

**Copia de arquivos via SOCKETS (CLIENT/SERVER)**

## 

**Eduardo Gomes Júnior – 30/10/2022**

Sumário

[1. Projeto: 3](#_Toc117960452)

[2. Instalação: 3](#_Toc117960453)

[3. Execução: 11](#_Toc117960454)

1. Projeto:

Aplicação para permitir copiar qualquer tipo de arquivo existente no servidor (SERVER) para qualquer máquina local. A configuração de servidor, porta e pasta de origem/destino é definida através de configurações em arquivos INIS. Tanto a aplicação SERVER como o CLIENT possuem arquivos de configuração INIS e podem ser configurados de acordo com a necessidade. Toda solicitação feita pelo CLIENT será registrada no SERVER em arquivo de LOG. Esse arquivo será usado para acompanhamento das solicitações (requisições) e tratamento de possíveis erros. Os arquivos disponíveis para copia são atualizados a cada solicitação ao SERVER e exibidas para o usuário em interface para que seja selecionado/escolhido o arquivo desejado.

1. Instalação:

Tanto o **BACK END** como **FRONT END** é desenvolvido em **PYTHON**. A versão do **PYTHON** usada é a 3.10, porém o mesmo funcionou normalmente nas versões 3.9 e 3.8. Para instalação/execução será necessário a instalação do **PYTHON** caso não exista.

DONWLOAD do PYTHON: <https://www.python.org/downloads/>

Aplicação desenvolvida para sistemas operacionais **Windows**. No **Linux** não foi testada.

A aplicação precisa de duas bibliotecas externas para o funcionamento.

- **configparser** (usada para leitura dos arquivos INIS)

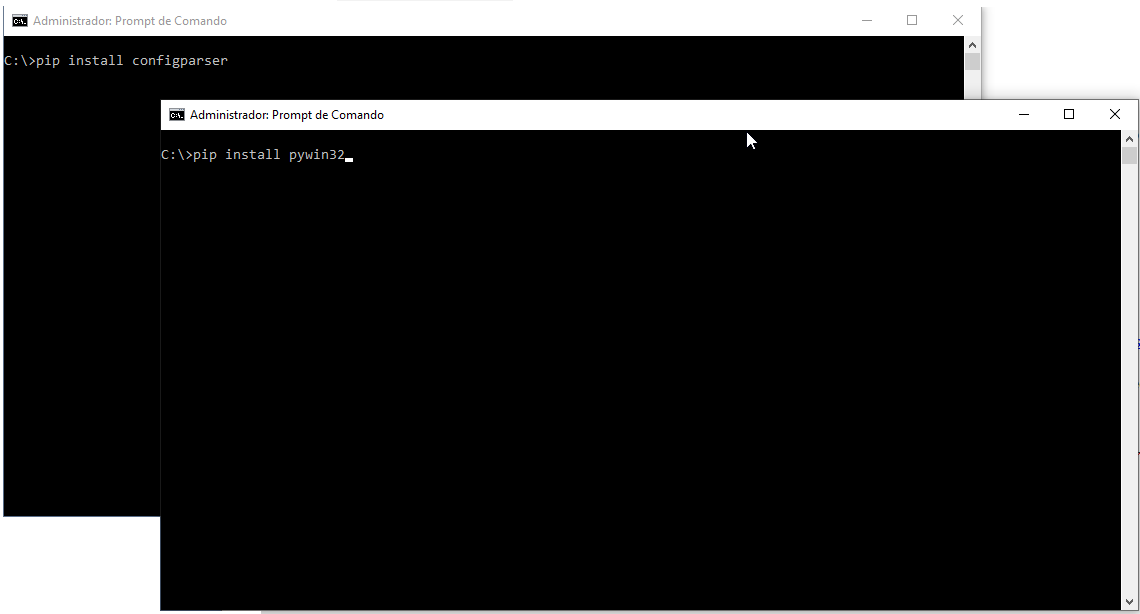
- **pywin32** (usada para criação/instalação do serviço no Windows)

Detalhes no arquivo: **requirements.txt**

Para instalar ambas utilizar PIP: acesse o PROMPT de comando (como administrador) e execute os comandos abaixo:

pip install configparser

pip install pywin32



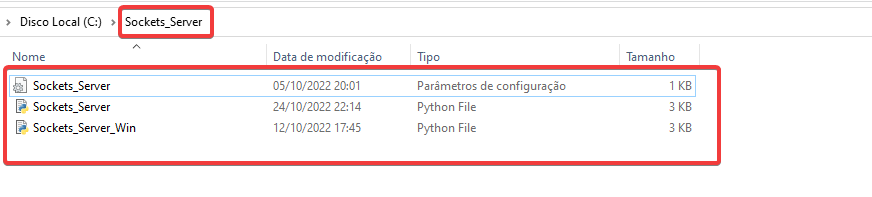
Após os pré-requisitos acima realizar o clone do repositório: <https://github.com/EduardoGomesJr/Sockets>

Como a aplicação é divida em duas partes: SERVIDOR e CLIENTE após descompactar o arquivo serão necessários realizar a instalação/configuração em duas etapas.

