

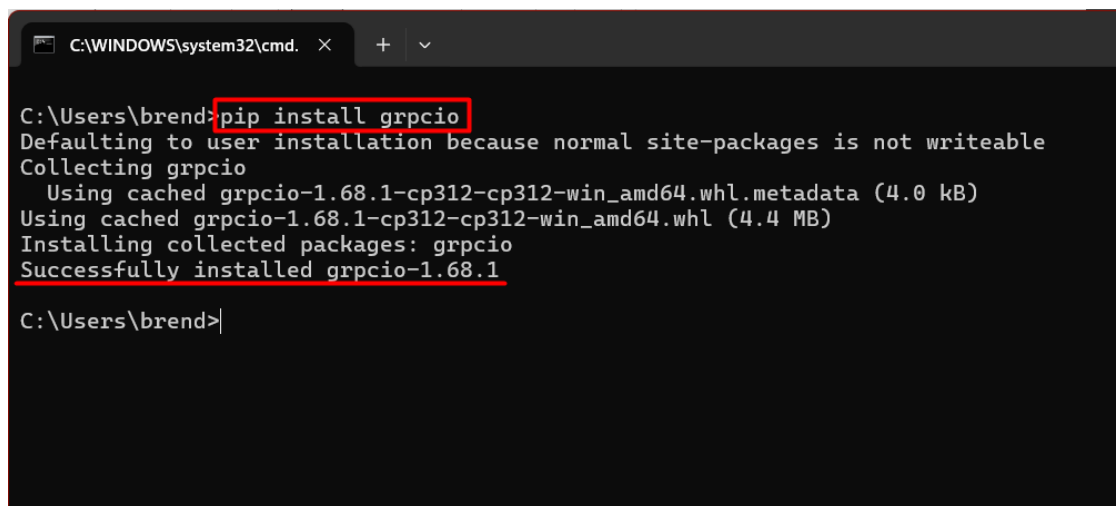
Desenvolvimento de um Sistema Distribuído com Comunicação Cliente-Servidor usando Sockets e RPC/gRPC

SIMULAÇÃO SISTEMA DE ESTOQUE DE HORTIFRUTI

Brenda Martinez
Análise e Desenvolvimento de Sistemas
dezembro/2024

Overview

- Essa documentação foi produzida utilizando o sistema operacional WINDOWS.
- Linguagem: Python 3
- Bibliotecas: socket, grpc, concurrent
- Instalando uma biblioteca python (Windows):
 - Necessário ter o python previamente instalado
 - Abrir o CMD e executar o comando “`pip install {nome da biblioteca}`”

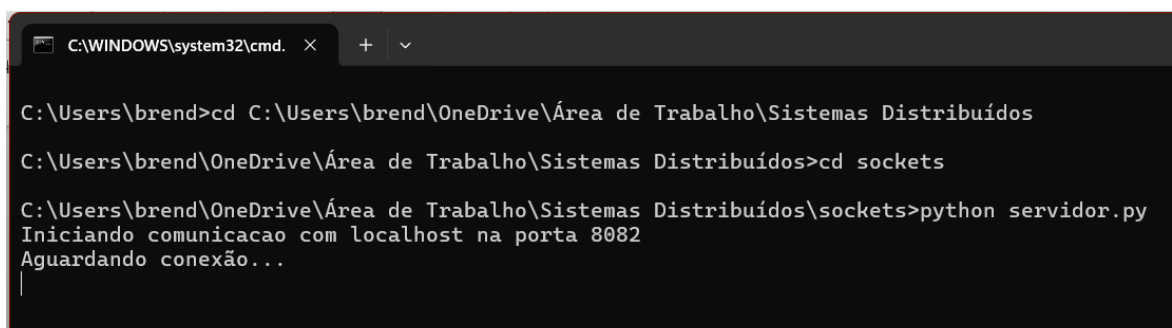


```
C:\WINDOWS\system32\cmd. X + v
C:\Users\brend>pip install grpcio
Defaulting to user installation because normal site-packages is not writeable
Collecting grpcio
  Using cached grpcio-1.68.1-cp312-cp312-win_amd64.whl.metadata (4.0 kB)
Using cached grpcio-1.68.1-cp312-cp312-win_amd64.whl (4.4 MB)
Installing collected packages: grpcio
Successfully installed grpcio-1.68.1
C:\Users\brend>
```

