilizador apenas conseguirá utilizá-las apenas se estiver autenticado.

Capítulo 6

Conclusão

Todo o desenvolvimento deste sistema se revelou um desafio devido aos diversos *stumbling blocks* encontrados pelo caminho.

O grupo começou por definir o formato dos documentos *Mongo* e quais os dados que se teria de assegurar a persistência. Antes de uma versão final foi necessário alterar o formato dos documentos várias vezes, umas vezes acrescentando, outras vezes retirando campos. Optou-se por criar documentos simplistas, que não contêm outros documentos, mas que contêm, quando necessário, referências aos documentos necessários, quer estejam na mesma coleção quer outras. Esta *design decision* permite separar os dados por coleções, que poderão nem estar na mesma base de dados.

A seguir, o colectivo de trabalho começou por definir a arquitectura do sistema, o que não foi feito em apenas uma vez, mas sim em versões incrementais e cada vez próximas da idealização de uma arquitectura de microsserviços.

Para cada coleção existe um servidor de *API*, o que concede uma certa independência entre os dados, apesar de alguns dados referenciarem coleções diferentes da sua. Mas isto foi concebido desta forma, em larga medida, devido às diferentes cargas dos servidores, por exemplo, espera-se que o servidor de mensagens tenha mais carga de trabalho que o servidor de grupos. Desta forma, também é conferida alguma escalabilidade numérica ao sistema.

De momento, também apenas um servidor de interface, o que pode ser um *bottleneck* e *single point of failure (SPOF)*. Mas, para resolver este problema, seria possível crescer este tipo de servidores verticalmente ou horizontalmente.

Também surgiram obstáculos relativos ao armazenamento de ficheiros, pois os ficheiros não poderiam ter o mesmo nome. Facilmente se contornou este problema, utilizando uma táctica falada pelo docente em aula. Esta consiste em calcular o *md5* do conteúdo do ficheiro, dividir a *hash* resultante em quatro partes e criar um sistema de directorias, guardando na última o ficheiro. Para mais tarde encontrar o ficheiro no sistema guarda-se na base de dados o caminho até ao mesmo. Contudo, a passagem dos ficheiros pelo servidor intermédio revelou-se um verdadeiro desafio, que demorou várias horas a ser ultrapassado. Depois de superada esta etapa o grupo definiu ainda número máximo de ficheiros por upload e tamanho máximo por ficheiro ou do conjunto de ficheiros.

A próxima adversidade encontrada foi a autenticação através do *facebook*. Para tal, foi necessário ligar ao *facebook developer*, que era uma coisa nova e nunca antes feita pelo grupo. De seguida, ainda foi necessário integrar a plataforma *facebook developer* com o *passport* do *node* de modo a criar uma estratégia de autenticação através do *facebook*. Ainda foi necessário completar os dados do *facebook* com informações que o *facebook* não dispõe e que são essenciais para a aplicação.

Por fim, o último obstáculo encontrado foi aquando do pedido de acesso a grupo por parte de um utilizador. Para tal foi preciso enviar um email ao administrador e gerar nesse email um link que permitisse autenticar o administrador e quando ele se autenticasse o utilizador é automaticamente adicionado ao grupo.

Assim, a maior complicaçāo encontrada na implementaçāo do sistema foi evitar o acesso directo às *APIs*. Tendo tudo que passar pelo *Interface Server*.