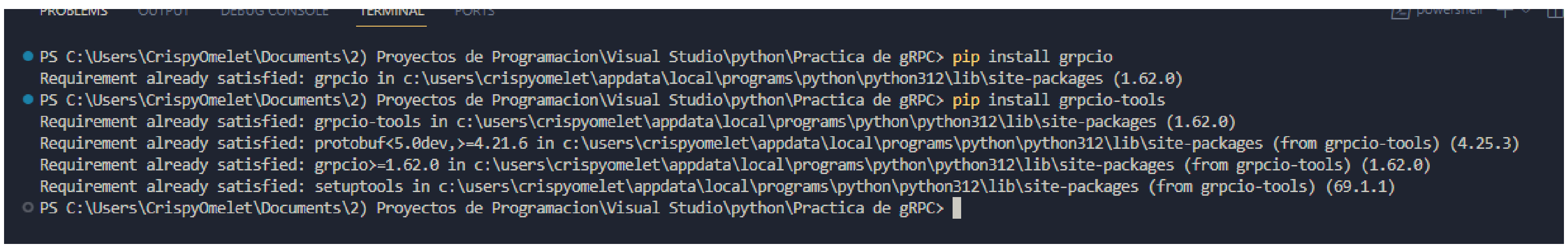
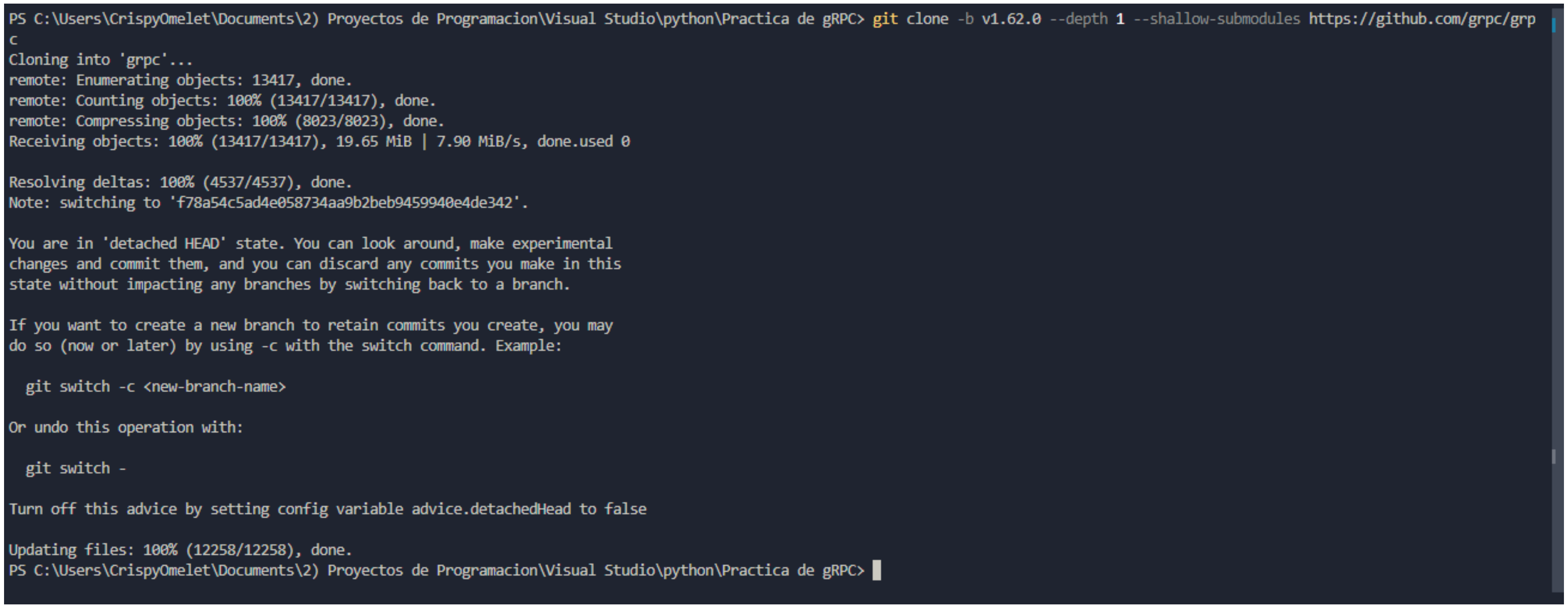
William José Chi Rico – 66211 – Octavo Semestre – Sistemas Distribuidos – Practica gRPC