* **Servidor (SERVER):**

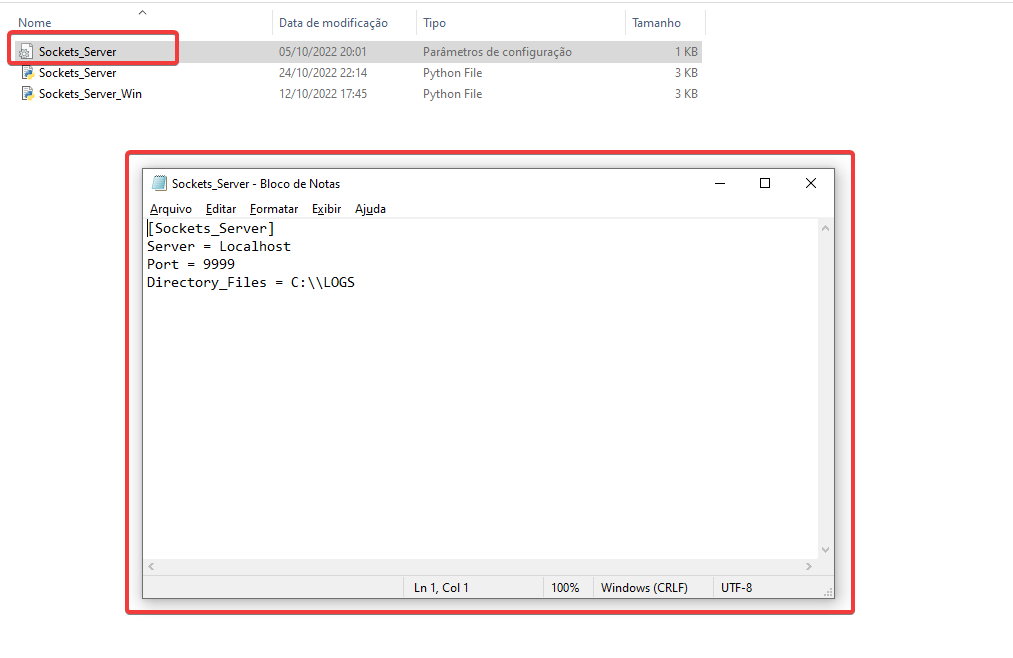
Na máquina que será usada como servidor criar uma pasta chamada: “Sockets\_Server”  
MD Sockets\_Server (para criar a pasta)

Dentro da pasta copiar os arquivos Sockets\_Server.INI, Sockets\_Server.PY e Sockets\_Server\_Win.PY



**Arquivos**:

Sockets\_Server.INI: contém a configuração de IP, porta do servidor e pasta (diretório) que contém os arquivos permitidos para copia através do CLIENT. Para os testes apresentados aqui o nome do Server está como LOCALHOST, pois os testes são em ambientes locais. Num ambiente oficial, basta trocar esse nome pelo IP do servidor e porta que será liberada. Exemplo 172.16.0.10 e porta 7070 e indicar um local valido para o DIRECTORY\_FILES (pasta que contém os arquivos liberados para copia).



A rotina (script) Sockets\_Server.py é a rotina usada como servidor (fica rodando (LOOP) em modo serviço aguardando a conexão de algum CLIENT e envia a informação/arquivo solicitada.

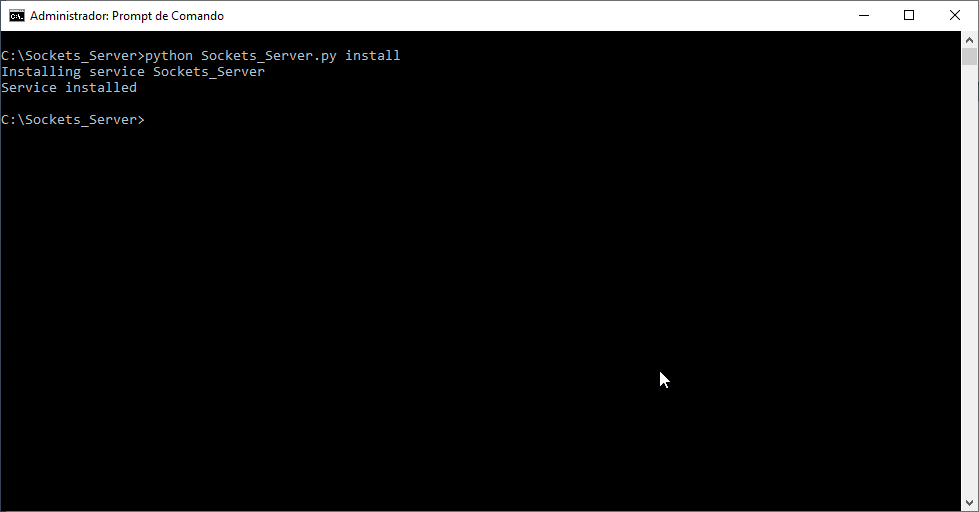
O Sockets\_Server\_Win.py é uma rotina responsável por criar um serviço PYTHON dentro do Windows.

