Sistemas Distribuídos

Ano letivo 2019/2020

Relatório
Projeto de Sistemas Distribuídos
Meta 2

FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Trabalho Realizado por:
Alexandre Maria Martins Magalhães Teixeira Serra – 2017248031
João Gabriel de Matos Fernandes – 2017247486

Departamento de Engenharia Informática
Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Arquitetura de software

Models

HeyBean:

Este foi o único model que usámos. Aqui guardamos a referência para o servidor RMI, o username, o nome (caso tenho conta do Facebook associada), o tipo de cliente que se trata (anónimo, admin ou normal) e o número de cliente atribuído.

Views

index.jsp:

Esta é a nossa página principal. O utilizador quando abre o Website vai para esta página. A partir daqui pode aceder a todas as outras funcionalidades disponíveis ao seu tipo de cliente.

register.jsp:

Página onde o utilizador se regista na plataforma, inserido um username e uma password. Poderá também registar-se com o Facebook.

login.jsp:

Página onde o utilizador faz o login na plataforma, inserido o seu username e a sua password. Poderá também entrar com o Facebook.

searchResults.jsp:

Página com o resultados de uma pequisa efetuada por um cliente. Os resultados apresentam o título, o url, uma citação e a linguagem dessa página. É possível ainda traduzir o título e a citação de um resultado específico e partilhar a pesquisa no Facebook (se o utilizador tiver conta de Facebook associada).

searchHistory.jsp:

Página com o histórico de pesquisas do utilizador com o login feito.

linksPointing.jsp:

Página que apresenta os URL's que apontam para o URL inserido pelo utilizador.

adminPrivileges.jsp:

Página onde um administrador pode promover outro utilizador a administrador.

indexNewUrl.jsp:

Página onde um administrador pode index um novo URL.

rts.jsp:

Página onde um administrador pode ver as 10 pesquisas mais frequentes, os 10 URL's com mais links a apontar para ele e os Multicast Servers ativos e respetivos endereços e portos.

facebookLogin.jsp:

Página que dá redirect para um URL. É usada para dar redirect para os URL's de login e partilha do Facebook.

exchangeTokenForCode.jsp:

Página que dá redirect para a action que vai permitir trocar o código atribuído ao utilizador pelo access token.

redirectSearch.jsp:

Página que dá redirect para a action de pesquisa. Um utilizador vai para esta página quando clica num link partilhado no Facebook.

error.jsp:

Página para onde o utilizador vai quando ocorre um erro. A partir desta página, o utilizador pode voltar ao index.jsp.

Controllers

AuthAction:

Controller usado para efeitos de login ou registo sem ser por Facebook. É passado como parâmetro no struts2 o tipo de ação (se é login ou registo).

LogoutAction:

Controller usado para efetuar o logout de um utilizador.

SearchAction:

Controller usado para obter os resultados de uma pesquisa. É passado como parâmetro no struts se se trata uma pesquisa normal ou de um link partilhado no

facebook. Se for um link partilhado no Facebook é necessário ir buscar a procura partilhada no URL.

TranslateAction:

Controller usado para traduzir o título e a citação de um resultado específico de uma procura.

SearchHistoryAction:

Controller usado para obter o histórico do utilizador.

LinksPointingAction:

Controller usado para obter os URL's que apontem para um certo URL dado pelo utilizador.

AdminPrivilegesAction:

Controller usado para promover um utilizador a administrador.

IndexNewUrlAction:

Controller usado para iniciar a indexação de um novo URL.

RtsAction:

Controller usado para apresentar estatísticas de tempo real, quando esta página é aberta.

FacebookLoginAction:

Controller usado para efetual o login com o Facebook.

ExchangeAction:

Controller usado para trocar o código fornecido pela API do Facebook por um access token e, posteriormente, fazer o login na plataforma.

FacebookShareAction:

Controller usado para obter o URL para partilhar no Facebook.

Integração do Struts2 com o Servidor RMI

Para efetuar a comunicação entre o browser recorremos ao Struts2, que é responsável por chamar actions. A referência do RMI server é guardada no Model "HeyBean". Quando uma action precisa de fazer uma chamada RMI, vai buscar essa referência e faz uma chamada RMI ao servidor, tal como era feito na 1ª meta.

Programação de WebSockets e sua integração com o servidor RMI

De maneira a possibilitar as estatísticas em tempo real (RTS, como é chamado no projeto) assim como as notificações de promoção a administrador, foram utilizados WebSockets.

A classe WebSocket implementa a interface client, a mesma utilizada pelos clientes RMI. Ao fazer isto, o nosso utilizador do browser passa a ser tratado pelo servidor RMI da mesma forma que são tratados os clientes RMI (com interface no terminal).

Deste modo apenas temos de tratar de enviar a informação correta para a interface do utilizador (tratada pelo websockets.js). Este ficheiro é incluído em todas as páginas necessárias, uma vez que não era possível manter uma ligação por WebSockets entre páginas através da nossa implementação (seria possível através de iframes, por exemplo). Dessa forma, sempre que entramos/saímos de uma página iniciamos/fechamos um websocket.

A classe GetHttpSessionConfigurator permite obter a sessão HTTP nos websockets, algo que não é diretamente acessível. Isto mostrou-se necessário uma vez que era necessária a referência para a ligação com o servidor RMI.

Notificação de administrador

Como o servidor RMI tem a referência de todos os clientes(web ou RMI), sempre que existe uma notificação de administrador esta é tratada por ele de forma igual.
Uma vez chamado o método notification da classe websocket, é enviada uma mensagem para o Javascript que este trata de mostrar em forma de *toast*.

