Week 3

gRPC & Protobuf

Trainers

Ta Trung Cang

Full-stack Engineer

Segmentation Platform

Hieu Vo

Engineering Manager

Transport

Group chat

https://m.me/join/Abb2RsdhvqF82R4j

Agenda

- What is RPC?
- What is gRPC?
- What is Protocol buffer (protobuf)?
- gPRC and JSON-based RESTful Comparison?
- Live code demo
- Assignment

What is RPC?

- A Remote Procedure Call (RPC) is a function call with execution details that are transparent to the callers
- The actual code is being executed on remote machines via TCP/UDP connections

RPC Alternatives

Protocol Type/Format	**Syntax**	**Advantages**	**Disadvantages**
Raw TCP	Varies depending on the defined protocol	Allows you to define your own protocol to be business optimized	Much more effort
HTTP/HTML Forms	a=b&c=d	Browser support	Lack of rich structure and type
HTTP/XML	<Root><Item><Key>a</Key><Value>b</Value></Item></Root	Well-structured and contains types	Verbose
HTTP/JSON	{“a”: “b”, “c”: “d”}	Simpler than XML, easier to read	N/A

What is gRPC?

- An implementation of RPC initially developed by Google.
- Uses Protocol Buffers to define the message structure.
- Uses HTTP/2.
- Contains best practices features like timeout, authentication, etc.

What is Protocol Buffers (protobuf)?

- Protocol buffers are Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data.
- Define data structure once, generate source code to easily write and read.
- Common programming languages support: Go, Java, Python, Ruby, etc.

What is Protocol Buffers (protobuf)?

Example:

```
service UserService {
 rpc GetUserList(GetUserListRequest) returns (GetUserListResponse) {
   option (google.api.http) = {
     get: "/users"
   };
 }
}

message GetUserListRequest {
 int32 limit = 1;
}

message GetUserListResponse {
 repeated User users = 1;
}

message User {
 int64 ID = 1;
 string name = 2;
}
```

gRPC and JSON-based RESTful Comparison

Advantages:

Developer Friendly

- API description using Protocol Buffers
- Code generation

Performance

- Leverages HTTP/2
- Efficiency in data transfer

gPRC and JSON-based RESTful Comparison

Disadvantages:

- Debugging is not as straightforward because it’s not in plaintext.
- Cannot be called directly from browsers, primarily for internal services
- Cannot change/modify structure of data on the fly

Live demo code

Assignment

Implement a simple passenger feedback service, with basic functions:

- Add passenger feedback
- Get by passenger id
- Get by booking code
- Delete by passenger id

Requirements:

- Implement gRPC server/client
- Simply use local variable as storage
- 1 booking has only 1 feedback
- 1 passenger can add multiple feedbacks

```
message PassengerFeedback {
 string bookingCode = 1;
 int32  passengerID = 2;
 string feedback    = 3;
}
```

References

https://grpc.io/docs/quickstart/go/

https://developers.google.com/protocol-buffers/

https://code.tutsplus.com/tutorials/rest-vs-grpc-battle-of-the-apis--cms-30711

https://viblo.asia/p/protocol-buffers-la-gi-va-nhung-dieu-can-ban-can-biet-ve-no-maGK7D99Zj2

Thanks