Gestão de trotinetes elétricas

Trabalho realizado por:

Beatriz Ribeiro Monteiro
João Carlos Fernandes Novais
João Pedro Machado Ribeiro
Telmo José Pereira Maciel

A95437
Beatriz Monteiro

A96626
João Novais

A95719
João Ribeiro

A96569
Telmo Maciel

Sistemas Distribuídos 2022/2023
Universidade do Minho

Índice

1 Introdução

Este trabalho prático foi realizado no âmbito da Unidade Curricular de Sistemas Distribuídos e teve como objetivo o desenvolvimento e implementação de uma plataforma de gestão de uma frota de trotinetes elétricas, que permitisse aos utilizadores reservar e estacionar as trotinetes em diferentes locais, podendo ou não ser recompensados tendo em conta o local onde as estacionavam.

Este sistema funcionará sob a forma de cliente/servidor em Java recorrendo a *sockets* e *threads*, em que um cliente, pré-registado ou não, se pode autenticar e enviar pedidos para o Servidor que os processará, os diferentes pedidos que podem ser enviados serão abordados posteriormente neste relatório.

2 Descrição das classes implementadas

2.1 Relacionadas com o Cliente

Cliente Esta classe representa o executável que qualquer utilizador irá utilizar para conseguir aceder às diferentes funcionalidades da nossa aplicação.

Efetua a ligação do cliente ao servidor e dá início à execução de duas *threads*: uma para o ClienteMenu e outra para o ClienteReader.

ClienteMenu Classe responsável pela implementação da interface que é apresentada ao utilizador. É, ainda, responsável pela construção e o envio das mensagens que correspondem ao *input* do utilizador ao servidor. Nesta classe são geradas as mensagens geradas a partir das escolhas do utilizador que são enviadas para o servidor. Através do método *server-request* o cliente fica bloqueado à espera de resposta.

ClienteReader Esta é a classe que vai ficar responsável por ler toda a informação proveniente do servidor, imprimindo-a para o *stdout*.

ClienteStatus Por fim, esta classe guarda toda a informação sobre o estado do utilizador, entre estar logado, terminar sessão, esperar por resposta, entre outras.

2.2 Relacionas com o Servidor

Servidor A classe *Servidor* está encarregue de receber as conexões dos vários clientes. Esta classe inicializa os sockets para a comunicação e fica ativamente à espera de conexões da parte do cliente.

Coord Esta classe é responsável apenas por representar uma coordenada no mapa.

Utilizador Esta classe tem funções relativas ao utilizador (obter o nome de utilizador, obter a *password*, adicionar e remover coordenadas à sua lista de posições a notificar).

ListaUtilizadores Esta classe é responsável por guardar todos os dados relativamente aos utilizadores que se vão registando na nossa aplicação, garantindo os princípios de exclusão mútua, e é implementada como uma classe *singleton*.

Mapa Classe responsável por gerar o nosso mapa e fazer todas as alterações ao mesmo ao longo da execução da aplicação, garantindo os princípios de exclusão mútua.

Contador Classe auxiliar das classes *TrabalhadorRecompensas* e *GestorRecompensas*, possui funções relativas a um simples contador, e é implementado como uma classe *singleton*.

Reserva Classe responsável pela funcionalidade de reserva de uma trotinete.

GestorReservas Classe responsável por gerir as várias reservas.

LockReservas Classe que funciona como um lock para as reservas, e é implementado como uma classe *singleton*.

GereMensagem Classe responsável por receber as mensagens do cliente e processá-las. As mensagens recebidas podem ter as seguintes assinaturas:

- REGISTAR
- LOGIN
- LOGOUT
- RESERVAR
- LISTARTROTINETES
- LISTARRECOMPENSAS
- ESTACIONAR
- NOTIFICAR

Recompensa Classe para representar os caminhos aos quais uma recompensa é associada.

TrabalhadorRecompensas Classe responsável por gerir a lista de recompensas, ou seja, verificar as zonas do mapa que estão com pouca abundância de trotinetes e gerar recompensas para quem levar trotinetes de outra zona para esta.

ListaRecompensas Classe que armazena todas as recompensas que estejam ativas.

TrabalhadorNotificacoes Classe responsável por notificar um utilizador caso estejam recompensas disnponíveis.

3 Comunicação cliente-servidor

De modo a que todos os requisitos pedidos no enunciado do trabalho prático, tanto o nosso servidor como o cliente foram implementados via *sockets* (TCP). A classe ClienteMenu atua como o cliente, ao inicializar o programa é criado um *socket* que é passado para a classe. Esta irá lidar com os *inputs* do utilizador e permitir que este comunique com o servidor para realizar todas as operações. Ainda no cliente, a classe ClienteReader funciona da mesma forma, é passada para a mesma o mesmo *socket* para esta se conectar ao cliente e receber todas as mensagens que este devolve e apresentar no *stdout* para o utilizador. Do lado do servidor também é criado um ServerSocket que é passado à classe GereMensagem, esta vai ser responsável por receber todas as mensagens do cliente e processá-las, tal como foi dito anteriormente.