Como Executar o Sistema (Windows)

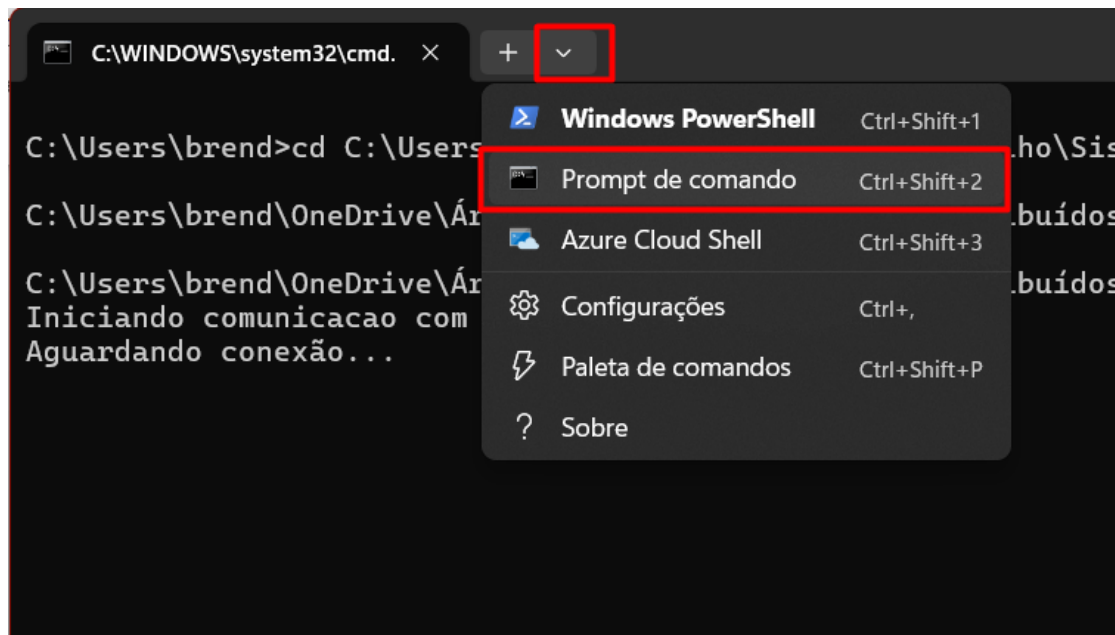
Implementação utilizando Sockets:

1. Abra o CMD do Windows e entre na pasta onde o arquivo do servidor está localizado.
2. Execute o servidor utilizando o comando “`python servidor.py`”

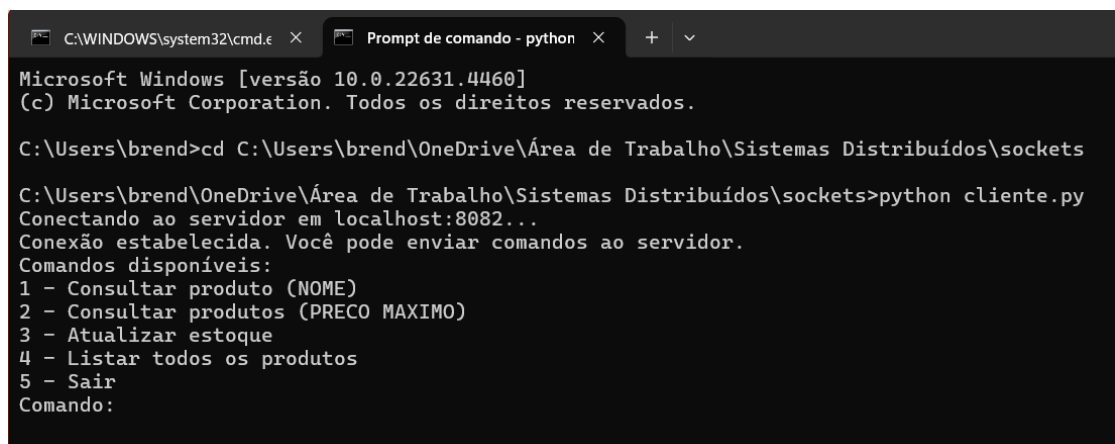


```
C:\WINDOWS\system32\cmd. X + v
C:\Users\brend>cd C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos>cd sockets
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\sockets>python servidor.py
Iniciando comunicacao com localhost na porta 8082
Aguardando conexão...
|
```