Paso #1 – Instalar grpcio y grpcio-tools



```
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\Practica de gRPC> pip install grpcio
Requirement already satisfied: grpcio in c:\users\crispyomelet\appdata\local\programs\python\python312\lib\site-packages (1.62.0)
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\Practica de gRPC> pip install grpcio-tools
Requirement already satisfied: grpcio-tools in c:\users\crispyomelet\appdata\local\programs\python\python312\lib\site-packages (1.62.0)
Requirement already satisfied: protobuf<5.0dev,>=4.21.6 in c:\users\crispyomelet\appdata\local\programs\python\python312\lib\site-packages (from grpcio-tools) (4.25.3)
Requirement already satisfied: grpcio>=1.62.0 in c:\users\crispyomelet\appdata\local\programs\python\python312\lib\site-packages (from grpcio-tools) (1.62.0)
Requirement already satisfied: setuptools in c:\users\crispyomelet\appdata\local\programs\python\python312\lib\site-packages (from grpcio-tools) (69.1.1)
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\Practica de gRPC>
```

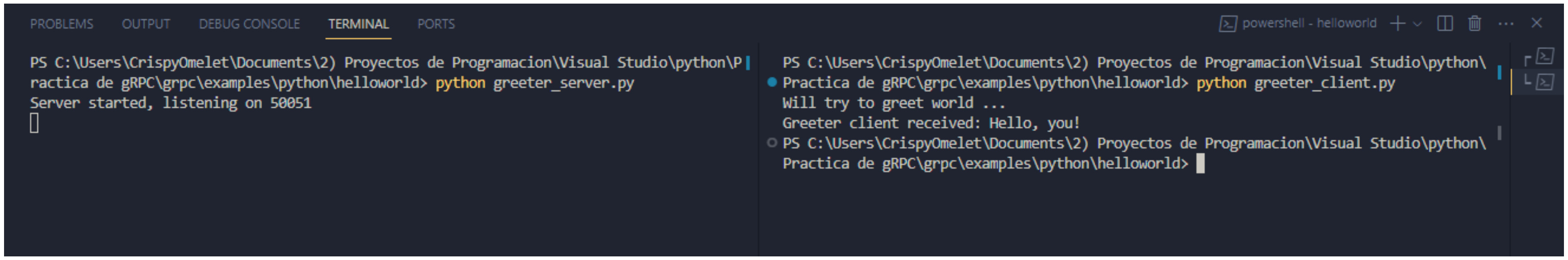
Paso #2 – Clonar el repositorio de github



```
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\Practica de gRPC> git clone -b v1.62.0 --depth 1 --shallow-submodules https://github.com/grpc/grp
c
Cloning into 'grpc'...
remote: Enumerating objects: 13417, done.
remote: Counting objects: 100% (13417/13417), done.
remote: Compressing objects: 100% (8023/8023), done.
Receiving objects: 100% (13417/13417), 19.65 MiB | 7.90 MiB/s, done.used 0

Resolving deltas: 100% (4537/4537), done.
Note: switching to 'f78a54c5ad4e058734aa9b2beb9459940e4de342'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

Updating files: 100% (12258/12258), done.
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\Practica de gRPC>
```

William José Chi Rico – 66211 – Octavo Semestre – Sistemas Distribuidos – Practica gRPC

