Plano de Teste

O presente Plano de Teste visa assegurar diversas funcionalidades do sistema de Manipulação de Arquivos Entre Processos. A estrutura do plano compreende testes de inicialização do servidor e cliente, operações de listagem, deleção, upload e download de arquivos, culminando em testes de conexão simultânea e operações concorrentes entre clientes. Através da execução e análise desses testes, almejamos garantir um ambiente estável e funcional para o sistema em questão.

Testes de Inicialização do Servidor:

Teste 1: Inicialização bem-sucedida do Servidor
Cenário: O servidor tenta inicializar um socket TCP com IP/Porta válidos.
Passos:
Inicia o servidor.
Resultado Esperado: O servidor imprime uma mensagem de sucesso.
Resultado Obtido: `Server listening on 192.168.56.1:12345`

Teste 2: Falha na Inicialização do Servidor (IP/Porta Inválidos)
Cenário: O servidor tenta inicializar um socket TCP com IP/porta inválidos.
Passos:
Configura o servidor com IP/porta inválidos.
Inicia o servidor.
Resultado Esperado: O servidor imprime uma mensagem de erro no terminal.
Resultado Obtido: `socket.gaierror: [Errno 11001] getaddrinfo failed`

Teste 2.1: Falha na Inicialização do Servidor (Servidor já iniciado)
Cenário: O servidor já foi iniciado e tenta ser iniciado em outro terminal.
Passos:
Inicia o servidor com IP/Porta válidos em um terminal.
Inicia o servidor em outro terminal.
Resultado Esperado: O servidor imprime uma mensagem de erro no segundo terminal.
Resultado Obtido:

```
OSError: [WinError 10048] Only one usage of each socket address
(protocol/network address/port) is normally permitted
```

Testes de Inicialização do Cliente:

Teste 3: Inicialização bem-sucedida do Cliente
Cenário: O cliente tenta inicializar a conexão com o servidor.
Passos:
Inicia o cliente.

Resultado Esperado: O cliente imprime a resposta do servidor (formato JSON) e uma mensagem de conexão bem-sucedida com o servidor.. O servidor imprime mensagens de conexão aceita, a mensagem de requisição (formato JSON) que foi enviada pelo cliente e uma mensagem de conexão estabelecida com o cliente.
Resultado Obtido:
Do lado do Cliente:

```
{"status": "success", "message": "Connection established."}
Client has connected with the server.
```

Do lado do Servidor:

```
Server listening on 192.168.56.1:12345
Accepted connection from ('192.168.56.1', 64026)
JSON_Request:
{"tipo_requisicao": "ESTABLISHING_CONNECTION"}
Connection established with the client.
```

Teste 4: Falha na Inicialização do Cliente (Servidor não Ouvindo, IP/Porta Inválidos)
Cenário: O cliente tenta inicializar a conexão com um servidor que não está ouvindo ou com IP/porta inválidos.
Passos:
Configura o cliente com IP/porta inválidos ou servidor indisponível.
Inicia o cliente.
Resultado Esperado: O cliente imprime uma mensagem de erro no terminal indicando que a conexão falhou.
Resultado Obtido:

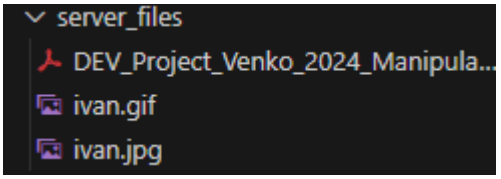
```
ConnectionRefusedError: [WinError 10061] No connection could
 be made because the target machine actively refused it
```

Testes de Listagem dos Arquivos Disponíveis no Servidor:

Teste 5: Listagem Bem-Sucedida (Com Arquivos no Diretório)
Cenário: Executa a operação de listar quando há arquivos no diretório.
Passos:
Adiciona arquivos ao diretório.
Executa a operação de listar.
Resultado Esperado: A lista de arquivos é exibida corretamente.
Resultado Obtido:

```
DEV_Project_Venko_2024_Manipulação_Arquivos.pdf
ivan.gif
ivan.jpg
```

Sendo estes mesmos os arquivos no diretório:

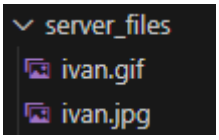


Teste 6: Listagem Sem Arquivos no Diretório
Cenário: Executa a operação de listar quando não há arquivos no diretório.

Passos:
Limpa o diretório.
Executa a operação de listar.
Resultado Esperado: Uma mensagem indicando que não há arquivos no diretório é exibida.
Resultado Obtido: `The server directory does not contain any files.`

Testes de Deleção de Algum Arquivo no Servidor:

Teste 7: Deleção Bem-Sucedida
Cenário: Executa a operação de deletar um arquivo existente.
Passos:
Adiciona um arquivo ao diretório.
Executa a operação de deletar o arquivo.
Resultado Esperado: O arquivo é removido da lista e uma mensagem de sucesso é exibida.
Resultado Obtido: Diretório do servidor antes da operação de deletar o arquivo:



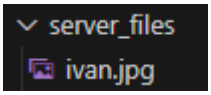
Operação realizada:

```
> delete
Filename: ivan.gif
```

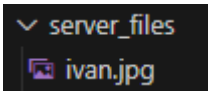
Mensagem de sucesso exibida após a operação:

```
{"status": "success", "message": "File deleted successfully."}
```

Diretório após a operação de deletar o arquivo:



Teste 8: Falha na Deleção (Arquivo Não Existente)
Cenário: Executa a operação de deletar um arquivo que não existe.
Passos:
Garante que o arquivo que será requisitado para deleção não existe no servidor.
Executa a operação de deletar um arquivo inexistente.
Resultado Esperado: Uma mensagem de erro indicando que o arquivo não existe é exibida.
Resultado Obtido: Diretório do servidor antes da operação de deletar o arquivo no servidor:



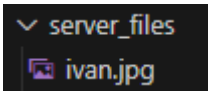
Operação Realizada:

```
> delete
Filename: ivan.gif
```