Para criar/subir o serviço é necessário executar os passos abaixo:

- Acessar o PROMPT de comando como administrador.

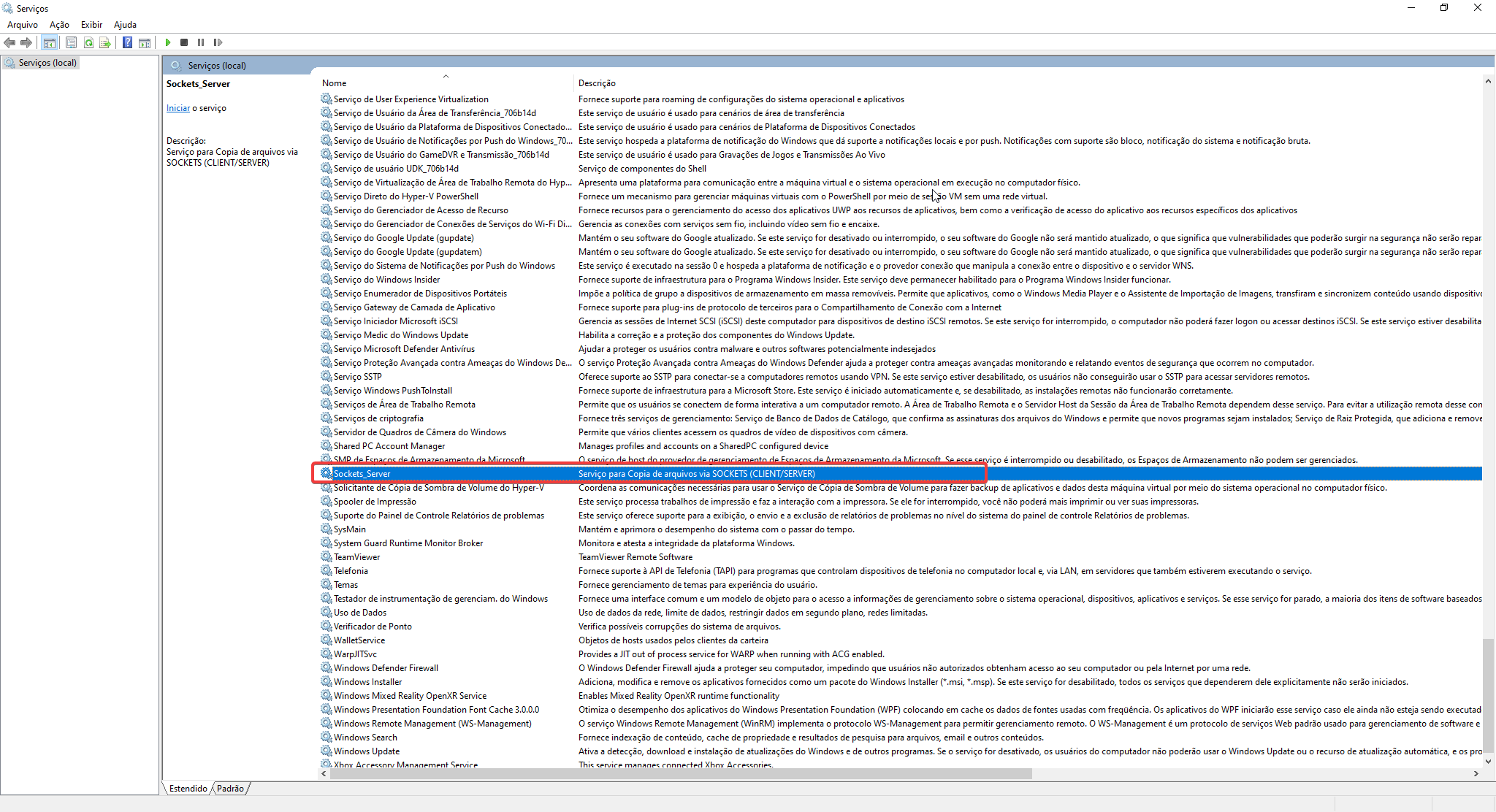
- Acessar a pasta criada do Sockets\_Server e dentro dela digitar o comando:

