Trabalho Prático

Meta 1

Sistemas Operativos

Departamento de engenharia informática (DEIS)

Daniel Bravo (2021137795)
Daniel Rodrigues (2021142013)

1. Introdução

Neste relatório vamos descrever um pouco das nossas funcionalidades e estruturas feitas para a meta 1.

2. Estruturas de dados

A estrutura cliente é responsável por armazenar a informação toda de um cliente/administrador, como, o seu username e password após fazer login, o seu saldo, o seu estado de atividade(ativo/offline), e um ponteiro de estrutura que irá armazenar o cliente seguinte.

A estrutura item é responsável por armazenar as informações de cada item contém, um id de identificação, um nome e categoria desse item, um valor base quando é lançado em leilão, um valor de compra imediata, o username do seu vendedor e comprador e o tempo de venda.

A estrutura vendas é responsável por armazenar a informação acerca do vendedor atual e da licitação em leilão.

A estrutura B_F irá ser usada para passar informação entre frontend e background, contém o valor de licitação e o preço máximo, valor do depósito, o tempo que o item irá estar em venda, o comando(funcionalidade) pedida pelo cliente pedido pelo frontend e uma estrutura do item a passar.

3. Frontend

No frontend fizemos uma leitura e validação do username e password dado pelo argc e argv, sendo esta a ordem username password, caso um desses parâmetros seja inválido (tamanho da string) ou seja diferente de 3 o argc sai do programa.

Caso seja validado com sucesso de seguida o é pedido ao cliente qual a funcionalidade a testar e é feito uma leitura dos comandos e a sua validação apresentando ser válido ou inválido.

4. Backend

No backend é feito uma validação inicial caso seja lido um valor no argc diferente de 1 é dado uma indicação de erro e sai do programa. Feito a validação é pedido ao administrador qual a função que ele queira usar.

Em comandos apenas é feita uma validação de parâmetros.

Em execução do promotor, primeiro criamos um pipe anónimo e em seguida criamos um processo filho no qual eremos executar o promotor. Durante este processo iremos guarda o id do processo filho na variável global pd. Entretanto no processo pai criamos um sinal SIGINT que ao se premir Ctrl+C irá usar a função *sair* a qual criará e enviará um sinal do tipo SIGUSR1 para o processo filho que irá terminar na receção deste final.

Em Utlizadores é usado as funções fornecidas pela biblioteca users_lib.h que permite fazer a leitura do ficheiro de texto onde guarda(username password saldo) com loadUsersFile(), é possível verificar através do nome e password se existe algum user com esse nome no ficheiro com a função isUserValid(), obter o saldo de um user com getUserBalance(), alterar o saldo de user com updateUserBalance(), os valores retornados em caso erro é possível verificar os erros com getLastErrorText(), para além disso guardar os dados alterados no ficheiro txt com saveUsersFile(), no final é decrementado -1 ao saldo de todos os users utilizando open, strtok(para retirar os usernames do txt) e as funções de users_lib.h.

Em Itens decidimos usar o fopen() para a leitura do ficheiro itens.txt que retorna para uma variável o seu conteúdo e com auxílio do fscanf() podemos retirar inteiros e strings para outras variáveis.

5. Makefile

O makefile tem a opção all que compila todos os programas, o backend compila os seus programas necessários como backend.c e users_lib.o, o frontend compila o frontend.c, e o clean remove todos os executáveis.