Relatório sistemas distribuidos

6-1-2019

José Crisanto 8160169 / Cesar Lobo 8150399

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

Sumário

[Introdução 2](#_Toc534531362)

[Funcionamento do programa 2](#_Toc534531363)

[Funcionamento do protocolo 3](#_Toc534531364)

Introdução

Neste trabalho prático pretende-se que implemente um sistema similar ao conhecido Uber. A ideia é ser capaz de chamar uma viatura com condutor para uma deslocação do ponto A a B.

De seguida descreve-se o que se pretende para cada um dos componentes cliente e servidor.

## Cliente (consumidor final)

O cliente consumidor final começa por se registar no serviço. Após este registo passa a ter a possibilidade de efetuar pedidos de viaturas com condutor. A um cliente deverá ser permitido:

* Registar;
* Logar;
* Inserir origem da viagem;
* Inserir destino da viagem;
* Solicitar uma viatura com condutor para uma viagem específica;
* Atribuir uma pontuação (1 a 5) ao condutor para uma viagem específica;
* Visualizar o seu histórico de viagens e respetiva pontuação atribuída;
* Sair.

## Cliente (condutor)

O cliente condutor começa por se registar no serviço. Após este registo passa a ter a possibilidade de aceitar pedidos de viaturas com condutor caso esteja disponível para o efeito. A um cliente deverá ser permitido:

* Registar;
* Logar;
* Alterar estado (disponível / indisponível)
* Receber pedidos de viatura com condutor para uma viagem específica e aceitar ou rejeitar os mesmos;
* Visualizar o seu histórico de viagens e respetiva pontuação recebida;
* Sair.

## Servidor

O servidor deve estar preparado para receber ligações de vários clientes em simultâneo. Deverá ainda ser capaz de guardar o registo de todas as informações relevantes (clientes, viagens, pontuações, etc).

Todas os pedidos de viagens do servidor para os clientes condutor devem ser efetuados via *multicast*.

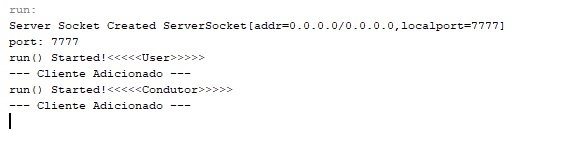
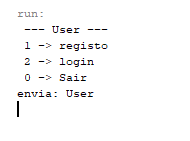
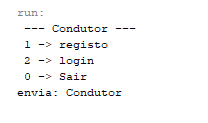
## Detalhes

Deve ser dada especial atenção aos seguintes detalhes que deverão estar cuidadosamente implementados na solução:

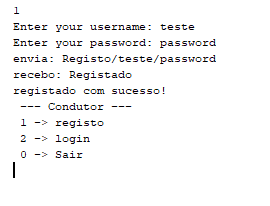
* Existência de um protocolo de comunicação ao nível da aplicação claro e documentado.
* Suporte para múltiplas *threads* e controlo sempre que exista acesso a recursos partilhados. O trabalho deve incluir um relatório explicando sucintamente a implementação e justificando quaisquer opções tomadas.

Funcionamento do programa

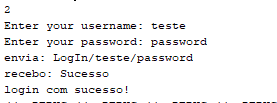
O projecto esta dividido em 3 programas, o servidor, o condutor e o utilizador

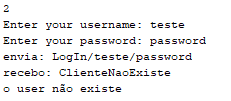


Registar um cliente é igual tanto pata o user como o condutor

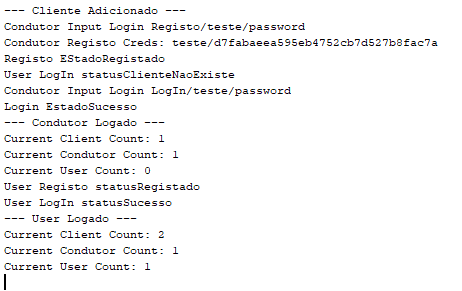


Mas os utilizadores e condutores tem contas diferentes para cada uma das aplicações

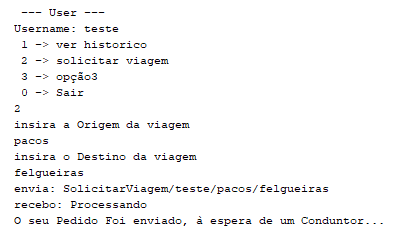
login condutor

login utilizador

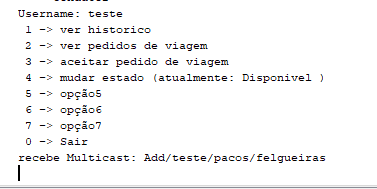
No servidor é registado os logs de tudo o que se passa



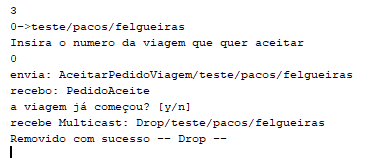
Pedido de viagem



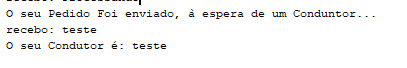
Lado do condutor quando recebe o pedido



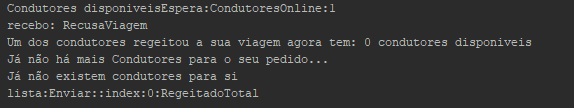
Aceitar a viagem



No lado do condutor é aceitado



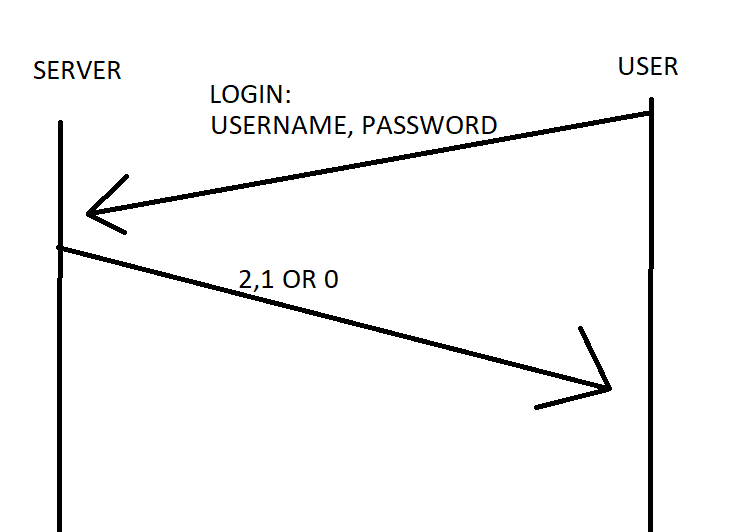
Quando rejeitado



Funcionamento do protocolo

O protocolo de comunicação utilizado

Para fazer login



o servidor devolvendo 0, é porque o user não existe

o servidor devolvendo 1 é porque a password está errada

o servidor devolvendo 2 o logIn foi feito com sucesso

Pedir viagem

