# File Transfer Socket Application

Projeto de uma aplicação em redes utilizando sockets apresentado como requisito avaliativo para a disciplina de Redes de computadores, ministrada no Instituto de Computação da Universidade Federal de Alagoas (UFAL)

## Alunos


- Lucas Lemos Cerqueira
- Pedro Henrique de Brito Nascimento


## Principais Funcionalidades

A aplicação dispõem de 4 funcionalidades:

- Upload de arquivos para o servidor:

  O cliente solicita a conexão com o servidor por meio do socket junto ao caminho absoluto do arquivo que deseja enviar. Caso o caminho seja válido o arquivo é salvo no servidor.

- Download de arquivos do servidor:

  O cliente solicita a conexão com o servidor por meio do socket junto ao nome do arquivo que deseja baixar (localizado na pasta "server_files"). Caso o arquivo cujo nome foi informado exista na pasta do servidor, o arquivo é baixado do servidor para o cliente (localizado na pasta "client_files").

- Listagem de arquivos do servidor:

  O cliente solicita a conexão com o servidor por meio do socket e o servidor retorna uma lista com os arquivos disponíveis no servidor (localizado na pasta "server_files").

- Remoção de arquivos do servidor:

  O cliente solicita a conexão com o servidor por meio do socket junto ao nome do arquivo que deseja remover (localizado na pasta "server_files"). Caso o arquivo cujo nome foi informado exista na pasta do servidor, o arquivo é excluído.

## O que poderia ser implementado a mais

O projeto poderia contar com algumas melhorias em suas funcionalidades, como por exemplo, a opção de upload e download poderia contar o envio/recebimento de múltiplos arquivos para evitar a necessidade de enviar/receber um arquivo por vez. Além disso, os arquivos armazenados no servidor poderia ser compactados para melhorar o aproveitamento de armazenamento.

## Principais dificuldade na implementação do projeto

Houveram dificuldades na utilização de algumas funções primitivas do socket, como recv() e send(), isso acontenceu pelo fato de que dependendo do tamanho do arquivo que se deseja enviar, o recv() não conseguia receber o arquivo completo, sendo necessário que na hora do envio do arquivo, além de enviar seu conteúdo e nome, era necessário também o envio do tamanho do arquivo, para que o servidor/cliente soubesse a hora de parar de receber dados.

## Executando o projeto

Com a pasta do projeto na sua máquina, primeiramente separe dois terminais um para o servidor e outro para o cliente, note que o servidor precisa ser o primeiro a ser executado. Para isso, siga o script abaixo para rodar o servidor.

```
$ cd server
$ python server.py
```

Esse script aceita dois parâmetros adicionais, um para definir o endereço Ip e o outro para definir o número da porta. Caso nenhum parâmetro seja adicionado, por padrão o socket será criado para o endereço IP da sua máquina na porta 20.

```
$ python server.py 192.168.0.21 8080
```

Com o lado do servidor em execução, utilize um outro terminal para executar o cliente. Para isso, siga o script abaixo para rodar o cliente.

```
$ cd client
$ python client.py
```

O script do cliente também aceita dois parâmetros adicionais da mesma forma que o servidor, um para o endereço IP e outro para definir o número da porta. Caso nenhum parâmetro seja adicionado, por padrão o socket será criado para o endereço IP da sua máquina na porta 20.

```
$ python client.py 192.168.0.21 8080
```

Com o cliente executando, será disposto um menu como o mostrado abaixo:

```
Functionalities:
- UPLD   Send File to the server
- DWLD   Download File from the server
- SHOW   List all files in the server
- DELT   Delete a File from the server
- HELP   Help
- DISC   Quit
Choose one command...
```

Ao escolher um comando, o mesmo contará com as informações necessárias para o correto funcionamento da funcionalidade.