Mensagem de erro exibida após a operação:

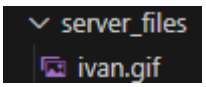
```
{"status": "error", "message": "File not found."}
```

Diretório do servidor após a operação de deletar o arquivo inexistente:



Testes de Upload de Arquivo para o Servidor:

Teste 9: Upload Bem-Sucedido (Arquivo Existente no diretório do cliente)
Cenário: Executa a operação de upload com um arquivo existente no diretório do cliente.
Passos:
Garante que o arquivo que será requisitado para upload existe no diretório do cliente.
Executa a operação de upload do arquivo.
Compara o arquivo encontrado no diretório do cliente e o arquivo adicionado ao diretório do servidor.
Resultado Esperado: O cliente exibe uma mensagem confirmando que o arquivo existe. O servidor responde ao cliente com status de sucesso e uma mensagem indicando que está recebendo o arquivo. Os dados do arquivo são exibidos corretamente no diretório do servidor.
Resultado Obtido: Diretório do servidor antes da operação de upload de arquivo para o servidor:



Operação realizada:

```
> upload
Filename: ivan.jpg
```

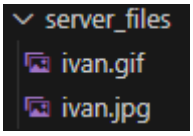
Mensagem de sucesso exibida pelo cliente após a requisição:

```
File exists: ./client_files/ivan.jpg
```

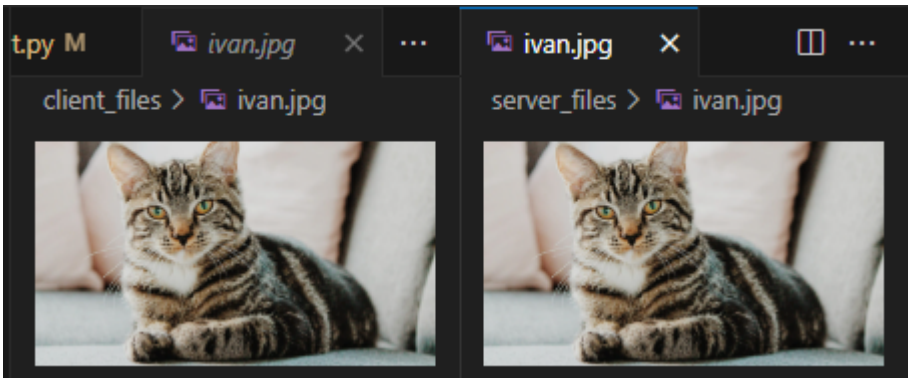
Resposta de sucesso enviada pelo servidor ao cliente após a requisição:

```
{"status": "success", "message": "Receiving file..."}
```

Diretório do servidor após a operação de upload de arquivo para o servidor:



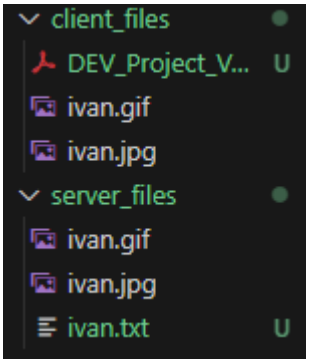
Comparação entre o arquivo encontrado no diretório do cliente e o arquivo adicionado ao diretório do servidor:



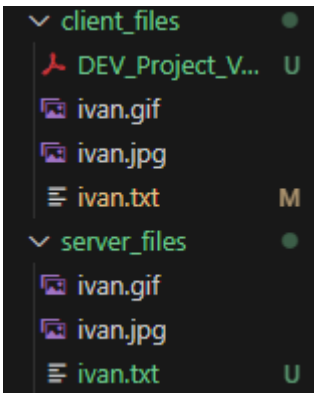
Teste 10: Falha no Upload (Arquivo Não Existente)
Cenário: Executa a operação de upload com um arquivo que não existe.
Passos:
Garante que o arquivo que será requisitado para upload ao servidor não existe no diretório do cliente.
Executa a operação de upload de um arquivo inexistente.
Resultado Esperado: Uma mensagem de erro indicando que o arquivo não existe é exibida.
Resultado Obtido:
```
File does not exist: ./client_files/ola mundo
```

Testes de Download de Arquivo do Servidor:

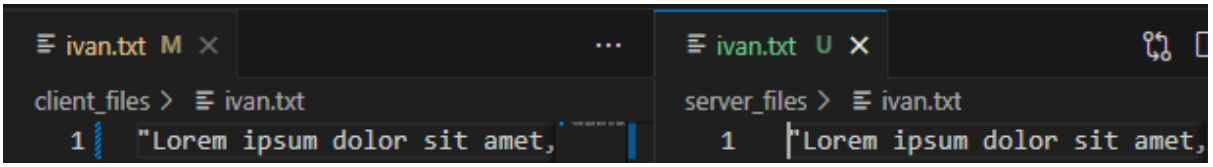
Teste 11: Download Bem-Sucedido (Arquivo Existente)
Cenário: Executa a operação de download com um arquivo existente no servidor.
Passos:
Garante que o arquivo que será requisitado para download existe no diretório do servidor.
Garante que o arquivo ainda não existe no diretório do cliente (para fins de comparação dos dados dos arquivos após a operação; embora também tenha sido realizado o teste com o arquivo já existindo no diretório do cliente: o resultado é igual, com o arquivo antigo sendo sobrescrito).
Executa a operação de download do arquivo.
Compara o arquivo encontrado no diretório do servidor e o arquivo adicionado ao diretório do cliente.
Resultado Esperado: O cliente imprime uma resposta de sucesso enviada pelo servidor ao cliente. O cliente imprime uma mensagem indicando que o download do arquivo foi bem-sucedido. Os dados do arquivo são exibidos corretamente no diretório do cliente.
Resultado Obtido:
Arquivos existentes nos diretórios do cliente e servidor antes da operação:



Operação realizada:
```
> download
Filename: ivan.txt
```
Resposta de sucesso enviada pelo servidor ao cliente:
```
{"status": "success", "message": "File found.
 Transference started."}
```
Mensagem exibida pelo cliente indicando que o download do arquivo foi bem-sucedido:
```
Download of file 'ivan.txt' completed. Saved
in './client_files/'.
```
Diretórios do cliente e servidor após a operação de download de arquivo para o cliente:



Comparação entre o arquivo encontrado no diretório do servidor e o arquivo adicionado ao diretório do cliente:



Teste 12: Falha no Download (Arquivo Não Existente no Servidor)
Cenário: Executa a operação de download com um arquivo que não existe no servidor.
Passos:
Garante que o arquivo que será requisitado na operação de download não existe no diretório do servidor.
Executa a operação de download de um arquivo inexistente.
Resultado Esperado: O cliente imprime uma resposta de erro enviada pelo servidor ao cliente. O cliente imprime uma mensagem de erro.
Resultado Obtido:

```
{"status": "error", "message": "File not found."}
Error during download of file 'ola mundo': File not found.
```

Testes de Conexão e Operação Simultânea de Clientes:

Para os testes a partir do teste 13, “Máquina A” representa uma máquina com Sistema Operacional Microsoft Windows 10. “Máquina B” representa uma máquina virtual com Sistema Operacional Ubuntu.

Teste 13: Casos de sucesso relatados nos testes anteriores são executados durante conexão simultânea de clientes.
Cenário: Uma determinada ordem de comandos respectivos aos casos de sucesso relatados nos testes anteriores são executados por determinado cliente em sua respectiva máquina.
Passos:
Garante que nenhum arquivo está presente no diretório do servidor.
Inicia o Servidor na Máquina A.
Inicia o Cliente na Máquina B.

Inicia o Cliente na Máquina A.
Máquina A requisita upload de um arquivo ao servidor.
Máquina A requisita a listagem de arquivos.
Máquina B requisita a listagem de arquivos.
Máquina B requisita upload de outro arquivo ao servidor.
Máquina A requisita a listagem de arquivos.
Máquina B requisita a listagem de arquivos.
Máquina A requisita o download do arquivo que foi adicionado pela máquina B.
Máquina B requisita o download do arquivo que foi adicionado pela máquina A.
Máquina A requisita a deleção de um dos arquivos.
Máquina A requisita a listagem de arquivos.
Máquina B requisita a listagem de arquivos.
Máquina B requisita o upload do arquivo que foi deletado por A.
Máquina A requisita a listagem de arquivos.
Máquina B requisita a listagem de arquivos.
Máquina B requisita a deleção de um dos arquivos.
Máquina A requisita a listagem de arquivos.
Máquina B requisita a listagem de arquivos.
Compara os arquivos encontrados no diretório do servidor, no diretório do cliente da Máquina A e no diretório do cliente da Máquina B.

Resultado esperado: O servidor é iniciado com sucesso. A conexão do cliente da máquina B é estabelecida com sucesso. A conexão do cliente da máquina A é estabelecida com sucesso. O cliente da máquina A realiza com sucesso o upload de um arquivo ao servidor. Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtém a mesma resposta: deve ser exibido apenas o arquivo que foi adicionado pelo cliente da máquina A na operação anterior. O cliente da máquina B realiza com sucesso o upload de outro arquivo do servidor. Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtêm a mesma resposta: devem ser exibidos ambos os arquivos que foram adicionados. O cliente da Máquina A realiza com sucesso o download do arquivo que foi adicionado pelo cliente da máquina B e o cliente da Máquina B realiza com sucesso o download do arquivo que foi adicionado pelo cliente da Máquina A. O cliente da Máquina A deleta com sucesso um dos arquivos do servidor. Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtêm a mesma resposta: apenas um dos arquivos deve ser listado. O cliente da Máquina B realiza com sucesso o upload do arquivo que havia sido deletado. Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtêm a mesma resposta: dois arquivos devem ser listados. O cliente da Máquina B deleta com sucesso um dos arquivos do servidor. Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtêm a mesma resposta: apenas um arquivo deve ser listado. Os dados do único arquivo presente no diretório do servidor devem ser iguais ao arquivo de mesmo nome que deve estar presente nos diretórios dos clientes da Máquina A e Máquina B. O outro arquivo utilizado no teste deve estar presente nos diretórios dos clientes da Máquina A e Máquina B e seus dados devem ser iguais.

Resultado Obtido: Servidor é iniciado com sucesso:

```
Server listening on 192.168.56.1:12345
```

Conexão do cliente da máquina B é estabelecida com sucesso:

```
{"status": "success", "message": "Connection established."}
Client has connected with the server.
```

Conexão do cliente da máquina A é estabelecida com sucesso:

```
{"status": "success", "message": "Connection established."}
Client has connected with the server.
```

O cliente da máquina A realiza com sucesso o upload de um arquivo ao servidor:

```
> upload
Filename: ivan.jpg
File exists: ./client_files/ivan.jpg
{"status": "success", "message": "Receiving file..."}
```

Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtém a mesma resposta: deve ser exibido apenas o arquivo que foi adicionado pelo cliente da máquina A na operação anterior:

```
> list
ivan.jpg
```

```
> list
ivan.jpg
```

O cliente da máquina B realiza com sucesso o upload de outro arquivo do servidor:

```
> upload
Filename: ivan.txt
File exists: ./client_files/ivan.txt
{"status": "success", "message": "Receiving file..."}
```

Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtêm a mesma resposta: devem ser exibidos ambos os arquivos que foram adicionados:

```
> list
ivan.jpg
ivan.txt
```

```
> list
ivan.jpg
ivan.txt
```

O cliente da Máquina A realiza com sucesso o download do arquivo que foi adicionado pelo cliente da máquina B e o cliente da Máquina B realiza com sucesso o download do arquivo que foi adicionado pelo cliente da Máquina A:

```
> download
Filename: ivan.txt
{"status": "success", "message": "File found. Transference started."}
Download of file 'ivan.txt' completed. Saved in './client_files/'.
```