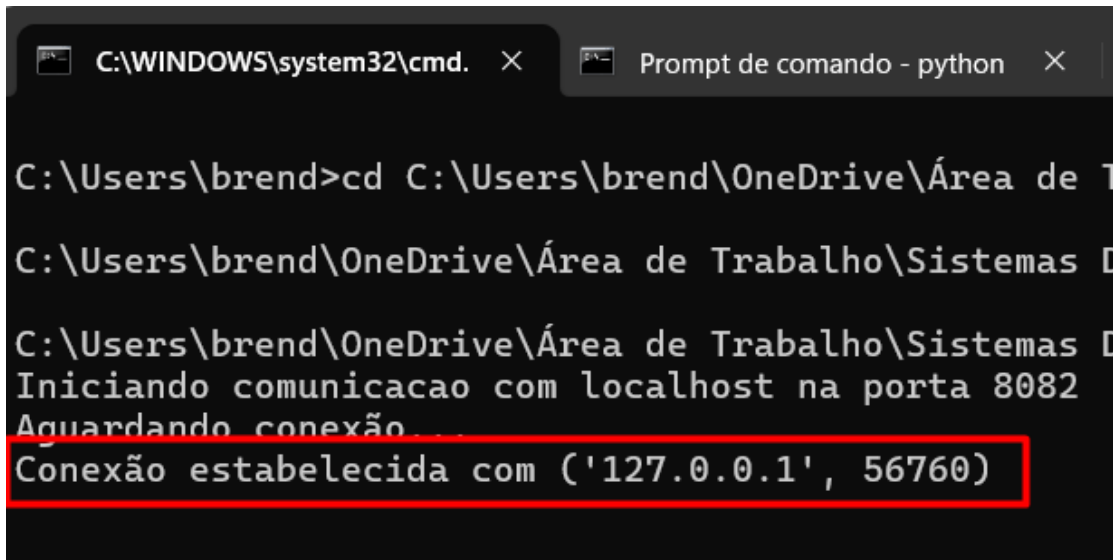
- Abra uma nova guia no CMD do Windows e entre na pasta onde o arquivo do cliente está localizado.



- Execute o cliente utilizando o comando “python cliente.py”