Estatísticas em tempo real

Quando um administrador entra na página de estatísticas em tempo real, é chamada uma action (RtsAction) que se encarrega de 'inicializar os dados'. Após isto, o código javascript envia uma mensagem ao websocket a 'avisar' que o utilizador se encontra nas estatísticas em tempo real, e que as pode receber.
Sempre que há atualizações, o servidor RMI chama um método do web server (tal como foi feito na Meta1 mas com clientes RMI), passando este a redirecionar a mensagem para o código javascript, que atualiza através de Jquery a informação.

Integração com o serviço REST

Facebook API

De forma a integrar o login com o Facebook com o nosso Website, foi criada uma app no “Facebook for developers”, no qual nos foi dado um App ID e uma App Secret. Estes vão ser usados para ser feito o login no website ou associar a conta do Facebook a uma conta já registada na plataforma.

Login/Registo com o Facebook:

1. Obter o “Authorization URL”:
    a. Após clicar em “Login with Facebook” é chamada a action “FacebookLoginAction”
    b. Neste controller, é feita uma chamada RMI.
    c. O servidor RMI devolve o URL necessário. Este URL contém um callback que redireciona o client para a view “exchangeTokenForCode.jsp”, após este concluir o passo 3.
2. A action redireciona o client para a view “facebookLogin.jsp” que contém um redirect para o “Authorzation URL”.
3. O utilizador introduz os seus dados do facebook.
4. É redirecionado para a view “exchangeTokenForCode.jsp”
5. Este view chama a action “ExchangeAction”
6. Obtém-se o código fornecido pela API do Facebook a partir do URL da view “exchangeTokenForCode.jsp”
7. É feita uma chamda RMI
    a. O servidor RMI vai proceder à troca do código pelo access Token.
    b. Com este, faz um pedido à API do Facebook para obter o ID e o nome do utilizador.
    c. Faz um pedido Multicast ao servidor Multicast, que irá fazer as devidas verificações das informações recebidas.
    d. O servidor RMI devolve um JSON ao controller com o nome e ID do utilizador no Facebook e a resposta do servidor Multicast.
8. No controller, se o login for válido, é guardado na session e no model o tipo de utilizador, o ID e o nome.

Associar conta com o Facebook:

O processo desta funcionalidade é muito semelhante ao do login. A única diferença é que a mensagem enviada ao Multicast Server é diferente, sendo que este vai verificar se a conta que está a ser associada já está associada a outra conta. Se não estiver, o nome do utilizador e o seu ID no Facebook são guardados no Multicast Server.

Partilhar pesquisa no Facebook:

1. O utilizador clica em “Share on Facebook”
2. É chamada a ação “FacebookShareAction”
    a. Gera o link que vai ser partilhado no Facebook
    b. Este link contém a view “redirectSearch.jsp” e a procura que foi feita e está a ser partilhada.

    c. Gera também o “Authorization URL”, para onde o utilizador que está a fazer partilha é redirecionado.
    d. De notar que o “Redirect URL” é igual ao link partilhado, de forma a que o utilizador, após efetuar a partilha, seja redirecionado para a mesma página.
3. O resultado da action manda o utilizador para a view “facebookLogin.jsp” que redireciona o client para o “Authorization URL”
4. Aqui o client efetua a partilha no facebook e é redirecionado para a mesma página.

Yandex API

Tal como na API do Facebook, para o Yandex também foi necessário criar uma App que nos forneceu uma API key.

Traduzir título e citação:

1. Utilizador clica no resultado que quer traduzir
2. É chamada a action “TranslateAction”
3. É feita uma chamada RMI
4. O RMI Server conecta-se à API do Yandex e faz um pedido para traduzir o texto (este processo repete-se 2 vezes: 1 vez para o título e outra para a citação)
5. É devolvido o texto traduzido e colocado na página no lugar do anterior.

Detetar linguagem de página:

1. Durante a indexação, o Multicast Server envia uma mensagem por Multicast ao RMI Server com as 100 primeiras palavras de uma página.
2. O Multicast Server fica à espera da resposta.
3. O RMI Server conecta-se à API do Yandex e efetua um pedido para detetar a linguagem do texto.
4. O RMI Server envia uma mensagem por Multicast ao Multicast Server.
5. Este recebe a mensagem, guarda a informação na classe URL e continua a indexação

Testes feitos à plataforma

Requisitos Funcionais	
Registar novo utilizador	✓
Acesso protegido com password (todas as páginas exceto pesquisas)	✓
Indexar novo URL introduzido por administrador	✓
Indexar iterativamente ou recursivamente todos os URLs encontrados	✓
Pesquisar páginas que contenham um conjunto de palavras	✓

Resultados ordenados por número de ligações para cada página	✓
Consultar lista de páginas com ligações para uma página específica	✓
Consultar lista de pesquisas feitas pelo próprio utilizador	✓
Dar privilégios de administrador a um utilizador	✓
Entrega posterior de notificações (offline users)	✓
WebSockets	
Notificação imediata de privilégios de administrador (online users)	✓
Página de administração atualizada em tempo real	✓
Atualização imediata da lista de servidores multicast ativos	✓
REST	
Associar conta de utilizador ao Facebook	✓
Partilha da página com o resultado de uma pesquisa no Facebook	✓
Mostrar em cada resultado a língua original da página	✓
Traduzir título e descrição das páginas para Português	✓
Registo com a conta do Facebook (sem conta ucBusca)	✓
Extra	
Utilização de HTTPS	✓
Utilização em smartphone ou tablet	✓