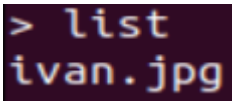
```
> download
Filename: ivan.jpg
{"status": "success", "message": "File found. Transference started."}
Download of file 'ivan.jpg' completed. Saved in './client_files/'.
```

O cliente da Máquina A deleta com sucesso um dos arquivos do servidor.

```
> delete
Filename: ivan.txt
{"status": "success", "message": "File deleted successfully."}
```

Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtêm a mesma resposta: apenas um dos arquivos deve ser listado:

```
> list
ivan.jpg
```

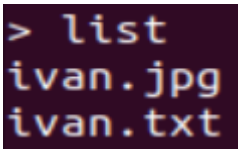


O cliente da Máquina B realiza com sucesso o upload do arquivo que havia sido deletado:

```
> upload
Filename: ivan.txt
File exists: ./client_files/ivan.txt
{"status": "success", "message": "Receiving file..."}
```

Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtêm a mesma resposta: dois arquivos devem ser listados:

```
> list
ivan.jpg
ivan.txt
```

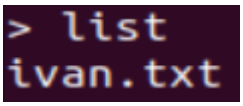


O cliente da Máquina B deleta com sucesso um dos arquivos do servidor:

```
> delete
Filename: ivan.jpg
{"status": "success", "message": "File deleted successfully."}
```

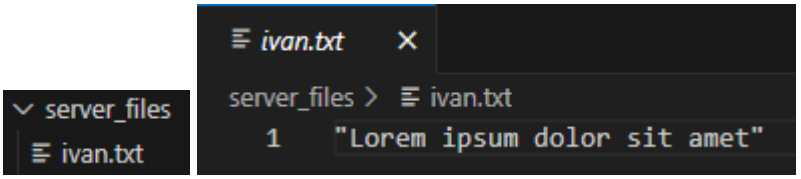
Ambos os clientes solicitam a listagem de arquivos disponíveis no servidor e obtêm a mesma resposta: apenas um arquivo deve ser listado:

```
> list
ivan.txt
```

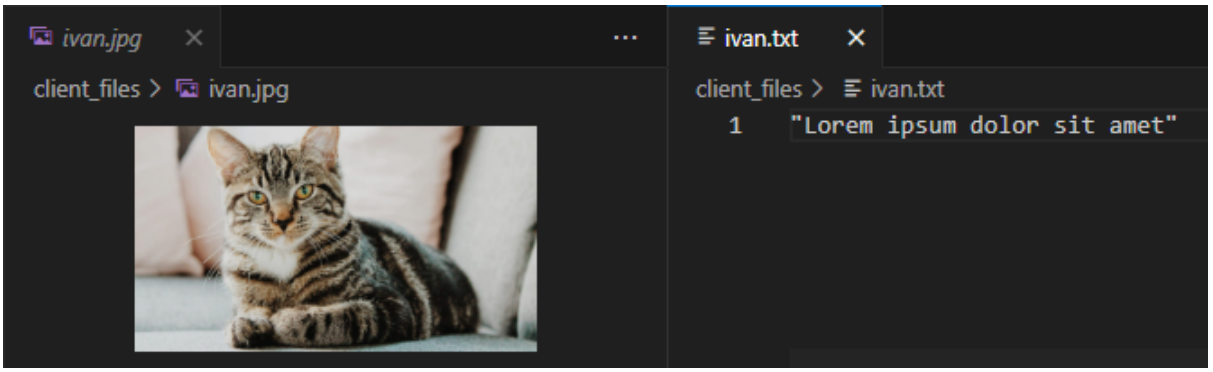


Comparação dos arquivos após as operações e seus dados:

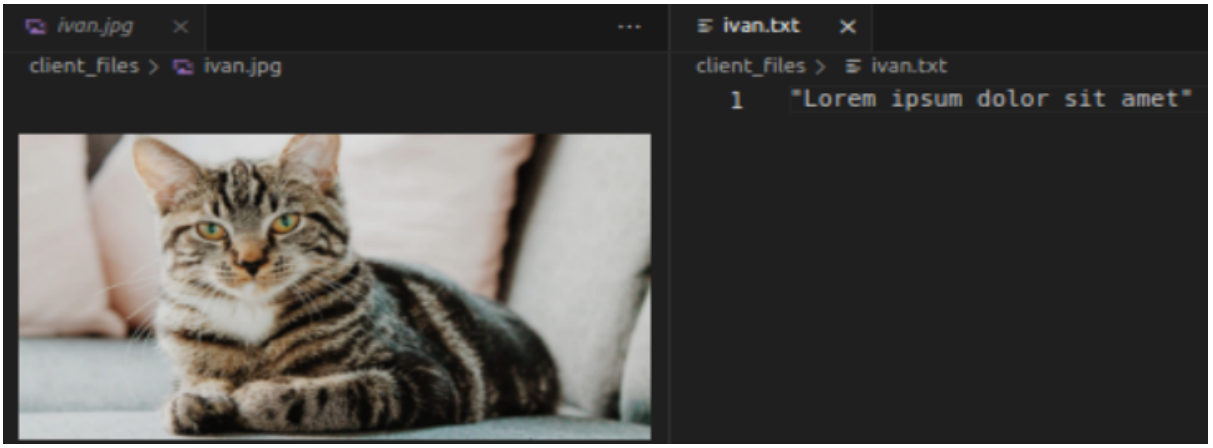
Arquivos do Servidor:



Arquivos do cliente da Máquina A:



Arquivos do cliente da Máquina B:



Teste 14: Dois clientes fazem simultaneamente upload de arquivos com nomes diferentes e dados diferentes. Um arquivo possui tamanho de 399MB e o outro arquivo possui tamanho 135KB.
Cenário: O cliente da Máquina A requisita o upload de um arquivo com tamanho de 399MB. Poucos milisegundos depois, cliente da Máquina B requisita o upload de outro arquivo com tamanho de 135KB.
Passos:
Cliente da Máquina A requisita o upload de um arquivo com 399MB.
Cliente da Máquina B requisita o upload de outro arquivo com 135KB.
Compara os arquivos encontrados no diretório do servidor, no diretório do cliente da Máquina A e no diretório do cliente da Máquina B.
Resultado esperado: Ambos os clientes recebem respostas de sucesso do servidor. Ambos os arquivos e seus respectivos dados devem estar presentes no diretório do servidor. É esperado que a transferência do arquivo de tamanho de 135KB – tamanho significativamente menor – deve finalizar antes do que a transferência do arquivo de tamanho de 399MB.