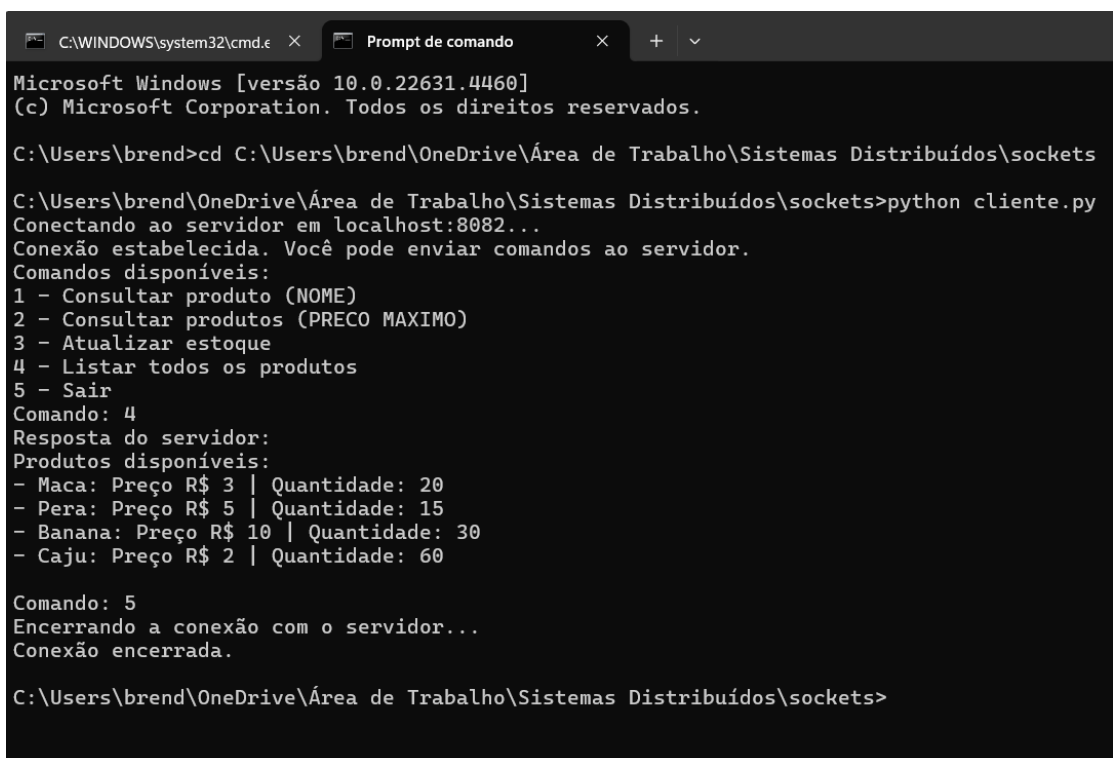


- Note que assim que o cliente é executado, o servidor retorna uma mensagem contendo o IP e a porta em que a conexão foi estabelecida.



```
C:\Users\brend>cd C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\sockets
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\sockets>python cliente.py
Conectando ao servidor em localhost:8082...
Aguardando conexão...
Conexão estabelecida com ('127.0.0.1', 56760)
```

6. Para interagir com o servidor, digite o comando desejado e aperte “Enter”. Para encerrar a conexão, digite “5” e aperte “Enter”.



```
Microsoft Windows [versão 10.0.22631.4460]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\brend>cd C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\sockets
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\sockets>python cliente.py
Conectando ao servidor em localhost:8082...
Conexão estabelecida. Você pode enviar comandos ao servidor.
Comandos disponíveis:
1 - Consultar produto (NOME)
2 - Consultar produtos (PREÇO MÁXIMO)
3 - Atualizar estoque
4 - Listar todos os produtos
5 - Sair
Comando: 4
Resposta do servidor:
Produtos disponíveis:
- Maca: Preço R$ 3 | Quantidade: 20
- Pera: Preço R$ 5 | Quantidade: 15
- Banana: Preço R$ 10 | Quantidade: 30
- Caju: Preço R$ 2 | Quantidade: 60
Comando: 5
Encerrando a conexão com o servidor...
Conexão encerrada.

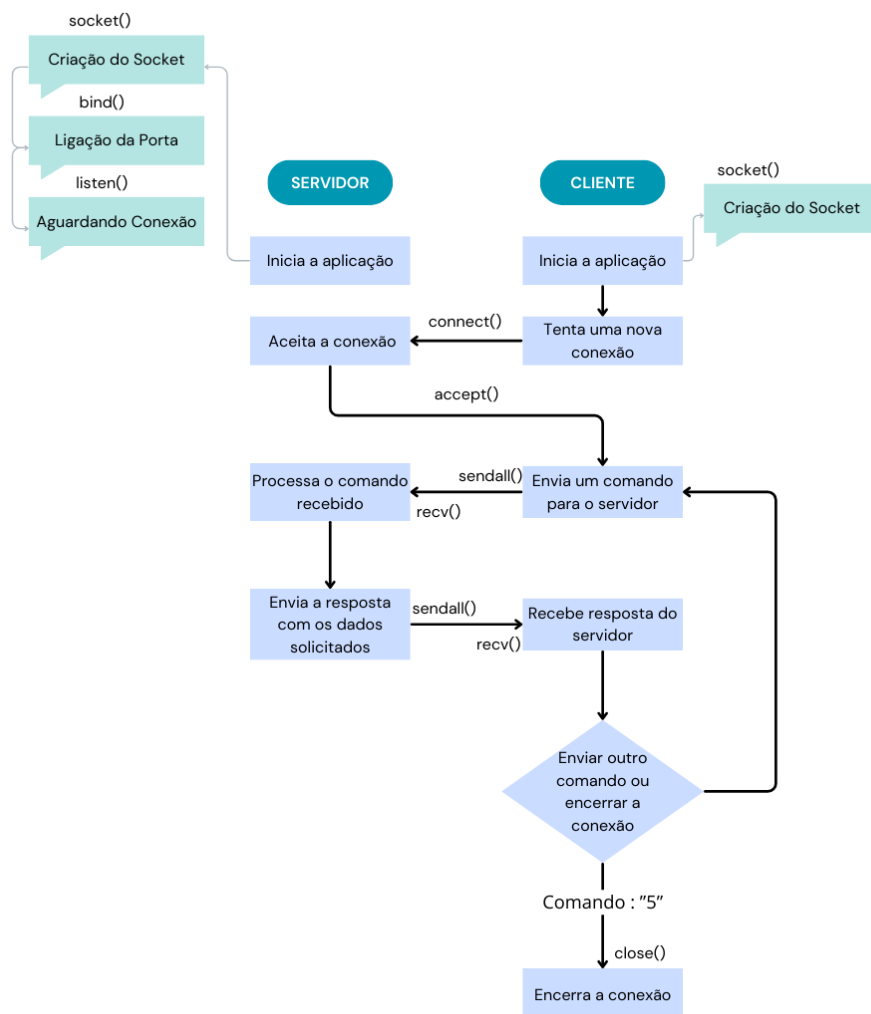
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\sockets>
```

