National College of Ireland

Automated Scheduling applications

Jiayu Meng

Student ID: 22101837

Distributed Systems
March 2021

Student information

Contents

1 Introduction

This proposal presents an automated scheduling app that assists individuals in scheduling tasks, appointments, and meetings. The application is composed of three main services, each with its unique and advanced functionalities and substantial contributions to the overall application.

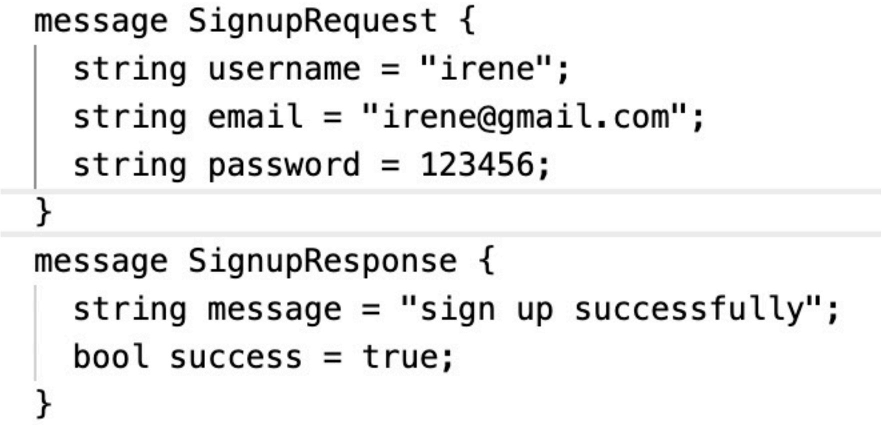
Service 1: User Management

This service deals with user authentication and authorization, including signup and login functionalities.

1.1 Methods

1.1.1 signup RPC

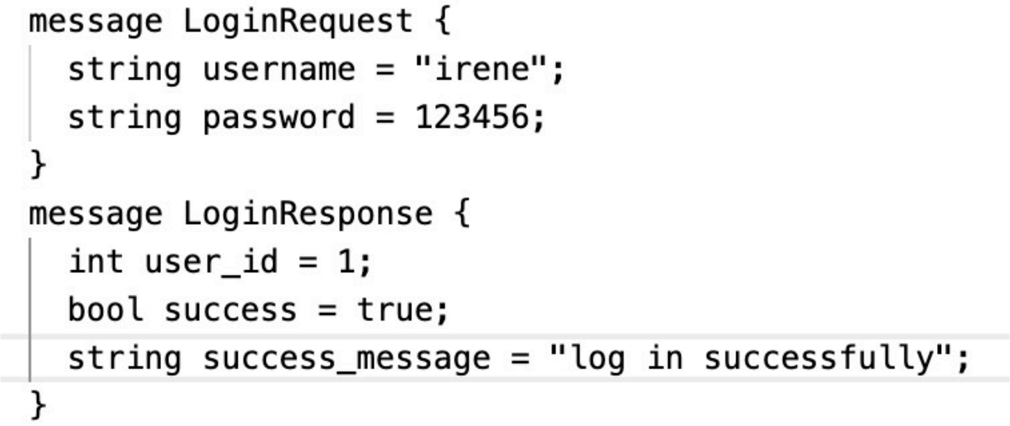
The signup RPC is a Unary RPC that registers a new user with a username and password, saves their info in the database, and delivers a success/error message to the client.



```
message SignupRequest {
  string username = "irene";
  string email = "irene@gmail.com";
  string password = 123456;
}
message SignupResponse {
  string message = "sign up successfully";
  bool success = true;
}
```

1.1.2 login RPC

The login method can be implemented using server streaming RPC, where the client sends the user's login information to the server and receives a stream of progress messages in response.



```
message LoginRequest {
  string username = "irene";
  string password = 123456;
}
message LoginResponse {
  int user_id = 1;
  bool success = true;
  string success_message = "log in successfully";
}
```

.