Resultado Obtido: Ambos os clientes receberam respostas de sucesso do servidor. Ambos os arquivos e seus respectivos dados estão presentes no diretório do servidor. A transferência do arquivo de 135KB finalizou antes do que a transferência do arquivo de 399MB.

Terminal da Máquina A:

```
> upload
Filename: ivan1.mp4
File exists: ./client_files/ivan1.mp4
{"status": "success", "message": "Receiving file..."}
```

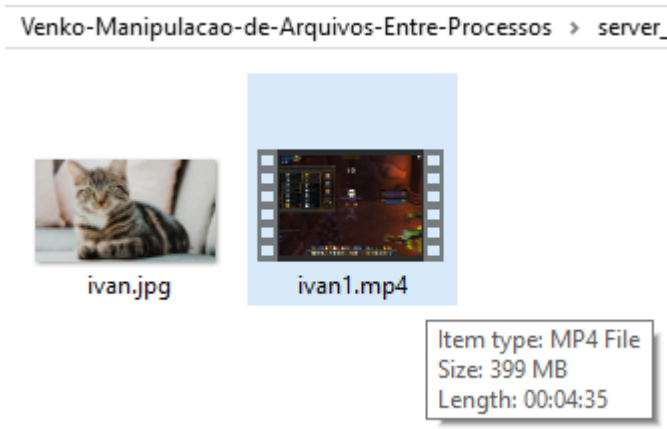
Terminal da Máquina B:

```
> upload
Filename: ivan.jpg
File exists: ./client_files/ivan.jpg
{"status": "success", "message": "Receiving file..."}
```

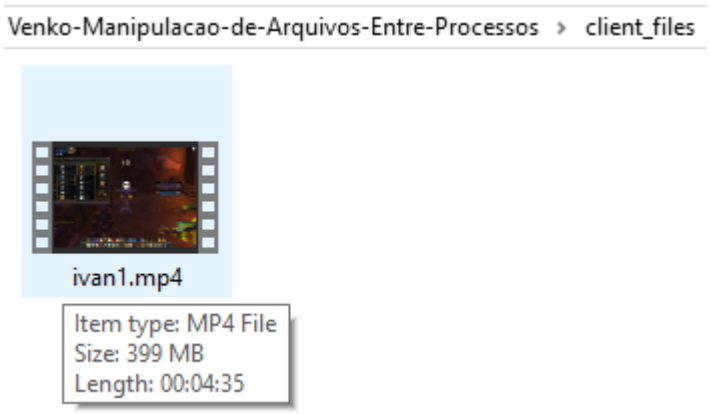
Terminal do Servidor:

```
JSON_Request:
{"tipo_requisicao": "UPLOAD", "nome_arquivo": "ivan1.mp4"}
Receiving file 'ivan1.mp4'...
JSON_Request:
{"tipo_requisicao": "UPLOAD", "nome_arquivo": "ivan.jpg"}
Receiving file 'ivan.jpg'...
Transfer of file 'ivan.jpg' completed.
Transfer of file 'ivan1.mp4' completed.
```

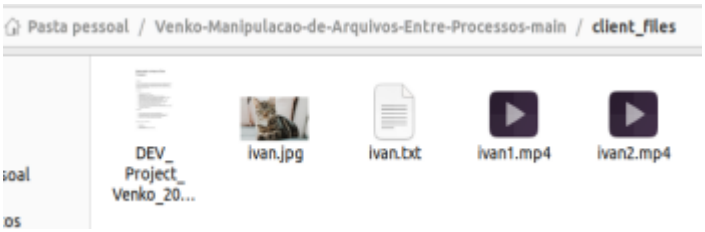
Diretório do Servidor, contendo os dois arquivos e seus respectivos dados corretamente:



Arquivos originais:
Diretório do Cliente da Máquina A:



Diretório do Cliente da Máquina B:



Teste 15: Dois clientes fazem simultaneamente upload de arquivos com nomes diferentes e dados diferentes. Ambos os arquivos com tamanhos próximos a 400MB.
Cenário: O cliente da Máquina A requisita o upload de um arquivo com tamanho próximo a 400MB. Poucos milisegundos depois, o cliente da Máquina B requisita o upload de outro arquivo com tamanho próximo a 400MB.
Passos:
Cliente da Máquina A requisita o upload de um arquivo com tamanho próximo a 400MB.
Cliente da Máquina B requisita o upload de outro arquivo com tamanho próximo a 400MB.
Compara os arquivos encontrados no diretório do servidor, no diretório do cliente da Máquina A e no diretório do cliente da Máquina B.
Resultado esperado: Ambos os clientes recebem respostas de sucesso do servidor. Ambos os arquivos e seus respectivos dados devem estar corretamente presentes no diretório do servidor.
Resultado Obtido: Ambos os clientes receberam respostas de sucesso do servidor. O arquivo enviado pelo Cliente da Máquina A foi transferido corretamente para o diretório do servidor. O arquivo enviado pelo Cliente da Máquina B também foi transferido corretamente para o servidor.
Terminal do Cliente da Máquina A:

```
> upload
Filename: ivan1.mp4
File exists: ./client_files/ivan1.mp4
{"status": "success", "message": "Receiving file..."}
```