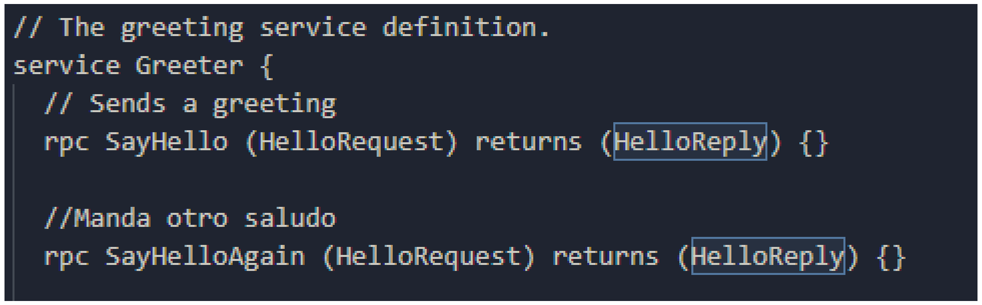
Paso #3 – Iniciar el servidor (lado izquierdo) y el cliente (lado derecho) del programa “greeter”



```
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\P
ractica de gRPC\grpc\examples\python\helloworld> python greeter_server.py
Server started, listening on 50051
```

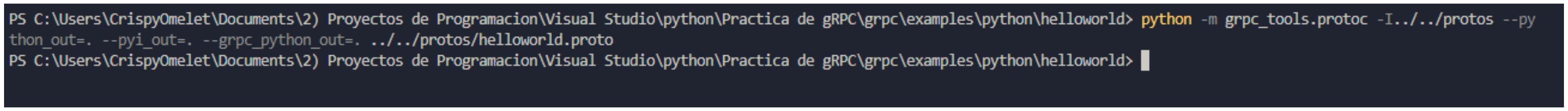
```
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\
Practica de gRPC\grpc\examples\python\helloworld> python greeter_client.py
Will try to greet world ...
Greeter client received: Hello, you!
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\
Practica de gRPC\grpc\examples\python\helloworld>
```

Paso #4 – Añadir otra función al archivo “helloworld.proto”



```
// The greeting service definition.
service Greeter {
  // Sends a greeting
  rpc SayHello (HelloRequest) returns (HelloReply) {}

  //Manda otro saludo
  rpc SayHelloAgain (HelloRequest) returns (HelloReply) {}
```

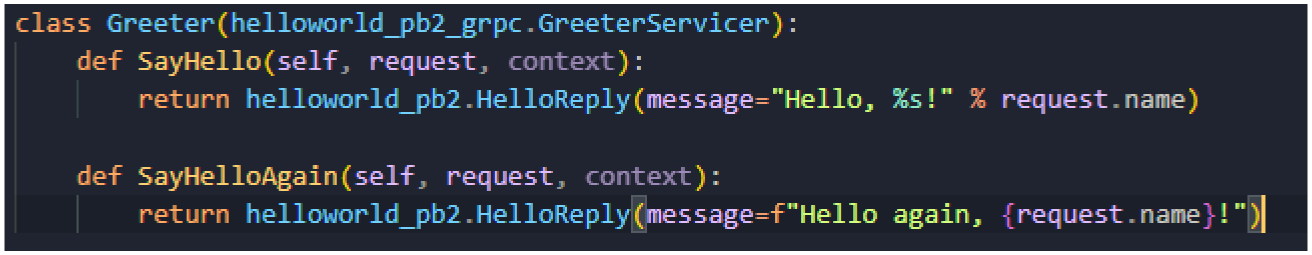
Paso #5 – Generar el código gRPC con el comando siguiente.



```
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\Practica de gRPC\grpc\examples\python\helloworld> python -m grpc_tools.protoc -I../../protos --py
thon_out=. --pyi_out=. --grpc_python_out=. ../../protos/helloworld.proto
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\Practica de gRPC\grpc\examples\python\helloworld>
```

