Sistemas Distribuídos

Trabalho Prático Reserva de voos

Rui Armada A90468
João Carvalho A93166
Leonardo Freitas A93281
Guilherme Gonçalves A88280

Engenharia Informática
Universidade do Minho
2021/2022

1 Introdução

Neste projeto pede-se a implementação de uma plataforma de reserva de voos sob a forma de um par cliente-servidor em Java utilizando sockets e threads. A essência do serviço é permitir aos utilizadores reservar viagens constituídas por vários voos, sendo informados pelo servidor quando a reserva foi feita com sucesso. Uma viagem reservada pode ser cancelada antes do encerramento do dia correspondente. Para simplificar, não são tidos em conta os horários dos voos, assumindo que é sempre possível fazer escalas (se existe um voo A→B e outro B→C num dado dia, é possível um passageiro fazer A→B→C se houver disponibilidade de lugares em ambos os voos). Assume-se que os mesmos voos se repetem todos os dias.

2 Descrição do Problema

2.1 Funcionalidade Básica

Este serviço deverá suportar a seguinte funcionalidade básica:

1. Autenticação e registo de utilizador, dado o seu nome e palavra-passe. Sempre que um utilizador desejar interagir com o serviço deverá estabelecer uma conexão e ser autenticado pelo servidor.
2. Autenticação de um utilizador especial de administração (admin).
3. Inserção de informação sobre voos (origem, destino, capacidade) pelo administrador.
4. Encerramento de um dia por parte do administrador, impedindo novas reservas para esse dia e cancelamento de reservas desse dia. 1
5. Reserva de uma viagem indicando o percurso completo com todas as escalas (por exemplo Porto → London → Tokyo, ou seja, de Porto a Tokyo com uma escala em London) e um intervalo de datas possíveis, deixando ao serviço a escolha de uma data em que a viagem seja possível. O servidor deverá responder com o código de reserva.
6. Cancelamento da reserva de uma viagem, usando o código de reserva, a pedido do utilizador a que pertence.
7. Obtenção da lista de todas os voos existentes (lista de pares origem→destino), a pedido do utilizador.

3 Arquitetura

3.1 Classes

3.1.1 Accounts

Esta Classe é responsável pelo tratamento de todas as contas presentes no sistema que, no nosso caso, faz um mapa Treemap onde a chave é o endereço de email da conta e o valor é um User.

3.1.2 Client

Esta Classe é responsável por toda a interação do user com o sistema, como o tratamento de todas as entradas do user, formatando e enviando para o Servidor.

3.1.3 Connect

Esta Classe é responsável pelo tratamento das conexões na aplicação, ou seja, lida com o fluxo dos frames do aplicativo.

3.1.4 Demultiplexer

Esta Classe é responsável pelo tratamento da comunicação entre cliente e servidor.

3.1.5 Flight

Esta Classe é responsável por todas as informações de voo que estão presentes no sistema e, no nosso caso, um voo é composto por um id, local inicial, local final e capacidade do voo.

3.1.6 Flights

Esta Classe é responsável pelo tratamento de todos os voos presentes no sistema que, no nosso caso, faz um mapa Treemap onde a chave é o sgID da reserva e o valor é um Voo.

3.1.7 Frames

Esta Classe contém uma mensagem com informações adicionais, onde, usando um frame, podemos enviar mensagens que contêm informações sobre o tipo de mensagem e o usuário que a enviou.

3.1.8 Reservation

Esta Classe é responsável por todas as informações da reserva que estão presentes no sistema que, no nosso caso, é composta por um endereço de e-mail do Usuário, uma lista de números inteiros contendo ids de voos e a data do voo da reserva.

3.1.9 Reservations

Esta Classe é responsável pelo tratamento de todas as reservas presentes no sistema que, no nosso caso, faz um mapa Treemap onde a chave é o id da reserva e o valor é uma Reserva.

3.1.10 Server

Esta Classe contém todas as informações armazenadas no sistema, e também é responsável pela parte lógica da aplicação. O server é responsável pelo tratamento das consultas feitas pelo cliente.

3.1.11 User

Esta Classe é responsável por todas as informações do usuário que estão presentes no sistema que, no nosso caso, é composto por um email, um nome de usuário, a senha do usuário e o sinalizador admin que indica se um usuário é um admin ou não.

4 Conclusão

Ao longo deste relatório, fomos explicando todo o processo de criação e desenvolvimento do projeto da UC de Sistemas Distribuídos. Consideramos que este trabalho nos permitiu consolidar os conhecimentos obtidos nas aulas de uma forma útil e interessante.

Para a realização deste trabalho foi necessário usar programação com sockets uma vez que o propósito de toda a aplicação era que os clientes pudessem enviar pedidos ao servidor. Utilizou-se também controlo de concorrência de forma que os diferentes users tivessem acesso em simultâneo sem que isso implicasse erros ou incoerências de dados.

O grupo considera que o trabalho prático terá sido realizado com sucesso, sendo todas as queries básicas implementadas, no entanto, gostaríamos de ter tido tempo para implementar as funcionalidades adicionais pedidas, apesar de isso não ter acontecido consideramos que temos um bom trabalho.