

# 实验内容



西安电子科技大学  
XIDIAN UNIVERSITY

计算机科学与技术学院  
School of Computer Science and Technology

## 第二次作业!

### ■ 题目:

利用RPC技术实现一个医院预约挂号系统，具体要求：

1. 客户端实现用户交互，服务器端实现预约信息的存储和管理。客户端与服务器端利用RPC机制进行协作。**中间件任选**。
2. 服务器端至少暴露如下RPC接口：
  - `bool bookAppointment(Appointment appointment)`: 预约挂号
  - `Appointment queryByID(int appointmentID)`: 根据预约 ID 查询预约信息
  - `List<Appointment> queryByPatient(String patientName)`: 根据患者姓名查询所有预约
  - `bool cancelAppointment(int appointmentID)`: 取消指定预约

Appointment对象应包含：预约 ID（唯一标识）、患者姓名、医生姓名、科室、预约日期、时间段等信息

### ■ 提交要求:

1. **4月20日前**将源程序打包通过**西电智课平台**提交
2. 打包文件命名方式：第2次作业+学号+姓名.zip

## 准备工作

我选择的语言是 Python，选用的中间件是 gRPC。

## 环境的配置

参考了 [快速入门 | Python | gRPC 框架](#)

## 写 .proto 文件

.proto 文件是 gRPC 的核心部分，它定义了通信使用的消息结构与RPC的接口方法。

根据实验内容，可以写出以下 .proto 文件。

我新加了一个叫做 `getStorageSize` 的接口，因为我认为在实际应用中，预约ID应当是服务端生成给客户端的，而最简单的ID生成方法就是从 1 开始挨个排，所以可以在服务端中定义一个全局的字典用来存储预约信息，每次给的 ID 就是字典内元素数量加 1。所以我们需要实现一个给客户端的接口用来获取这个服务端的全局的字典内的元素个数。

```

syntax = 'proto3';

message Appointment{
    int32 appointmentId = 1;
    string patientName = 2;
    string doctorName = 3;
    string department = 4;
    string date = 5;
    string timeSlot = 6;
}

message BoolResult{
    bool res = 1;
}

message AppointmentId{
    int32 appointmentId = 1;
}

message PatientName{
    string patientName = 1;
}

message AppointmentList{
    repeated Appointment appointments = 1;
}

message Empty{}

message StorageSize{
    int32 size = 1;
}

service AppointmentService{
    rpc bookAppointment(Appointment) returns (BoolResult);
    rpc queryById(AppointmentId) returns (Appointment);
    rpc queryByPatient(PatientName) returns (AppointmentList);
    rpc cancelAppointment(AppointmentId) returns (BoolResult);
    rpc getStorageSize(Empty) returns (StorageSize);
}

```

## 使用 protoc 生成 Python 代码

使用 cmd，切换到对应目录下，由于我的文件名为 HospitalAppointment.proto，所以在 cmd 中输入以下命令来生成 Python 代码。

```

python -m grpc_tools.protoc -I. --python_out=. --grpc_python_out=.
HospitalAppointment.proto

```

这会生成两个文件：

- HospitalAppointment\_pb2.py：包含消息类定义
- HospitalAppointment\_pb2\_grpc.py：包含服务的接口定义

## 编写服务端代码

创建 Python 文件 `server.py`，实现 `.proto` 文件中定义的接口，并启用 gRPC 服务监听端口。

这几个接口其实并不难写，每个接口都有 `request`, `context` 两个参数，而这个 `request` 就相当于客户端向服务端请求的数据，对于每个方法，这个数据的内容就是 `.proto` 文件里所定义的对应的 `message`。可以写出以下的 `Server.py`。

```
from concurrent import futures
import grpc
import HospitalAppointment_pb2
import HospitalAppointment_pb2_grpc

appointment_storage = {}

class
AppointmentServiceServicer(HospitalAppointment_pb2_grpc.AppointmentServiceSe
rvicer):
    # 预约挂号
    def bookAppointment(self, request, context):
        print("预约挂号的信息为: ")
        print(request)
        print("-----")
        BoolResult = getattr(HospitalAppointment_pb2, 'BoolResult')
        appointment_storage[request.appointmentId] = request
        return BoolResult(res=True)
    # 通过ID查询
    def queryById(self, request, context):
        print("查询的ID为: ")
        print(request)
        print("-----")
        Appointment = getattr(HospitalAppointment_pb2, 'Appointment')
        appointment = appointment_storage.get(request.appointmentId)
        if appointment is None:
            print("没有找到该挂号ID")
            return Appointment()
        else:
            return appointment
    # 通过病人姓名查询
    def queryByPatient(self, request, context):
        print("查询的病人姓名为: ")
        print(request)
```

```

print("-----")
patient_name = request.patientName
AppointmentList = getattr(HospitalAppointment_pb2,
'AppointmentList')
appointment_list = AppointmentList()
for appointment in appointment_storage.values():
    if appointment.patientName == patient_name:
        appointment_list.appointments.add(
            appointmentId = appointment.appointmentId,
            patientName = appointment.patientName,
            doctorName = appointment.doctorName,
            department = appointment.department,
            date = appointment.date,
            timeSlot = appointment.timeSlot
        )
if len(appointment_list.appointments) == 0:
    print("没有找到该病人挂的号")
    return appointment_list
else:
    return appointment_list