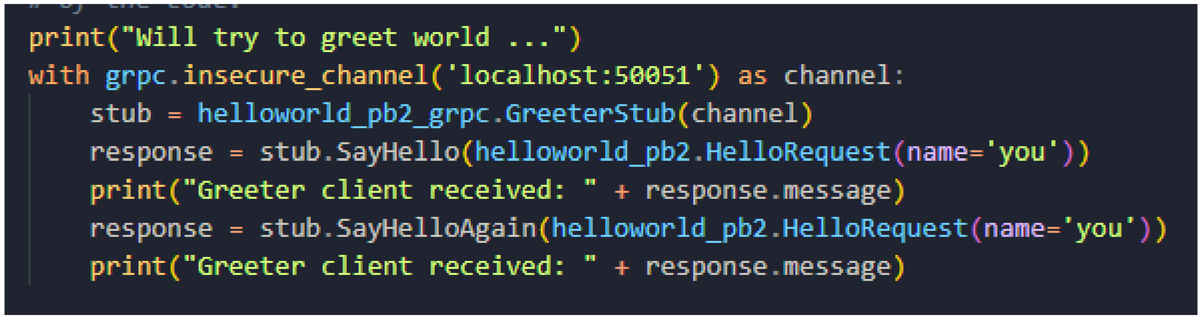
William José Chi Rico – 66211 – Octavo Semestre – Sistemas Distribuidos – Practica gRPC

Paso #6 – Añadirle la nueva función a la clase "Greeter" dentro del archivo "greeter_client"



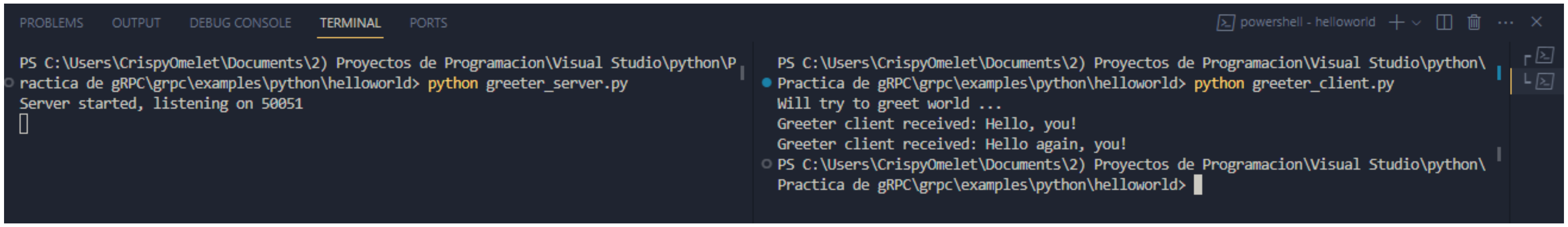
```python
class Greeter(helloworld_pb2_grpc.GreeterServicer):
    def SayHello(self, request, context):
        return helloworld_pb2.HelloReply(message="Hello, %s!" % request.name)

    def SayHelloAgain(self, request, context):
        return helloworld_pb2.HelloReply(message=f"Hello again, {request.name}!")
```

Paso #7 – Añadirle la respuesta y la impresión al archivo "greeter_server"



```python
print("Will try to greet world ...")
with grpc.insecure_channel('localhost:50051') as channel:
    stub = helloworld_pb2_grpc.GreeterStub(channel)
    response = stub.SayHello(helloworld_pb2.HelloRequest(name='you'))
    print("Greeter client received: " + response.message)
    response = stub.SayHelloAgain(helloworld_pb2.HelloRequest(name='you'))
    print("Greeter client received: " + response.message)
```

Paso #8 – Iniciar nuevamente el servidor (lado izquierdo) y el cliente (lado derecho) del programa "greeter"



```
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\P
ractica de gRPC\grpc\examples\python\helloworld> python greeter_server.py
Server started, listening on 50051
```

```
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\
Practica de gRPC\grpc\examples\python\helloworld> python greeter_client.py
Will try to greet world ...
Greeter client received: Hello, you!
Greeter client received: Hello again, you!
PS C:\Users\CrispyOmelet\Documents\2) Proyectos de Programacion\Visual Studio\python\
Practica de gRPC\grpc\examples\python\helloworld>
```