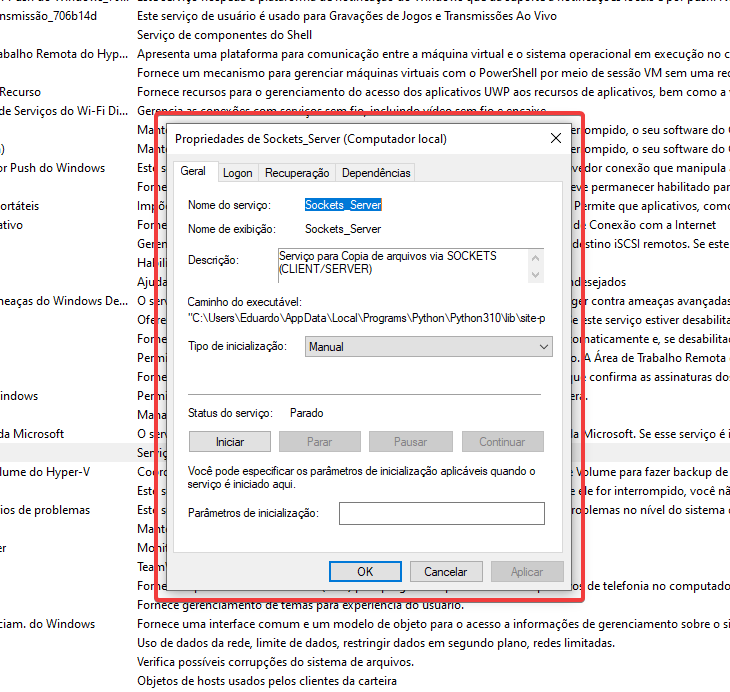
**python Sockets\_Server.py install <ENTER>**

****

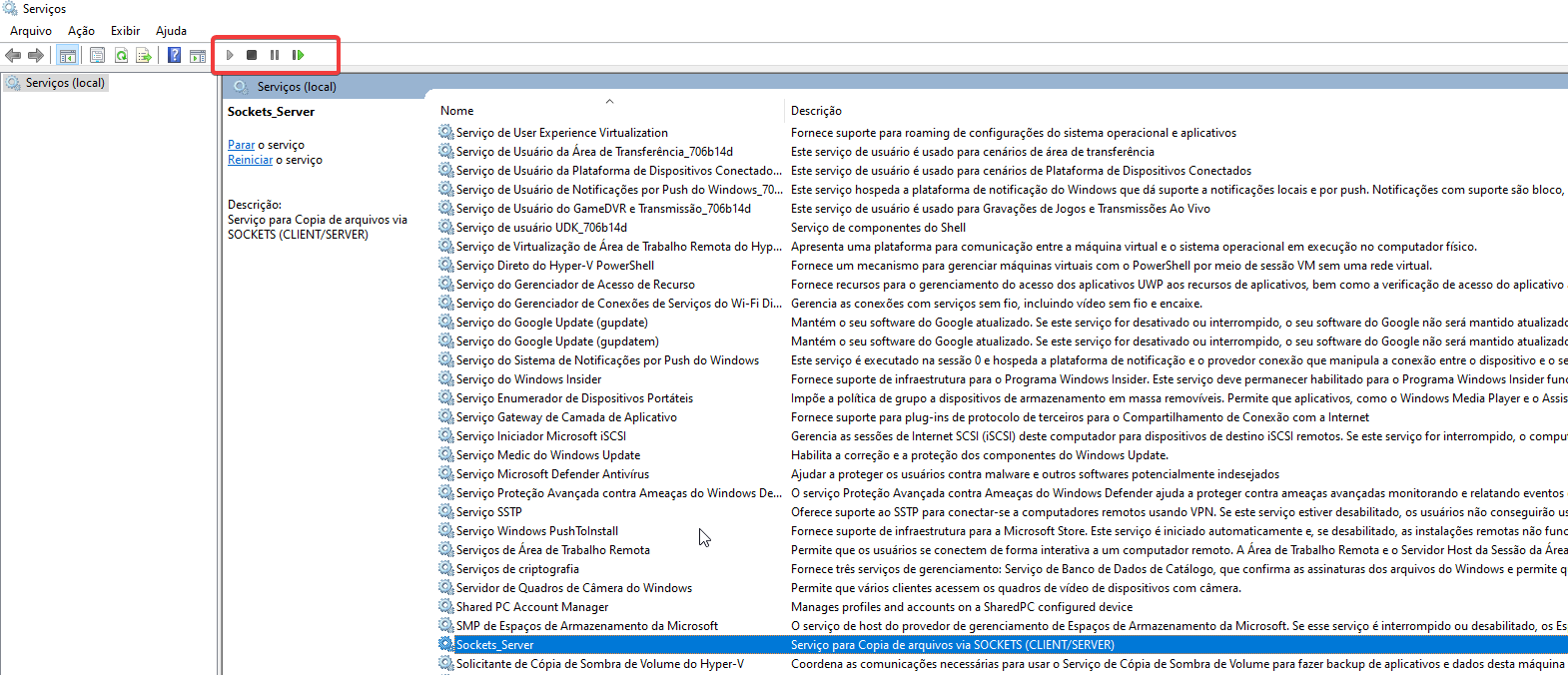
Através do comando acima será criado no serviço do Windows um serviço chamado Sockets\_Server o nome e informações exibidas nesse serviço ficam dentro da rotina Sockets\_Server.py