7. Após 3 conexões, o servidor é fechado automaticamente.

```
C:\WINDOWS\system32\cmd. X Prompt de comando X Prompt de comando X + v
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\sockets>python servidor.py
Iniciando comunicacao com localhost na porta 8082
Aguardando conexão...
Conexão estabelecida com ('127.0.0.1', 56966)
Comando recebido: 2 8
Comando recebido: 4
Aguardando conexão...
Conexão estabelecida com ('127.0.0.1', 56967)
Comando recebido: 3 Caju -20
Aguardando conexão...
Conexão estabelecida com ('127.0.0.1', 56968)
Comando recebido: 4
Numero max de conexoes atingidas.

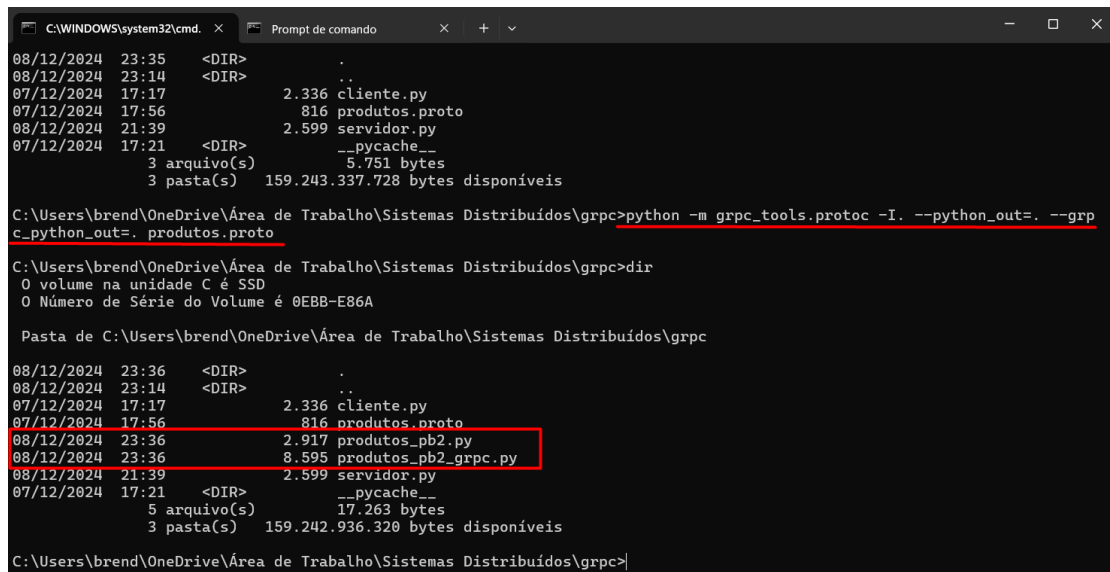
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\sockets>
```

