Projeto 1

Sistemas Distribuídos, 2024-25

(Versão 1.4, 4 de Abril de 2025)

Prazos

1º Projeto - 26 de abril, 23h59 (online - código + relatório/formulário)

Objetivo

O objetivo do trabalho é desenvolver um sistema distribuído que funciona como uma rede social, semelhante a um fórum de discussão, com respostas, em que os utilizadores podem votar nos conteúdos disponibilizados. A inspiração para este trabalho é a plataforma Reddit (https://reddit.com).

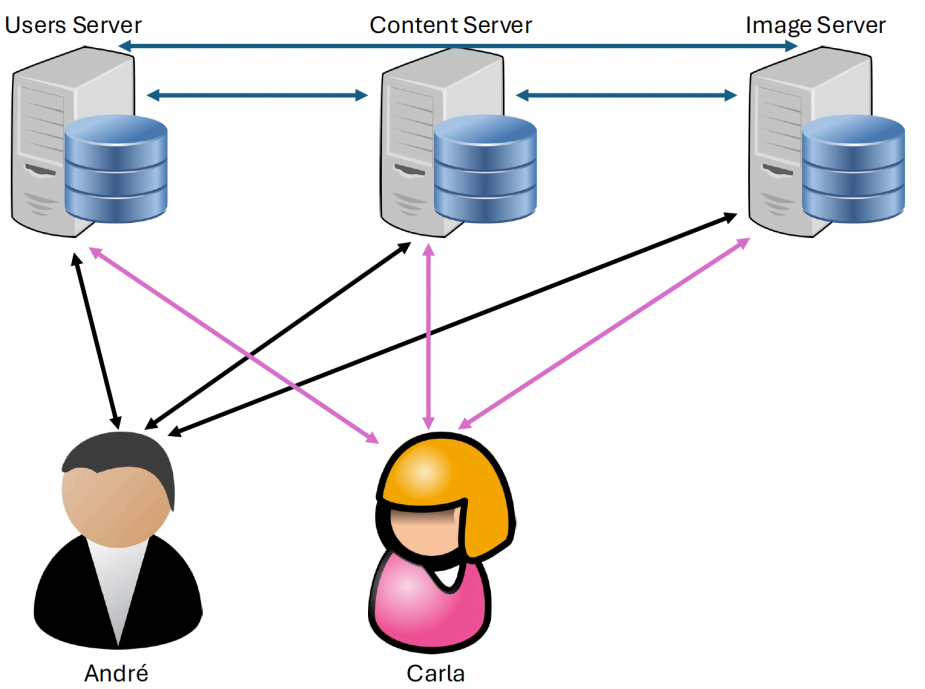
O sistema vai, por isso, ser composto por um conjunto de utilizadores registados, que podem criar (editar e apagar) entradas no fórum, emitir respostas a entradas (suas ou de outros utilizadores) assim como respostas a respostas, em que todos estas variantes de conteúdo são denominados ***Posts***, Os utilizadores podem também votar favoravelmente (***up-vote***) ou desfavoravelmente (***down-vote***) a entradas ou respostas existentes no fórum.

Adicionalmente, os utilizadores podem ter uma imagem de perfil (denominada avatar) e tanto as entradas no fórum como respostas podem (opcionalmente) incluir um elemento multimédia,

Arquitetura e Funcionalidades

O sistema será composto por um conjunto de servidores, em que cada servidor fornece um serviço específico que contribui para a funcionalidade total do sistema.

No primeiro trabalho, consideramos três tipos de servidores. A figura seguinte fornece uma visão global e de alto-nível da arquitetura da aplicação. Note-se que para fornecer toda a funcionalidade da aplicação os diferentes servidores têm de cooperar entre si.



A aplicação combina três serviços: um serviço de utilizadores (***Users***) que gere a informação de todos os utilizadores; um serviço de conteúdos (***Content***) que gere toda a informação relativa a entradas no fórum, respostas, *up-votes* e *down-votes* desses conteúdos; e, um serviço de imagens (***Images***) que é responsável por armazenar e fornecer os conteúdos multimédia (imagens), associados aos avatares dos utilizadores ou às entradas e respostas no fórum.

A gestão de utilizadores efetuada pelo serviço ***Users***, deve suportar as seguintes operações:

- Criação de um utilizador;
- Obtenção da informação associada a um utilizador existente;
- Modificação da informação associada a um utilizador;
- Remoção de um utilizador;
- Pesquisa de utilizadores.