Para conferir o serviço gerado, basta abrir o serviço do Windows e procurar o serviço criado:



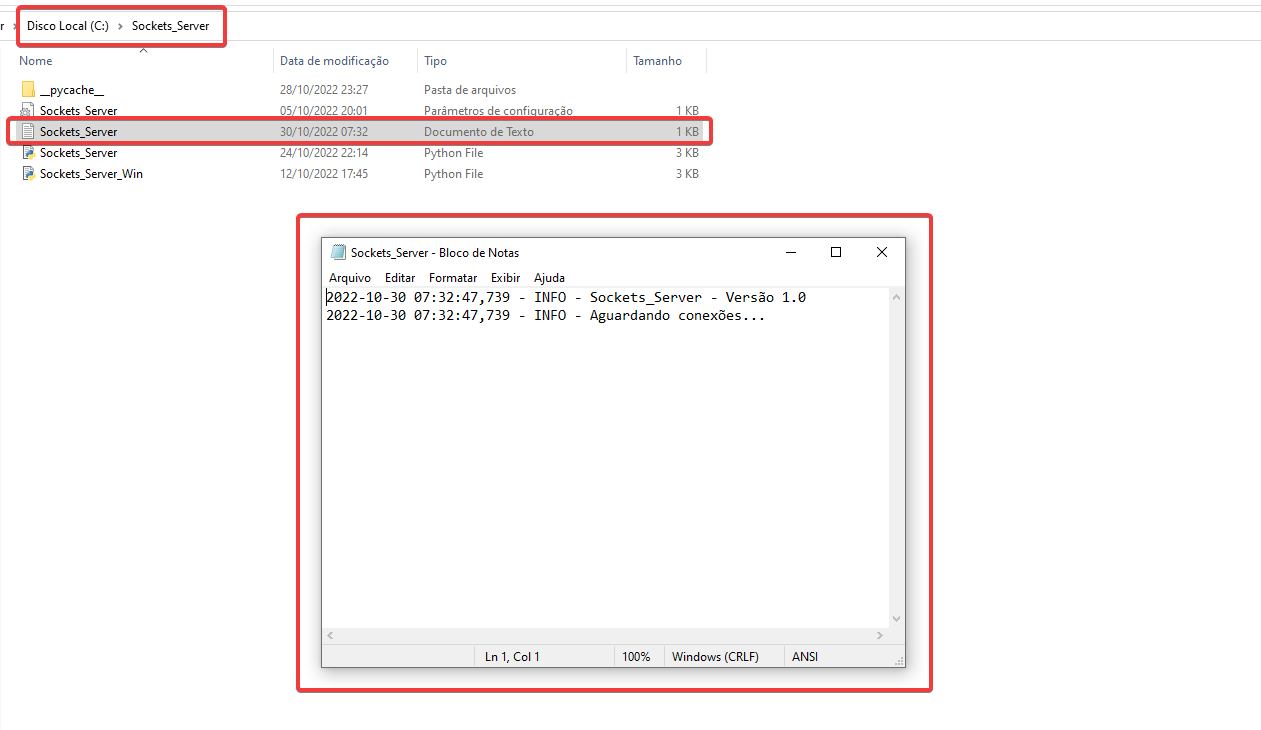


Para iniciar/parar o serviço basta iniciar ele normalmente usando as opções disponíveis.



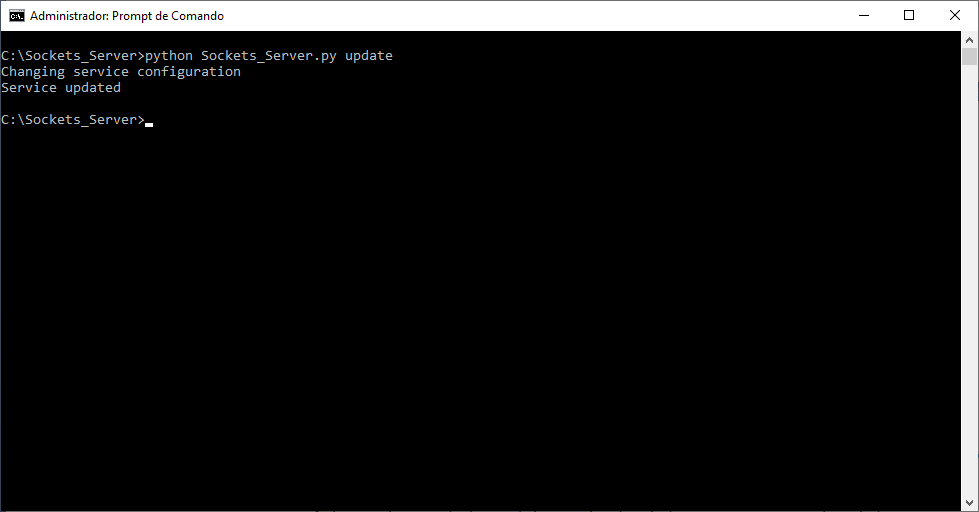
Após o serviço iniciado a parte do servidor não precisa mais ser alterado, pode ficar rodando o tempo necessário.