FLUXOGRAMA



Implementação com gRPC:

1. Abra o CMD do Windows e entre na pasta onde o arquivo “produtos.proto” está localizado.
2. Digite o comando “`python -m grpc_tools.protoc -I. --python_out=. --grpc_python_out=. produtos.proto`” para gerar os arquivos contendo o código gRPC (caso os mesmos já tenham sido gerados, ignorar esse passo, ou excluir os arquivos e executar o comando para gerar novamente).



```
C:\WINDOWS\system32\cmd. x Prompt de comando
08/12/2024 23:35 <DIR> .
08/12/2024 23:14 <DIR> ..
07/12/2024 17:17 2.336 cliente.py
07/12/2024 17:56 816 produtos.proto
08/12/2024 21:39 2.599 servidor.py
07/12/2024 17:21 <DIR> __pycache__
3 arquivo(s) 5.751 bytes
3 pasta(s) 159.243.337.728 bytes disponíveis

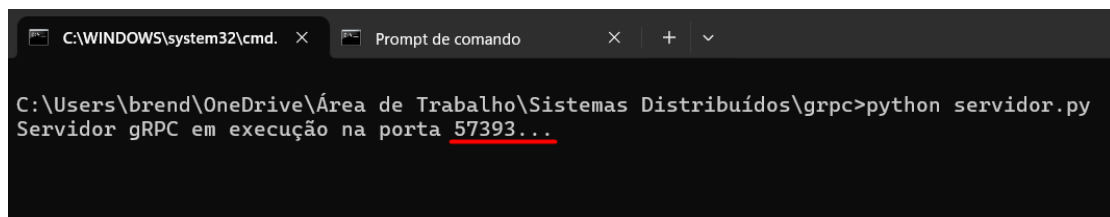
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\grpc>python -m grpc_tools.protoc -I. --python_out=. --grpc_python_out=. produtos.proto

C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\grpc>dir
O volume na unidade C é SSD
O Número de Série do Volume é 0EBB-E86A

Pasta de C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\grpc
08/12/2024 23:36 <DIR> .
08/12/2024 23:14 <DIR> ..
07/12/2024 17:17 2.336 cliente.py
07/12/2024 17:56 816 produtos.proto
08/12/2024 23:36 2.917 produtos_pb2.py
08/12/2024 23:36 8.595 produtos_pb2_grpc.py
08/12/2024 21:39 2.599 servidor.py
07/12/2024 17:21 <DIR> __pycache__
5 arquivo(s) 17.263 bytes
3 pasta(s) 159.242.936.320 bytes disponíveis

C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\grpc>
```

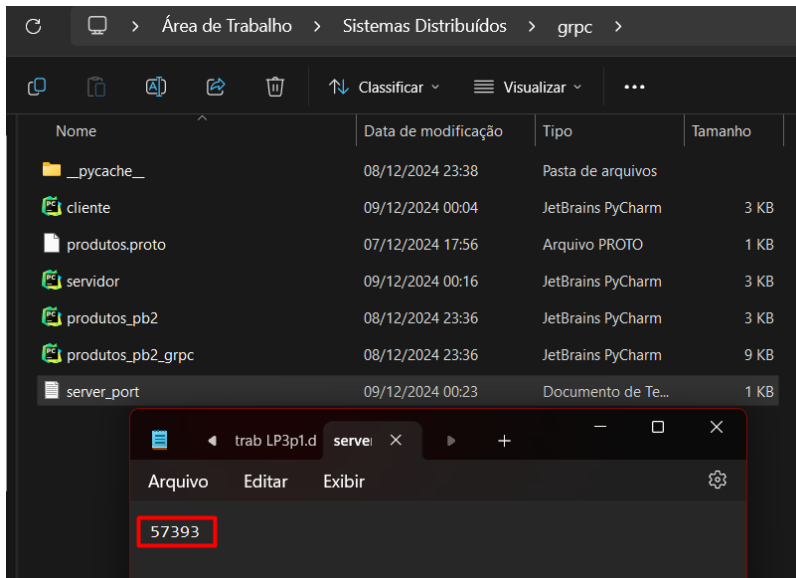
3. Execute o servidor utilizando o comando “`python servidor.py`”



```
C:\WINDOWS\system32\cmd. x Prompt de comando

C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\grpc>python servidor.py
Servidor gRPC em execução na porta 57393...
```