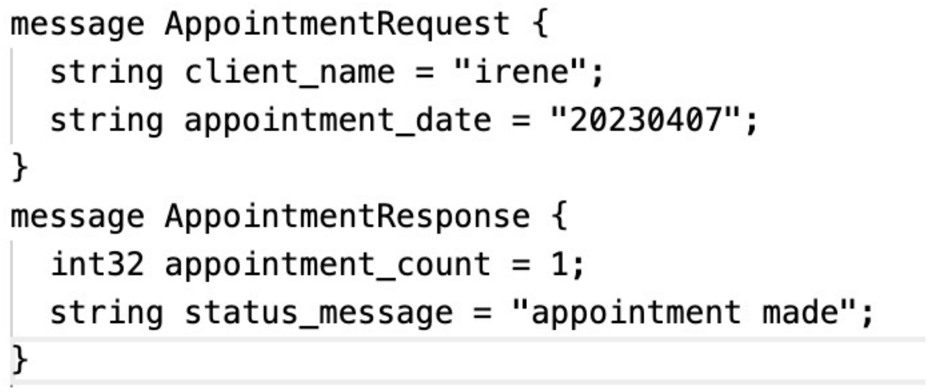
Service 2: Appointment Management

The service allows users to schedule appointments by setting a specific date and time and adding participant information. Users can modify or reschedule appointments and cancel them with the "Cancel Appointment" feature.

2.1 Methods

2.1.1 RPC Method 1 Schedule Appointment

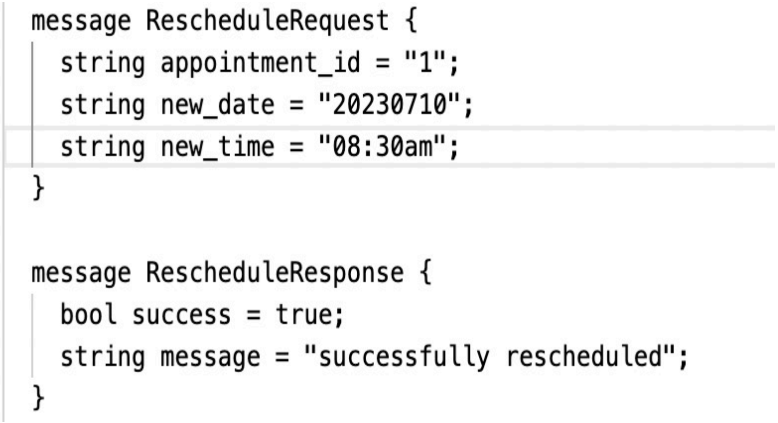
Client streaming RPC is used to schedule appointments by sending a stream of requests to the server, which response with a status message after processing them all.



```
message AppointmentRequest {
  string client_name = "irene";
  string appointment_date = "20230407";
}
message AppointmentResponse {
  int32 appointment_count = 1;
  string status_message = "appointment made";
}
```

2.1.2 RPC Method 2 Reschedule Appointment

Rescheduling appointments can be done using client streaming RPC, where a stream of rescheduling requests is sent to the server and a single status message is returned.

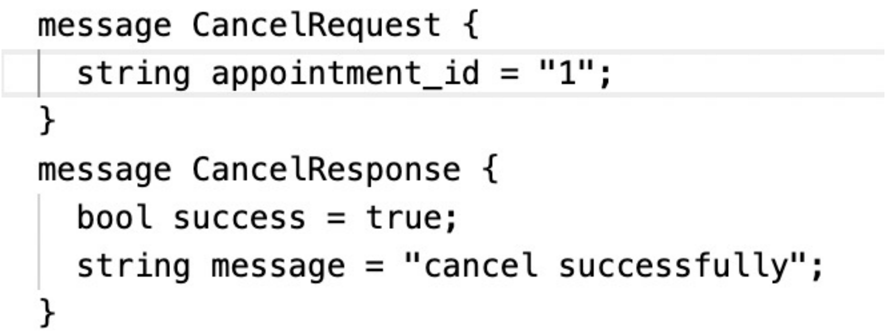


```
message RescheduleRequest {
  string appointment_id = "1";
  string new_date = "20230710";
  string new_time = "08:30am";
}

message RescheduleResponse {
  bool success = true;
  string message = "successfully rescheduled";
}
```