A gestão das entradas e respostas será efetuada pelo serviço ***Content***, com recurso às seguintes operações:

- Criação de uma nova entrada no fórum ou uma resposta;
- Obtenção da lista de identificadores das entradas de topo do fórum (não incluí respostas), opcionalmente indicando uma estampilha temporal (i.e., *timestamp*), sendo que nesse caso apenas devem ser retornados os identificadores das entradas de topo criadas depois dessa estampilha temporal;

Criação de uma resposta associada a uma entrada já existente (uma entrada de topo ou uma resposta).

- Obtenção dos identificadores de todas as respostas a uma entrada;
- Obtenção do conteúdo de uma entrada no fórum (entrada de topo ou resposta);
- Remoção de uma entrada de topo ou resposta dado o seu identificador (apenas o utilizador que criou esse elemento o pode apagar);
- Adicionar um *up-vote* a uma entrada (entrada de topo ou resposta). Esta operação só pode ser feita uma vez por utilizador e apenas se este não fez anteriormente um *down-vote* nessa mesma entrada;
- Adicionar um *down-vote* a uma entrada (entrada de topo ou resposta). Esta operação só pode ser feita uma vez por utilizador e apenas se este não fez anteriormente um *up-vote* nessa mesma entrada;
- Remoção de um *up-vote* de uma entrada (entrada de topo ou resposta);
- Remoção de um *down-vote* de uma entrada (entrada de topo ou resposta);
- (Opcionalmente) obtenção da contagem de down-votes e *up-votes* de uma entrada.
- (Opcionalmente) obtenção dos identificadores de entradas de topo no fórum por ordem de entradas com mais up-votes ou entradas de topo com mais respostas (neste último caso consideram-se também as respostas a respostas das diferentes entradas).
- (Opcionalmente) leitura bloqueante para novas respostas a uma entrada de topo.

A gestão de conteúdos multimédia será assegurada pelo serviço ***Image*** que disponibiliza as seguintes operações:

- Criação de uma imagem.
- Obtenção de uma imagem.
- Remoção de uma imagem.

Semântica operacional dos serviços

Em baixo podem encontrar alguns detalhes sobre o modelo de interação dos utilizadores com alguns serviços e sobre o comportamento de algumas operações de forma a desambiguar como estas devem ser implementadas.

Serviço Users:

- A remoção de um utilizador deve ter consequências sobre o estado dos serviços *Content* e *Image*, em particular, no caso do utilizador ter uma imagem definida como Avatar, essa imagem deve ser apagada. Para além disso, todas as entradas no serviço Content criadas por esse utilizador devem perder a

referências ao utilizador que criou essa entrada (a variável authorId deve ficar a null, o que impede esse post de ser apagado ou editado a partir daí). Estes efeitos devem ser garantidos pelo servidor que executa o serviço Users através de interações com os restantes servidores.

Serviço Content:

- A edição de uma entrada no fórum apenas é possível de ser feita pelo utilizador que criou essa entrada e apenas se essa entrada não tiver outra entrada que lhe faça referência ou um upVote ou downVote, sendo que se já existirem referências à entrada deve ser retornado o erro CONFLICT.
- A edição de uma entrada no fórum nunca deve manipular os seguintes elementos dessa entrada: postId, authorId, creationTimestamp, parentUrl, upVote, e downVote.
- A remoção de uma entrada no fórum que tenha um elemento multimédia associado deve também apagar esse elemento do serviço de Image. Essa funcionalidade deve ser assegurada pelo serviço de Content através de interação com o serviço Image.

Observações Gerais:

- Em qualquer serviço, numa operação que requeira as credenciais de um utilizador, as verificações e respostas de erro devem seguir a seguinte ordem.
    a. Se a operação afeta um recurso específico e este não existir deve ser retornado o erro NOT_FOUND.
    b. Se o utilizador que tenta fazer a operação não existe deve ser retornado o erro NOT_FOUND.
    c. Se o utilizador que tenta fazer a operação é o correto mas a password está errada deve ser retornado o erro FORBIDDEN.
    d. Se o utilizador que tenta fazer a operação não é o correto (i.e., tentar apagar uma imagem criada por outro utilizador) deve ser retornado o erro FORBIDDEN independentemente da password estar ou não correta.

Interfaces dos serviços

Os servidores dos domínios do sistema serão desenvolvidos com recurso à tecnologia REST (JAX-RS) e (opcionalmente) através de gRPC.

Para garantir a interoperabilidade com a bateria de testes a disponibilizar, os serviços a implementar por cada grupo terão de respeitar as interfaces de programação pré-definidas e os efeitos esperados das suas operações (incluindo resultados e exceções).

