TRANSMISSÃO DE DADOS COM GRPC

Anna Julia
Larissa Samara
Yasmim Fernandes

CLIENT.PY

```python
import grpc
import file_transfer_pb2
import file_transfer_pb2_grpc

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
def upload_file(stub, file_path):
    with open(file_path, 'rb') as f:
        content = f.read()
        filename = file_path.split('/')[-1]
        request = file_transfer_pb2.FileRequest(filename=filename, content=content)
        response = stub.UploadFile(request)
        print(response.message)

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
def main():
    channel = grpc.insecure_channel('localhost:50051')
    stub = file_transfer_pb2_grpc.FileTransferStub(channel)
    file_path = input("Entre com o nome do arquivo .txt para fazer o upload: ")
    upload_file(stub, f"files/{file_path}")

if __name__ == "__main__":
    main()

```

- O programa conecta a um servidor gRPC.
- O usuário informa o nome de um arquivo .txt.
- O programa lê o arquivo e o envia ao servidor.
- O servidor retorna uma mensagem de confirmação, exibida no terminal.

PROTO

```
syntax = "proto3";

service FileTransfer {
  rpc UploadFile (FileRequest) returns (FileResponse);
}

message FileRequest {
  string filename = 1;
  bytes content = 2;
}

message FileResponse {
  string message = 1;
}
```

- O cliente usa o serviço FileTransfer e chama o método UploadFile.
- O cliente envia uma mensagem FileRequest com:
  O nome do arquivo.
  O conteúdo do arquivo em binário.
- O servidor processa essa solicitação e responde com uma mensagem FileResponse contendo uma confirmação ou status.

SERVER.JS

```
const grpc = require('@grpc/grpc-js');
const protoLoader = require('@grpc/proto-loader');
const fs = require('fs');

const PROTO_PATH = './file_transfer.proto';
const packageDefinition = protoLoader.loadSync(PROTO_PATH);
const {FileTransfer} = grpc.loadPackageDefinition(packageDefinition);

Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery
const uploadFile = (call, callback) => {
    const { filename, content } = call.request;

    fs.writeFileSync(`./files/upload-${filename}`, content);
    callback(null, { message: `Arquivo upload-${filename} carregado com sucesso!` });
};

const server = new grpc.Server();
server.addService(FileTransfer.service, { UploadFile: uploadFile });

const PORT = '50051';

server.bindAsync(`0.0.0.0:${PORT}`, grpc.ServerCredentials.createInsecure(), () => {
    console.log(`Servidor executando na porta ${PORT}`);
    server.start();
});
```

- O servidor é configurado para ouvir chamadas gRPC no método UploadFile.
- Quando um cliente chama o método:
- O servidor salva o arquivo recebido no diretório ./files/ com o prefixo upload-.
- Responde ao cliente com uma mensagem de sucesso.
- O servidor fica escutando na porta 50051 para receber novas requisições.

THANK YOU