Ao executar o programa, ele irá gerar um arquivo de texto contendo a porta em que a conexão foi estabelecida.



4. Abra uma nova guia no CMD do Windows e entre na pasta onde o arquivo do cliente está localizado (conforme ensinado anteriormente).
5. Execute o cliente utilizando o comando “`python cliente.py`”.

```
C:\WINDOWS\system32\cmd.exe x Prompt de comando - python x + v
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\grpc>python cliente.py
Conectando ao servidor em localhost:57393...
Conexão estabelecida. Você pode enviar comandos ao servidor.
Comandos disponíveis:
1 - Consultar produto (NOME)
2 - Consultar produtos (PREÇO MÁXIMO)
3 - Atualizar estoque
4 - Listar todos os produtos
5 - Sair
Digite sua escolha:
```

6. Para interagir com o servidor, digite o comando desejado e aperte “Enter”. Para encerrar a conexão, digite “5” e aperte “Enter”.

```

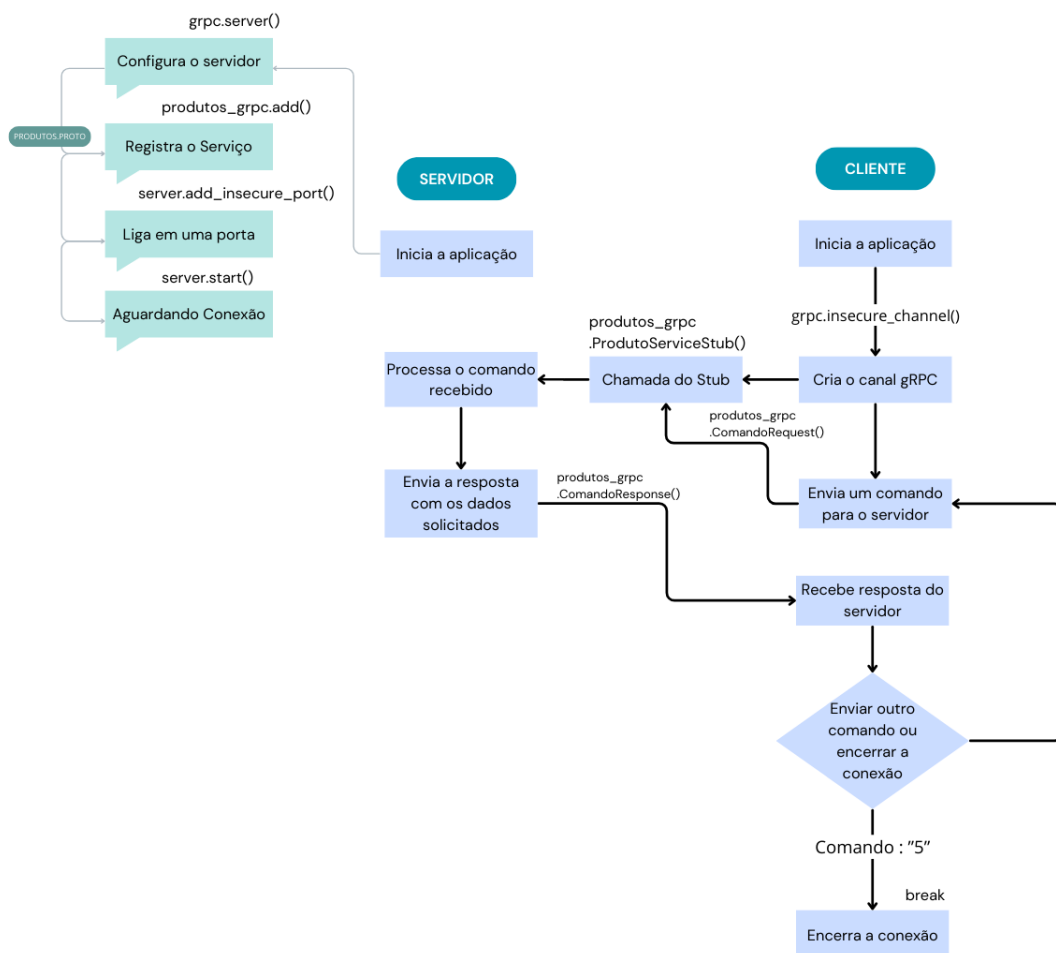
C:\WINDOWS\system32\cmd.exe x Prompt de comando x + v
C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\grpc>python cliente.py
Conectando ao servidor em localhost:57393...
Conexão estabelecida. Você pode enviar comandos ao servidor.
Comandos disponíveis:
1 - Consultar produto (NOME)
2 - Consultar produtos (PREÇO MÁXIMO)
3 - Atualizar estoque
4 - Listar todos os produtos
5 - Sair
Digite sua escolha: 4
Maca: R$3.0, Estoque: 20
Pera: R$5.0, Estoque: 15
Banana: R$10.0, Estoque: 30
Caju: R$2.0, Estoque: 60
Digite sua escolha: 5
Encerrando o cliente...

C:\Users\brend\OneDrive\Área de Trabalho\Sistemas Distribuídos\grpc>