O código da interface dos vários serviços está disponível no clip, em documentação, problemas, no ficheiro sd2425-proj-api.zip. O código tem documentação que especifica como as operações devem ser invocadas e qual o resultado da sua invocação (incluindo erros a enviar).

Este zip contém um projeto Maven, que podem (e devem) usar como base para a realização do trabalho.

Nota: As interfaces podem ser modificadas apenas introduzindo novas operações, ou através de parâmetros opcionais nas operações pré-existentes. A falha de respeito desta regra tornará o projeto incompatível com o *tester* automático fornecido.

Nota: As interfaces definidas, com os pedidos a serem efetuados sobre canais não cifrados, não são seguras. Os aspetos de segurança serão endereçados no segundo trabalho.

Auto-configuração

Os servidores deverão ser capazes de se autoconfigurar, não podendo depender de endereços IP fixos. Para tal, deverão implementar um mecanismo de descoberta baseado em anúncios periódicos (por parte dos servidores) e comunicação IP multicast.

Este mecanismo também será o meio utilizado pela bateria de testes (e por clientes) para descobrir as instâncias dos servidores presentes na rede local de forma a poderem interagir com estes.

O protocolo de descoberta consiste em enviar periodicamente para o endereço IP multicast e porto pré-acordados, uma mensagem, contendo uma String, com o seguinte formato:

`<identificador-do-serviço><tab><uri-do-servidor>`

O `<uri-do-servidor>` deve terminar com /rest para indicar, respectivamente, que se trata de um servidor REST ou /grpc para um servidor gRPC. Os identificadores do serviço são respectivamente “Users”, “Content” e “Image”.

Requisitos da solução

O foco deste primeiro trabalho será centrado nas tecnologias de Invocação Remota na Web e na Distribuição.

A solução não precisa tolerar falhas de componentes. As únicas falhas a contemplar são as falhas (temporárias) de comunicação. (Não há, assim, necessidade de introduzir replicação de componentes.)

A compatibilidade com as interfaces e operações pré-definidas tem que ser observada. No entanto, poderá ser necessário adicionar mais operações para atender a alguns dos requisitos da solução.

Requisitos mínimos (max: 9 valores)

- API REST - Servidores na sua versão REST funcionais, com todas as operações (exceto as operações marcadas acima como opcionais);
- Auto-configuração - A descoberta por multicast dos servidores funciona corretamente;

Requisitos base (max: 15 valores)

- Servidores REST funcional, com todas as operações (exceto as marcadas como opcionais acima) a executar corretamente quando existem falhas de comunicação transientes. O sistema funciona apenas com interação entre servidores REST.
- Servidores gRPC funcional, com todas as operações (exceto as marcadas como opcionais acima) a executar corretamente quando existem falhas de comunicação transientes. O sistema funciona apenas com interação entre servidores gRPC.

Elementos valorativos (max: 20 valores)

- Servidores REST e gRPC que disponibilizam o serviço de *Content* suportam corretamente a operação de obtenção da contagem de down-votes e up-votes de uma entrada (Max 1 valor adicional).
  Testes: TBD
- Solução suporta transparentemente qualquer combinação de servidores REST e gRPC na sua arquitetura (Max 2 valores adicionais).
  Testes: TBD
- Servidores REST e gRPC que disponibilizam o serviço de *Content* suportam corretamente a operação que permite a obtenção dos identificadores de entradas de topo no fórum por ordem de entradas com mais up-votes ou entradas de topo com mais respostas (Max 2 valores adicionais).
  Testes: TBD
- Servidores REST e gRPC que disponibilizam o serviço de *Content* suportam corretamente a operação de leitura bloqueante para novas respostas a uma entrada de topo (Max 3 valores adicionais).

Nota: Os elementos valorativos não serão considerados nos casos em que os requisitos base não sejam atingidos de forma satisfatória. Não é necessário realizar todos os elementos valorativos para atingir a nota máxima.

Fatores depreciativos

O código entregue deverá seguir boas práticas de programação. A repetição desnecessária de código, inclusão de constantes mágicas, o uso de estruturas de dados inadequadas, etc., poderá incorrer numa penalização. (penalização max: 2 valores)

Falta de robustez e comportamentos erráticos da solução são motivo para penalização. (penalização máxima: variável de acordo com a instabilidade observada)

Relembra-se que para os requisitos mínimos e base a solução deve contemplar as falhas temporárias dos canais de comunicação.