```

# 取消预约

```

def cancelAppointment(self, request, context):
    print("取消的挂号ID为: ")
    print(request)
    print("-----")
    BoolResult = getattr(HospitalAppointment_pb2, 'BoolResult')
    appointmentId = request.appointmentId
    if appointmentId in appointment_storage:
        del appointment_storage[appointmentId]
        return BoolResult(res=True)
    else:
        print("没有找到该挂号ID, 取消失败")

```

# 获取当前预约信息数量

```

def getStorageSize(self, request, context):
    StorageSize = getattr(HospitalAppointment_pb2, 'StorageSize')
    size = len(appointment_storage)
    return StorageSize(size=size)

```

```

def serve():
    server = grpc.server(futures.ThreadPoolExecutor(max_workers=10))

```

```

HospitalAppointment_pb2_grpc.add_AppointmentServiceServicer_to_server(Appoin
tmentServiceServicer(), server)
print("正在启动服务端")
server.add_insecure_port('[::]:50051')
server.start()
print("服务端已启动, 在端口 50051 监听")
server.wait_for_termination()

```

```
if __name__ == '__main__':  
    serve()
```

## 编写客户端代码

创建 Client.py。

外层一个 `while True`，然后给用户五个选择，除了实验内容要求的四个功能外，添加了一个退出功能用于退出 `while` 循环。

```
from http.client import responses  
  
import grpc  
from pkg_resources import empty_provider  
  
import HospitalAppointment_pb2  
import HospitalAppointment_pb2_grpc  
  
def run():  
    # 连接到 gRPC 服务器  
    channel = grpc.insecure_channel('localhost:50051')  
    stub = HospitalAppointment_pb2_grpc.AppointmentServiceStub(channel)  
  
    Appointment = getattr(HospitalAppointment_pb2, 'Appointment')  
    AppointmentId = getattr(HospitalAppointment_pb2, 'AppointmentId')  
    PatientName = getattr(HospitalAppointment_pb2, 'PatientName')  
    Empty = getattr(HospitalAppointment_pb2, 'Empty')  
  
    while True:  
        print("请选择功能（输入对应的数字）：")  
        print("1.预约挂号")  
        print("2.通过ID查询预约信息")  
        print("3.通过病人查询预约信息")  
        print("4.取消预约")  
        print("5.退出")  
        opt = int(input())  
        # 预约挂号  
        if opt == 1:  
            empty = Empty()  
            size = stub.getStorageSize(empty)  
            appointmentid = size.size + 1  
            patientname = input("请输入病人姓名：")  
            doctorname = input("请输入医生姓名：")  
            department = input("请输入科室：")  
            date = input("请输入预约日期：")  
            timeSlot = input("请输入预约时间段：")  
            appointment = Appointment()
```