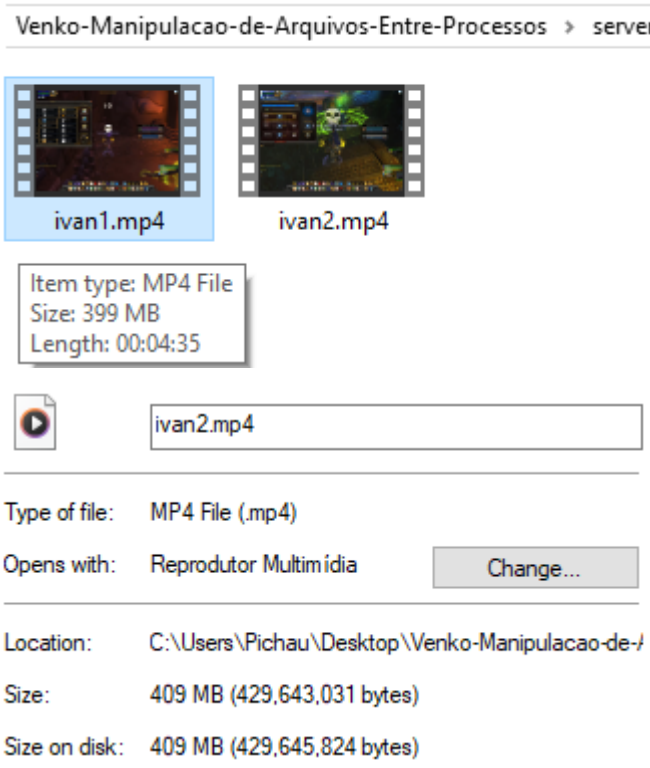
Terminal do Cliente da Máquina B:

```
> upload
Filename: ivan2.mp4
File exists: ./client_files/ivan2.mp4
{"status": "success", "message": "Receiving file..."}
```

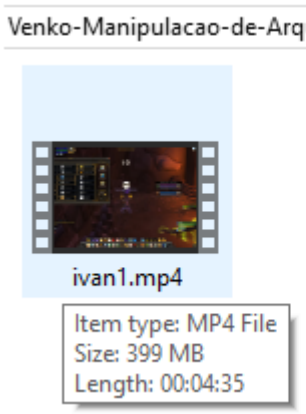
Terminal do Servidor:

```
JSON_Request:
{"tipo_requisicao": "UPLOAD", "nome_arquivo": "ivan1.mp4"}
Receiving file 'ivan1.mp4'...
JSON_Request:
{"tipo_requisicao": "UPLOAD", "nome_arquivo": "ivan2.mp4"}
Receiving file 'ivan2.mp4'...
Transfer of file 'ivan1.mp4' completed.
Transfer of file 'ivan2.mp4' completed.
```

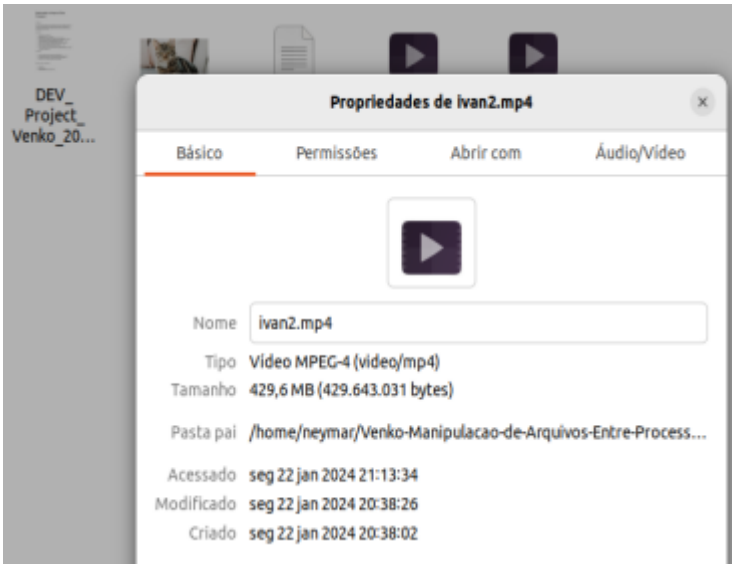
Diretório do Servidor, contendo os dois arquivos e seus respectivos dados corretamente:



Arquivos originais:
Diretório do Cliente da Máquina A:



Diretório do Cliente da Máquina B:



Teste 16: Dois clientes fazem simultaneamente um upload de um arquivo com tamanho próximo a 400MB. O arquivo contém o mesmo nome, porém dados diferentes.
Cenário: O cliente da Máquina A requisita o upload de um arquivo com tamanho próximo a 400MB. Poucos milisegundos depois, cliente da Máquina B requisita o upload de outro arquivo com tamanho um pouco superior a 400MB, porém com o mesmo nome.
Passos:
Cliente da Máquina A requisita o upload de um arquivo com tamanho próximo a 400MB.
Cliente da Máquina B requisita o upload de um arquivo diferente, porém de mesmo nome, com tamanho um pouco superior a 400MB.
Compara o arquivo encontrado no diretório do servidor ao final das operações com os arquivos que foram enviados por cada cliente.
Resultado esperado: Ambos os clientes recebem respostas de sucesso do servidor. Como os arquivos possuem o mesmo nome, ao final da operação, se espera que o arquivo enviado pela Máquina B – que fez a requisição poucos milissegundos mais tarde, que transfere um arquivo de tamanho um pouco maior e que possui uma taxa de transmissão mais lenta – deve ser o único arquivo disponível no diretório do servidor, por ter sobrescrito

o arquivo de mesmo nome que foi enviado simultaneamente ao servidor pelo cliente da Máquina A.
Resultado Obtido: Ambos os clientes receberam respostas de sucesso do servidor. Ao final da operação, apenas o arquivo que foi enviado pelo cliente da Máquina B pode ser encontrado no servidor. O arquivo encontrado no diretório do servidor possui dados iguais ao arquivo encontrado no diretório do cliente da Máquina B.
Terminal do Cliente da Máquina A:

```
> upload
Filename: 7.mp4
File exists: ./client_files/7.mp4
{"status": "success", "message": "Receiving file..."}
```