Execução

O trabalho pode ser executado em grupo, de 1 ou 2 alunos. Os alunos do mesmo grupo não precisam de pertencer ao mesmo turno prático, embora tal seja fortemente recomendado.

Avaliação

A avaliação do trabalho terá em conta os seguintes critérios:

- Funcionalidades desenvolvidas e a sua conformidade com a especificação, tendo como base os resultados da bateria de testes automáticos;
- Qualidade da solução;
- Qualidade do código desenvolvido.

A classificação final do aluno na componente prática é individual e será menor ou igual à classificação do trabalho, sendo estas variações em função dos resultados obtidos numa discussão individual, no caso de o aluno ser convocado para esta (falha de comparência na discussão levará a uma nota final na avaliação de zero valores). No caso do aluno não ser convocado para discussão, a nota do trabalho passará a ser a nota final do aluno na componente prática.

Bateria de testes

Existe um tester que permite verificar, através de um conjunto de testes, se as funcionalidades implementadas estão de acordo com a especificação. De forma a executar o tester os alunos devem mudar a sua identificação no ficheiro pom.xml

(variáveis xxxxx e yyyyy) e gerar uma imagem docker com o Dockerfile providenciado junto com a API do projeto (ficheiro zip no clip).

Existe no código da API do projeto fornecido, scripts .sh (Linux e Mac) e .bat (Windows) para facilitar a execução do tester. O tester recorre a valores no ficheiro fctreddit.props para obter informação relativamente a como lançar os vários serviços e como fazer a descoberta dos mesmos, devem verificar cuidadosamente a correção destes valores. A alteração deste ficheiro requer que recriem a imagem docker do projeto antes de a voltar a testar.

O tester vai evoluíndo ao longo do tempo com testes adicionais que verificam funcionalidades de forma crescente. A informação sobre o uso e os testes implementados no tester encontra-se em: https://smduarte.github.io/sd2425-tester

Recomenda-se que os alunos verifiquem esta página frequentemente. Note-se que a passagem nos testes automáticos indicam uma forte possibilidade de que a solução esteja correta, no entanto pode acontecer que uma implementação com erros passe os testes.

Ambiente de desenvolvimento

Todo o material de apoio fornecido pressupõe que o desenvolvimento será em ambiente Linux e Java 17. A validação do trabalho por via da bateria de testes automática fará uso da tecnologia Docker.

Prazos e Entrega

26 de abril, 23h59 (online - código + relatório/formulário)

Para submeter o trabalho, deve criar um ficheiro comprimido com o formato zip (formatos de compressão alternativos não serão aceites), contendo: os ficheiros fctreddit.props, pom.xml, Dockerfile e a diretoria src com o código do vosso trabalho. Se fizerem *mvn clean* e, a seguir, comprimir a diretoria do trabalho, terão o que se pretende (devem fazer este passo sem ter o IDE ativo). O ficheiro deve ter o nome sd-trab1-xxxxx-yyyyy.zip para os alunos com os números xxxxx e yyyyy, em que xxxxx < yyyyy. (ou sd-trab1-xxxxx.zip para os alunos que fizeram o trabalho sozinhos, substituindo o xxxxx pelo número do aluno que realizou o trabalho).

Histórico de alterações

17 Março 2025

- Primeiro draft do enunciado.

23 Março 2025

- Revisão interna Disponibilização da primeira versão do enunciado.

24 Março 2025

- Atualização do diagrama de interações entre serviços.
- Especificação de efeitos adicionais em algumas operações (secção “Semântica operacional dos serviços”)

26 Março 2025

- Atualização nas APIs realizadas no código fornecido (código atualizado no Clip).
    - Clarificação das condições para retornar certas operações de erro.
    - Correção dos tipos de operações para manipular upvotes/downvotes no serviço Content
    - Atualização da especificação de mensagens no gRPC.

27 Março 2025

- Correção de um erro na API do serviço de Image (código atualizado no Clip).

4 Abril 2025

- Disponibilização da versão zero do tester (verifica a coerência da imagem docker produzida, do ambiente de execução e a descoberta automática do servidor de Users com API REST)
- Modificações no API fornecida (sd2425-proj-api.zip fornecido no clip):
    - Modificação do ficheiro .props
    - Modificação dos scripts para execução do tester (.sh e .bat)
    - Modificação do ficheiro Dockerfile para compatibilidade com o tester
    - Modificações nas interfaces (Java, REST, e gRPC) do serviço Content, de forma a fornecer um parâmetro opcional na operação que consulta as respostas a um post, para esta ter um comportamento bloqueante - de forma a suportar a última opção valorativa).