```

O servidor continuará aberto para novas conexões utilizando a mesma porta, conforme registro no arquivo “server_port.txt”.

FLUXOGRAMA



Comparação entre as abordagens Sockets/gRPC:

A abordagem em sockets e a abordagem em gRPC apresentam diferenças significativas em termos de implementação, manutenção e eficiência, tanto do ponto de vista do servidor quanto do cliente. Ao utilizar sockets, a comunicação ocorre de forma direta via protocolo TCP, onde o cliente envia mensagens em texto simples, e o servidor interpreta, processa e retorna as respostas. Essa abordagem oferece flexibilidade e baixo nível de abstração, permitindo ao desenvolvedor controlar completamente o protocolo de comunicação. Contudo, essa liberdade vem acompanhada de desafios, como a necessidade de gerenciar manualmente conexões e interpretar mensagens. Essas tarefas tornam o código mais propenso a erros e difícil de manter, especialmente em sistemas que precisam escalar ou que possuem múltiplos clientes simultâneos.

Por outro lado, a abordagem com gRPC, baseada em Remote Procedure Call, utiliza o formato Protocol Buffers para serializar e desserializar mensagens. Isso abstrai muitos detalhes da comunicação, permitindo que o desenvolvedor foque na lógica da aplicação. No lado do servidor, o gRPC facilita o gerenciamento de múltiplas conexões, garantindo eficiência e segurança com suporte nativo a TLS. Além disso, o gRPC permite definir interfaces de serviço de forma declarativa, garantindo consistência e simplificando a manutenção. Do ponto de vista do cliente, a comunicação com o servidor é simplificada, já que as chamadas ao servidor se assemelham a chamadas de métodos locais, graças ao uso de stubs gerados automaticamente pelo Protobuf.

Enquanto sockets são mais adequados para implementações simples, gRPC é mais indicado para sistemas robustos e escaláveis. A abordagem em sockets demanda mais trabalho manual para implementar funcionalidades básicas, como a validação de mensagens e o suporte a múltiplos clientes, o que pode se tornar um obstáculo em projetos maiores. Por outro lado, o gRPC introduz dependências adicionais e requer aprendizado de ferramentas como Protobuf, mas compensa com uma interface clara, suporte nativo a múltiplas linguagens e um formato de mensagens mais eficiente.