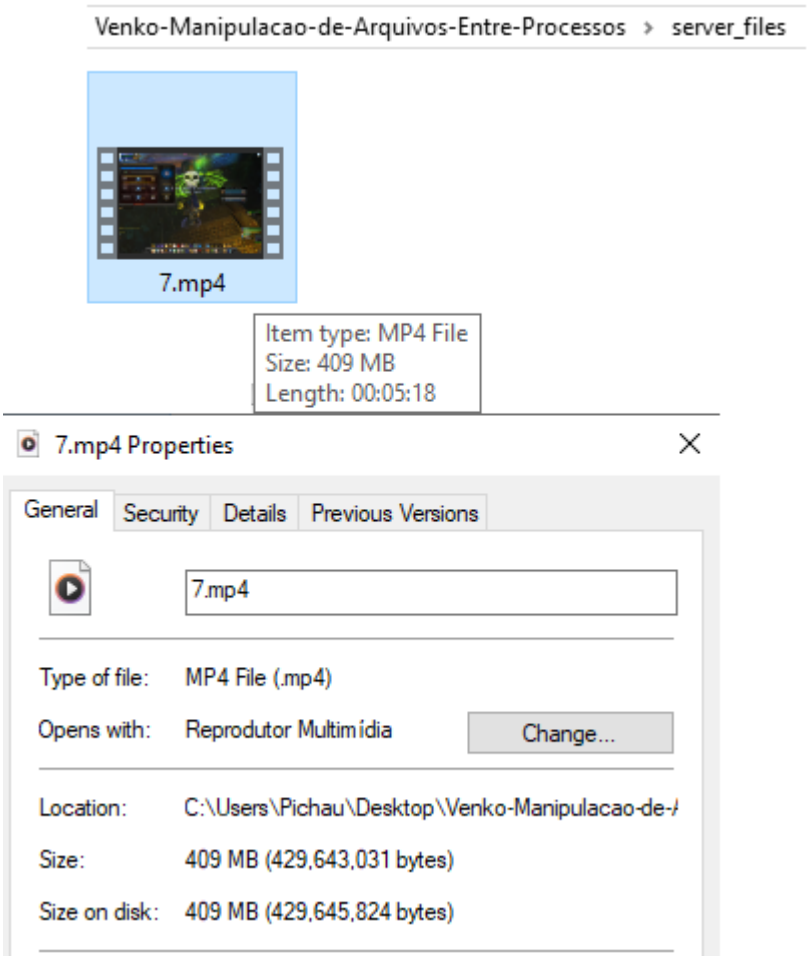
Terminal do Ciente da Máquina B:

```
> upload
Filename: 7.mp4
File exists: ./client_files/7.mp4
{"status": "success", "message": "Receiving file..."}
```

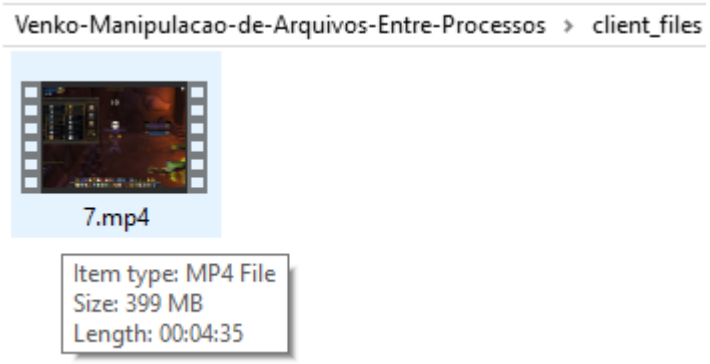
Terminal do Servidor:

```
JSON_Request:
{"tipo_requisicao": "UPLOAD", "nome_arquivo": "7.mp4"}
Receiving file '7.mp4'...
JSON_Request:
{"tipo_requisicao": "UPLOAD", "nome_arquivo": "7.mp4"}
Receiving file '7.mp4'...
Transfer of file '7.mp4' completed.
Transfer of file '7.mp4' completed.
```

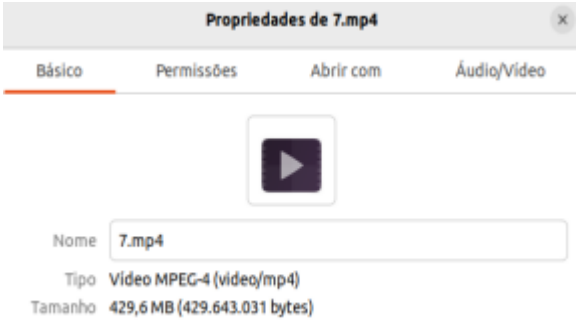
Diretório do Servidor, contendo o arquivo que foi enviado pelo cliente da Máquina B:



Arquivo que havia sido enviado pelo cliente da Máquina A:



Arquivo que havia sido enviado pelo cliente da Máquina B:



Teste 17: Dois clientes fazem simultaneamente download de arquivos diferentes do servidor.
Cenário: O cliente da Máquina A requisita o download de um arquivo. Poucos milisegundos depois, o cliente da Máquina B requisita o download de outro arquivo.
Passos:
Cliente da Máquina A requisita o download de um arquivo.
Cliente da Máquina B requisita o download de outro arquivo.
Compara os arquivos encontrados no diretório do servidor, no diretório do cliente da Máquina A e no diretório do cliente da Máquina B.
Resultado esperado: Ambos os clientes recebem respostas de sucesso do servidor. O diretório do cliente da Máquina A deve ter recebido corretamente os dados do arquivo requisitado ao servidor. O diretório do cliente da Máquina B também deve ter recebido corretamente os dados do arquivo requisitado ao servidor.
Resultado Obtido: Ambos os clientes receberam respostas de sucesso do servidor. O diretório do cliente da Máquina A recebeu corretamente os dados do arquivo requisitado ao servidor, O diretório do cliente da Máquina B também recebeu corretamente os dados do arquivo requisitado ao servidor.
Terminal do Cliente da Máquina A:

```
> download
Filename: ivan.jpg
{"status": "success", "message": "File found. Transference started."}
Download of file 'ivan.jpg' completed. Saved in './client_files/'.
```