4 Controlo de concorrência

De forma a garantir o controlo de concorrência utilizamos *locks* (ReentrantLocks) e *conditions* . Estes *locks* foram colocados nos métodos em que é necessário garantir que não estão a ser acedidos por várias *threads* ao mesmo tempo, como por exemplo, a reserva de trotinetes, o seu estacionamento, a listagem de trotinetes livres, entre outros. Desta forma conseguimos garantir que todo o nosso programa corre normalmente sem qualquer conflito entre as operações das *threads*.

5 Descrição das funcionalidade da aplicação

5.1 Registar

Ao efetuar um registo, o utilizador terá de introduzir as suas credenciais (*username* e *password*, podendo a *password* ser vazia). Estas serão enviadas para o servidor que devolverá uma mensagem de sucesso, caso assim o seja.

5.2 Log In

No processo de *login* o utilizador fornece as suas credenciais que serão passadas ao servidor, que irá analisar as mesmas e, caso estas estejam corretas, devolve a mensagem de sucesso e o utilizador entra no menu de funcionalidades. Caso contrário, transmite uma mensagem de erro e volta ao menu inicial.

5.3 Logout

Logout simplesmente termina a sessão do utilizador, retrocedendo ao menu de início de sessão.

5.4 Fazer reserva

Nesta funcionalidade, o utilizador vai indicar as suas coordenadas e o sistema efetua a reserva da sua trotinete e indica o seu número de reserva. Caso não exista nenhuma trotinete nesse local o sistema vai selecionar a trotinete mais próxima num raio de 2 unidades.

5.5 Listar trotinetes livres

Ao pedir para listar as trotinetes livres o utilizador indica a sua posição e o sistema devolve todas as trotinetes disponíveis num raio de 2 unidades.

5.6 Listar recompensas

Nesta funcionalidade o utilizador indica as suas coordenadas e o sistema devolve uma lista das recompensas disponíveis naquela área, caso existam.

5.7 Estacionar trotinete

Para efetuar o estacionamento de uma trotinete o utilizador indica a posição em que vai deixar a trotinete e o seu código de reserva e o sistema informa quanto é que foi o custo da reserva em relação ao tempo da reserva e distância percorrida.

5.8 Notificar recompensas

Ao selecionar esta funcionalidade, o utilizador ativa/desativa o pedido de notificações sempre que houver recompensas disponíveis.

6 Interface Gráfica

Temos uma interface gráfica simples, uma vez que o propósito deste projeto era ser utilizado no terminal. Tal como foi pedido no enunciado foi desenvolvida na linguagem de programação Java.

Figure 1: Menu de Início de Sessão

São apresentados dois menus distintos. Um inicial que permite o utilizador fazer o seu Registo e em seguida o Login que o levará para o segundo menu.

Neste segundo menu o utilizador pode testar as diferentes funcionalidades que implementamos no nosso trabalho.

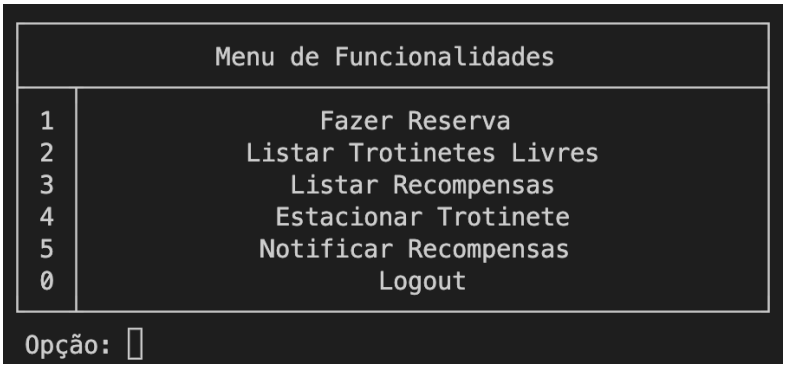

Figure 2: Menu de Funcionalidades

7 Conclusão

Durante a realização deste projeto conseguimos aplicar e aprofundar os conhecimentos que adquirimos nas aulas de Sistemas Distribuídos relativos ao controlo de concorrência e à comunicação entre Cliente e Servidor, recorrendo a *sockets* TCP.

A realização deste projeto permitiu-nos aplicar os conhecimentos que adquirimos nas aulas a um problema que pode ser facilmente transportado para o mundo real. Conseguimos ver com mais lucidez como esses conhecimetos se encaixam num mundo com uma escala muito superior à que estavamos habituados nos guiões práticos.

Por fim, face ao projeto desenvolvido, achamos que o mesmo foi bem conseguido e avaliamos positivamente o nosso desempenho, cumprindo todos os requisitos pedidos pela equipa docente.