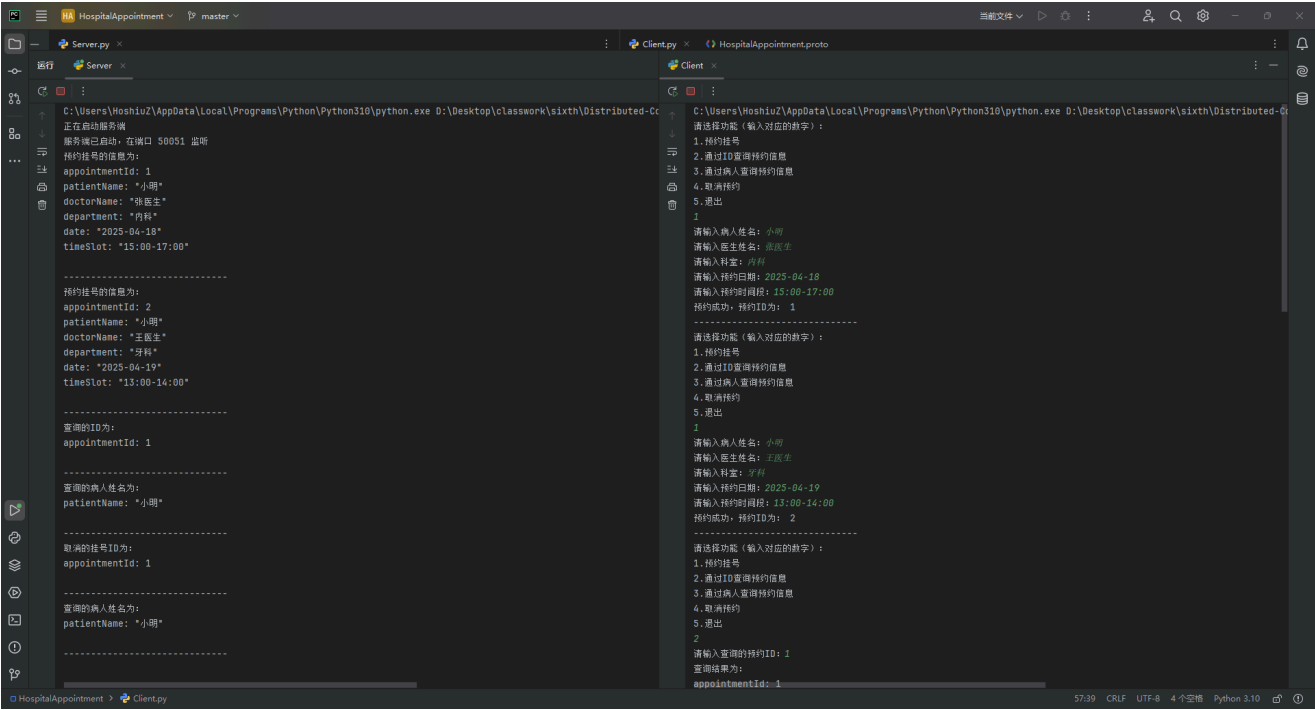
```

        appointmentId = appointmentid,
        patientName = patientname,
        doctorName=doctorname,
        department=department,
        date=date,
        timeSlot=timeSlot
    )
    response = stub.bookAppointment(appointment)
    if response.res:
        print(f"预约成功, 预约ID为: {appointmentid}" )
    else:
        print("预约失败")
    print("-----")
# 通过ID查询
elif opt == 2:
    appointmentid = int(input("请输入查询的预约ID: "))
    query_by_id = AppointmentId(appointmentId=appointmentid)
    response = stub.queryById(query_by_id)
    if not response.ListFields():
        print("没有查询到该ID对应的信息")
    else:
        print("查询结果为: ")
        print(response)
    print("-----")
# 通过病人姓名查询
elif opt == 3:
    patientname = input("请输入要查询的病人姓名: ")
    query_by_name = PatientName(patientName=patientname)
    response = stub.queryByPatient(query_by_name)
    if len(response.appointments) == 0:
        print("没有该病人的挂号记录")
    else:
        print("查询结果为: ")
        print(response)
    print("-----")
elif opt == 4:
    appointmentid = int(input("请输入需要取消预约的ID: "))
    cancel_appointment = AppointmentId(appointmentId=appointmentid)
    response = stub.cancelAppointment(cancel_appointment)
    if response.res:
        print("取消成功")
    else:
        print("取消失败")
else:
    break

if __name__ == '__main__':
    run()

```

# 运行示例



```
Server.py
C:\Users\Hoshiz\AppData\Local\Programs\Python\Python310\python.exe D:\Desktop\classwork\sixth\Distributed-C...
正在启动服务端
服务端已启动，在端口 50051 监听
预约挂号的信息为：
appointmentId: 1
patientName: "小明"
doctorName: "张医生"
department: "内科"
date: "2025-04-18"
timeSlot: "15:00-17:00"
-----
预约挂号的信息为：
appointmentId: 2
patientName: "小明"
doctorName: "王医生"
department: "牙科"
date: "2025-04-19"
timeSlot: "13:00-14:00"
-----
查询的ID为：
appointmentId: 1
-----
查询的病人姓名为：
patientName: "小明"
-----
取消的挂号ID为：
appointmentId: 1
-----
查询的病人姓名为：
patientName: "小明"
-----

Client.py
C:\Users\Hoshiz\AppData\Local\Programs\Python\Python310\python.exe D:\Desktop\classwork\sixth\Distributed-C...
请选择功能（输入对应的数字）：
1. 预约挂号
2. 通过ID查询预约信息
3. 通过病人查询预约信息
4. 取消预约
5. 退出
1
请输入病人姓名：小明
请输入医生姓名：张医生
请输入科室：内科
请输入预约日期：2025-04-18
请输入预约时段：15:00-17:00
预约成功，预约ID为： 1
-----
请选择功能（输入对应的数字）：
1. 预约挂号
2. 通过ID查询预约信息
3. 通过病人查询预约信息
4. 取消预约
5. 退出
1
请输入病人姓名：小明
请输入医生姓名：王医生
请输入科室：牙科
请输入预约日期：2025-04-19
请输入预约时段：13:00-14:00
预约成功，预约ID为： 2
-----
请选择功能（输入对应的数字）：
1. 预约挂号
2. 通过ID查询预约信息
3. 通过病人查询预约信息
4. 取消预约
5. 退出
2
请输入查询的预约ID： 1
查询结果为：
appointmentId: 1
```