2.1.3 RPC Method 3 Cancel Appointment

The cancel appointment function using client streaming RPC sends a stream of appointment cancellation requests to the server, receiving a single message indicating the status of the appointments.



```
message CancelRequest {
  string appointment_id = "1";
}
message CancelResponse {
  bool success = true;
  string message = "cancel successfully";
}
```

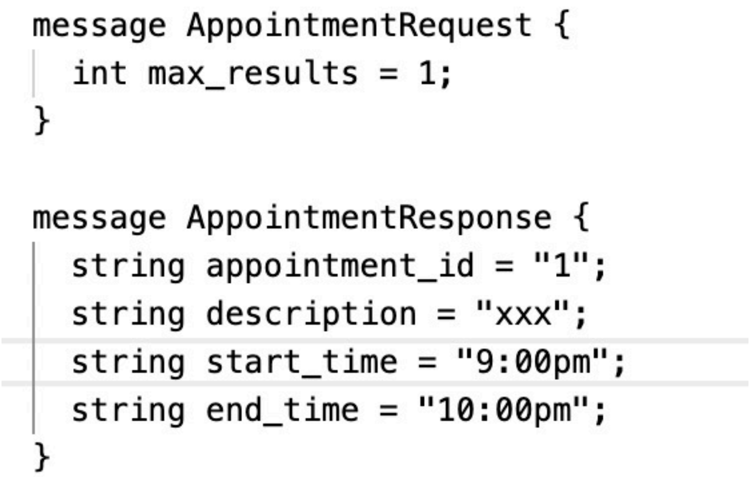
Service 3: Calendar service

The Calendar service manages and synchronizes user calendars across multiple platforms, allowing users to access, edit, and create events while setting reminders. The service also sends real-time notifications about upcoming appointments and meetings through various channels, such as email and SMS, based on user preferences and settings.

3.1 Methods

3.1.1 RPC Method 1 Get Upcoming Appointments

Bidirectional streaming RPC enables real-time communication between client and server, making it ideal for an automated scheduling app's "Get Upcoming Appointments" method.

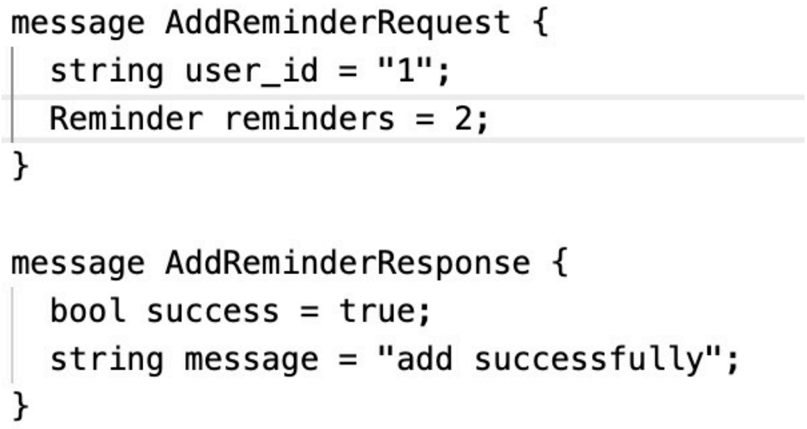


```
message AppointmentRequest {
  int max_results = 1;
}

message AppointmentResponse {
  string appointment_id = "1";
  string description = "xxx";
  string start_time = "9:00pm";
  string end_time = "10:00pm";
}
```

3.1.2 RPC Method2 Add Reminder

Client streaming RPC allows adding a reminder by sending a stream of requests to the server and receiving a single response message indicating the status of the request.



```
message AddReminderRequest {
  string user_id = "1";
  Reminder reminders = 2;
}

message AddReminderResponse {
  bool success = true;
  string message = "add successfully";
}
```

In conclusion, I want to emphasize that this proposal is only a preliminary idea and is subject to change as I proceed with the development of the distributed system.