Terminal do Cliente da Máquina B:

```
> download
Filename: ivan.gif
{"status": "success", "message": "File found. Transference started."}
Download of file 'ivan.gif' completed. Saved in './client_files/'.
```

Terminal do Servidor:

```
JSON_Request:
{"tipo_requisicao": "DOWNLOAD", "nome_arquivo": "ivan.gif"}
File exists: ./server_files/ivan.gif
Transfer of 'ivan.gif' completed.
JSON_Request:
{"tipo_requisicao": "DOWNLOAD", "nome_arquivo": "ivan.jpg"}
File exists: ./server_files/ivan.jpg
Transfer of 'ivan.jpg' completed.
```

Teste 18: Dois clientes fazem simultaneamente download de um mesmo arquivo do servidor, de tamanho de 135KB.
Cenário: O cliente da Máquina A requisita o download de um arquivo com tamanho de 135KB. Poucos milisegundos depois, o cliente da Máquina B requisita o download do mesmo arquivo.
Passos:
Cliente da Máquina A requisita o download de um arquivo com tamanho de 135KB.
Cliente da Máquina B requisita o download do mesmo arquivo.
Compara os arquivos encontrados no diretório do servidor, no diretório do cliente da Máquina A e no diretório do cliente da Máquina B.

Resultado esperado: Ambos os clientes recebem respostas de sucesso do servidor. O diretório do cliente da Máquina A deve ter recebido corretamente os dados do arquivo requisitado ao servidor. O diretório do cliente da Máquina B também deve ter recebido corretamente os dados do arquivo requisitado ao servidor.

Resultado Obtido: Idem ao teste 19.

Teste 19: Dois clientes fazem simultaneamente download de um mesmo arquivo do servidor, de tamanho próximo a 400MB
Cenário: O cliente da Máquina A requisita o download de um arquivo com tamanho próximo a 400MB. Poucos milisegundos depois, o cliente da Máquina B requisita o download do mesmo arquivo.
Passos:
Cliente da Máquina A requisita o download de um arquivo com tamanho próximo a 400MB.
Cliente da Máquina B requisita o download do mesmo arquivo.
Compara os arquivos encontrados no diretório do servidor, no diretório do cliente da Máquina A e no diretório do cliente da Máquina B.
Resultado esperado: Ambos os clientes recebem respostas de sucesso do servidor. O diretório do cliente da Máquina A deve ter recebido corretamente os dados do arquivo requisitado ao servidor. O diretório do cliente da Máquina B também deve ter recebido corretamente os dados do arquivo requisitado ao servidor.

Resultado Obtido: O Cliente da Máquina A recebeu resposta de sucesso do servidor. O diretório do cliente da Máquina A recebeu corretamente os dados do arquivo requisitado ao servidor. O Cliente da Máquina B recebeu um erro inesperado relacionado ao tentar decodificar uma sequência de bytes usando UTF-8, onde encontrou um byte que não é válido no início de um caractere UTF-8.
Terminal do Cliente da Máquina A:

```
> download
Filename: 7.mp4
{"status": "success", "message": "File found. Transference started."}
Download of file '7.mp4' completed. Saved in './client_files/'.
```

Terminal do Cliente da Máquina B:

```
50      json_response = client_socket.recv(BUFFER_SIZE).decode()

Exception has occurred: UnicodeDecodeError ×
'utf-8' codec can't decode byte 0xa7 in position 102: invalid start byte
  File "/home/neymar/Venko-Manipulacao-de-Arquivos-Entre-Processos-main/client.py", line 50, in download_file
    json_response = client_socket.recv(BUFFER_SIZE).decode()
  File "/home/neymar/Venko-Manipulacao-de-Arquivos-Entre-Processos-main/client.py", line 130, in run_command
    download_file(client_socket, filename, CLIENT_DIR)
  File "/home/neymar/Venko-Manipulacao-de-Arquivos-Entre-Processos-main/client.py", line 158, in main
    run_command(client_socket, user_input.lower())
  File "/home/neymar/Venko-Manipulacao-de-Arquivos-Entre-Processos-main/client.py", line 163, in <module>
    main()
UnicodeDecodeError: 'utf-8' codec can't decode byte 0xa7 in position 102: invalid start byte
```

Terminal do Servidor:

```
JSON_Request:
{"tipo_requisicao": "DOWNLOAD", "nome_arquivo": "7.mp4"}
File exists: ./server_files/7.mp4
JSON_Request:
{"tipo_requisicao": "DOWNLOAD", "nome_arquivo": "7.mp4"}
File exists: ./server_files/7.mp4
```

Conclusão

A inicialização do servidor e do cliente demonstrou comportamento esperado, gerando mensagens de sucesso e erro conforme previsto nos cenários específicos. As operações de listagem, deleção, upload e download de arquivos foram bem-sucedidas, evidenciando a capacidade do sistema de gerenciar eficientemente as requisições do usuário.

Os testes de conexão simultânea de clientes corroboraram a estabilidade do sistema, assegurando que múltiplos usuários possam interagir de forma eficaz. A capacidade de lidar com upload simultâneo de arquivos, mesmo com nomes idênticos, retornou o comportamento esperado pela atual implementação.

No momento, os testes 18 e 19 falharam. É necessário alterações no código a fim de obter o resultado esperado.

Em síntese, os resultados obtidos até o momento indicam – com exceção dos testes 18 e 19, que tratam de download simultâneo de um mesmo arquivo – que o sistema está em conformidade com as expectativas e requisitos estabelecidos, proporcionando uma base sólida para futuras implementações e melhorias.