Assim que o serviço é criado/iniciado dentro da pasta Sockets\_Server é criado o arquivo de LOG para monitoramento das solicitações/requisições/envio. O arquivo de LOG é gravado como Sockets-Server.LOG é um arquivo texto e pode ser visualizado com qualquer editor.



Caso haja necessidade de alguma alteração nas rotinas Sockets\_Server.py ou Sockets\_Server\_Win.py para subir essas alterações para o serviço. Pasta acessar novamente a pasta do Sockets\_Server como administrador e executar o comando:

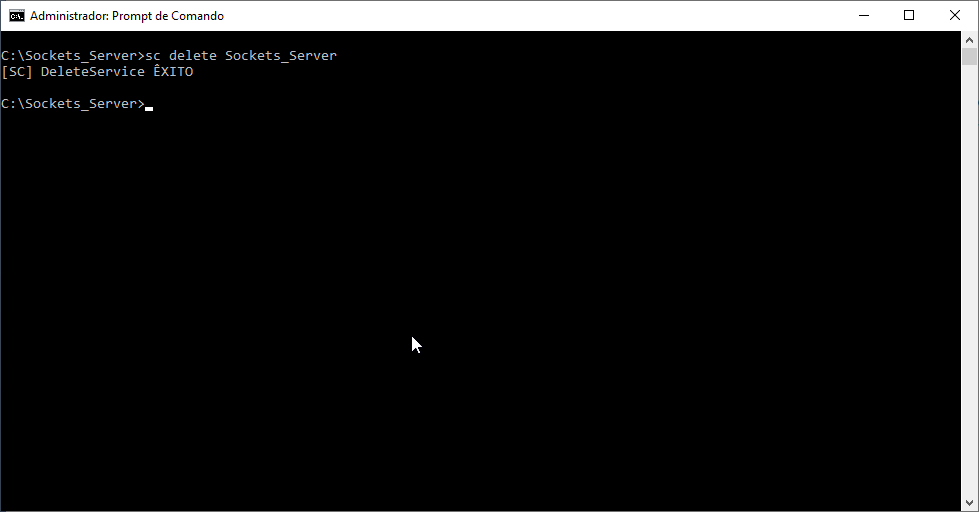
**python Sockets\_Server.py update <ENTER>**

****

Através desse comando as alterações são enviadas para o serviço criado.

Caso haja necessidade de excluir o serviço por algum motivo, basta usar o comando abaixo:

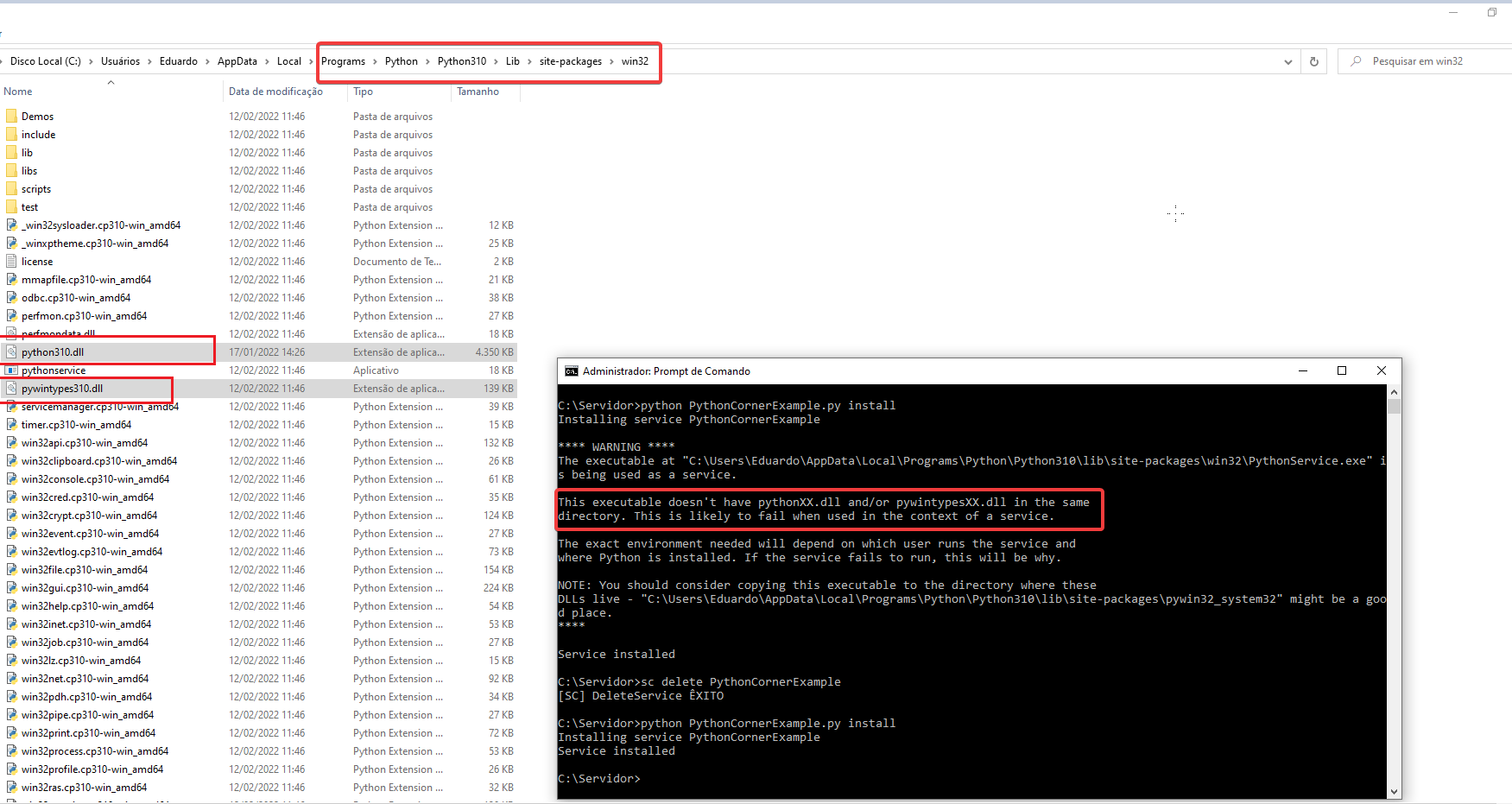
**sc delete Sockets\_Server**

****

Após isso o serviço é excluído do Windows.

Observações: em algumas instalações do serviço pode ocorrer erro de DLLS (bibliotecas) não encontradas. Caso ocorra esse problema pasta copiar os arquivos python310.dll e pywintypes310.dll para dentro da pasta: Python\Python310\lib\site-packages\win32 que o problema é resolvido. PYTHON310 corresponde a versão do Python instalada esse número pode mudar caso seja usado outra versão.

A DLL python310.dll fica no caminho: “\AppData\Local\Programs\Python\Python310” já a pywintypes310.dll em: \AppData\Local\Programs\Python\Python310\Lib\site-packages\pywin32\_system32



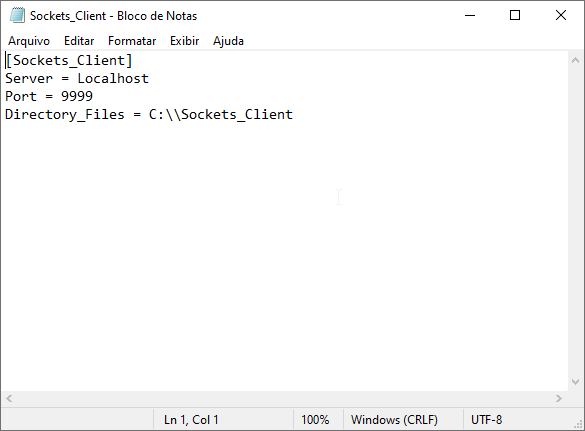
* **CLIENT (CLIENTE):**

Para instalação do CLIENT o processo é igual ao SERVER. Deve ser criada uma pasta chamada: Sockets\_Client

MD Sockets\_Server (para criar a pasta)

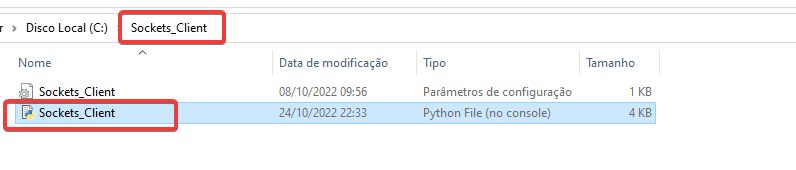
Dentro da pasta copiar os arquivos Sockets\_Client.INI e Sockets\_Client.PYW

O Arquivo INI do CLIENT contém as informações do servidor que será acessado juntamente da sua porta iguais ao do SERVER. A única diferença aqui é o Directory\_Files o caminho dele indica onde serão gravados os arquivos copiados do servidor.

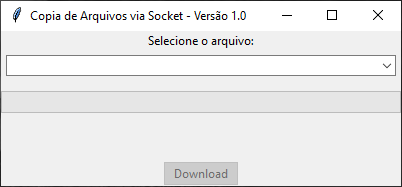


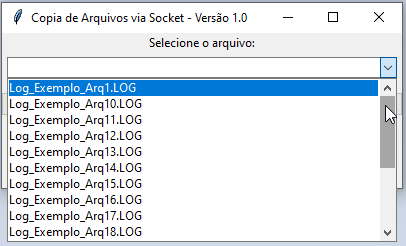
1. Execução:

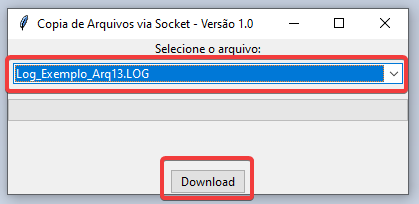
Com o serviço criado/iniciado, basta executar o rotina (script) Sockets\_Cliente.pyw (dentro da pasta Sockets\_Client) após isso é carregado a interface de seleção de arquivos.



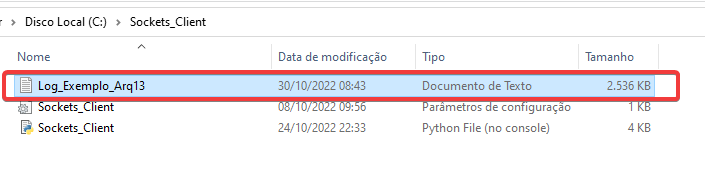
Será aberta a tela (interface) para seleção de arquivos. Basta pressionar o COMBOBOX e todos os arquivos disponíveis no servidor são carregados para está lista. Após isso selecionar o arquivo deseja e pressionar o botçao DOWNLOAD. Com isso o arquivo selecionado é copiado do SERVIDOR para o CLIENTE (a pasta que ele irá ser salvo é a pasta informada no arquivo INI do CLIENT).



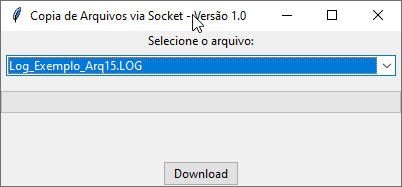


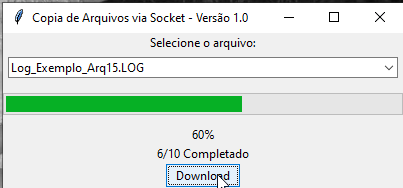


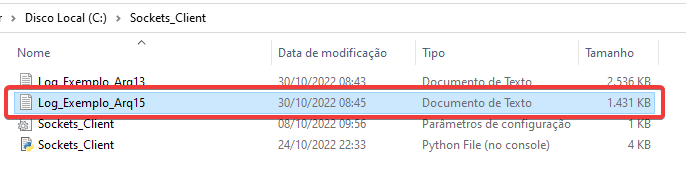
Após pressionar o botão DOWNLOAD o arquivo selecionado é copiado para o CLIENTE.



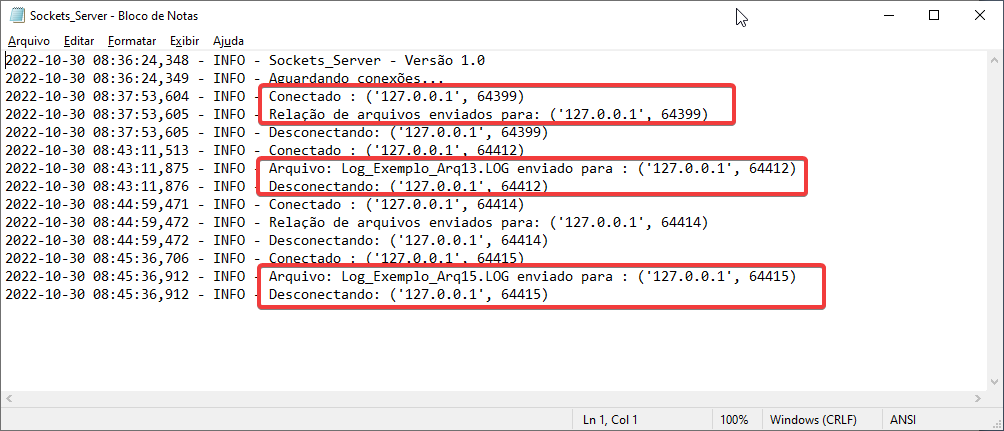
Realizando outra copia:







Consultando o arquivo de LOG do SERVIDOR, temos todas as solicitações/requisições realizadas no arquivo:



Quando é aberta a rotina CLIENT é enviada uma solicitação para o servidor solicitando a relação de arquivos existentes no diretório configurado para copia, após isso essa relação é atualizada no COMBOBOX da rotina, na sequencia o usuário seleciona um determinado arquivo e pede para o servidor mandar para ele, com isso o arquivo é enviado. Todas essas solicitações/requisições são gravadas no LOG do Server.

**Observações**: na parte do servidor é necessário ter o PYTHON instalado para poder criar/subir o serviço no Windows. Já para o CLIENTE, caso ele não queria instalar o PYTHON pode ser criado um executável da rotina Sockets\_Client.PYW. Esse executável terá todas as bibliotecas básicas usadas (deve ser criado um ambiente virtual).

Para criação do executável, precisa ser instalado o pacote: pyinstaller

pip install pyinstaller

Após isso gerar o executável: pyinstaller --onefile -w Sockets\_Client.pyw