

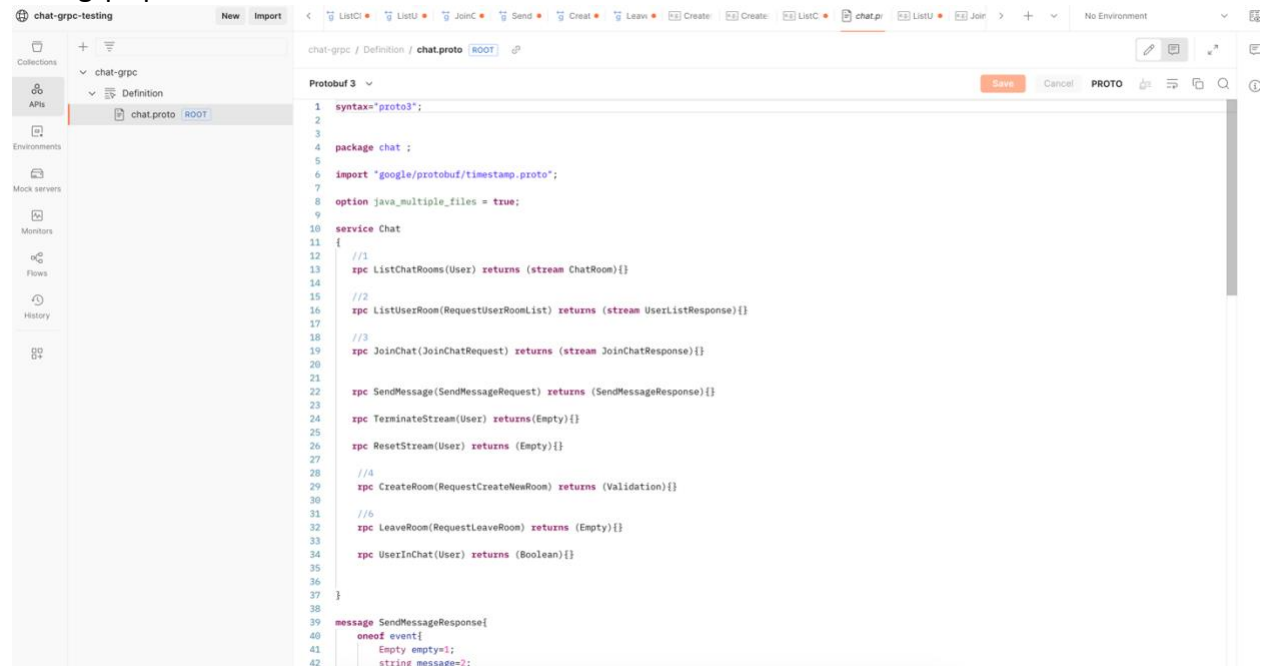
Public tests: <https://www.postman.com/orange-zodiac-60388/workspace/chat-grpc-testing/collection/65d130d7af0c01151a5cc1a8?action=share&creator=30379959>

The testing was made with Postman to ensure all requests and responses are consistently delivered, well function and reliable.

Steps to create tests: (those are already completed).

1. Create API

chat-grpc.proto



2. Add new gRPC request

3. Enter url: select the protobuf API created above

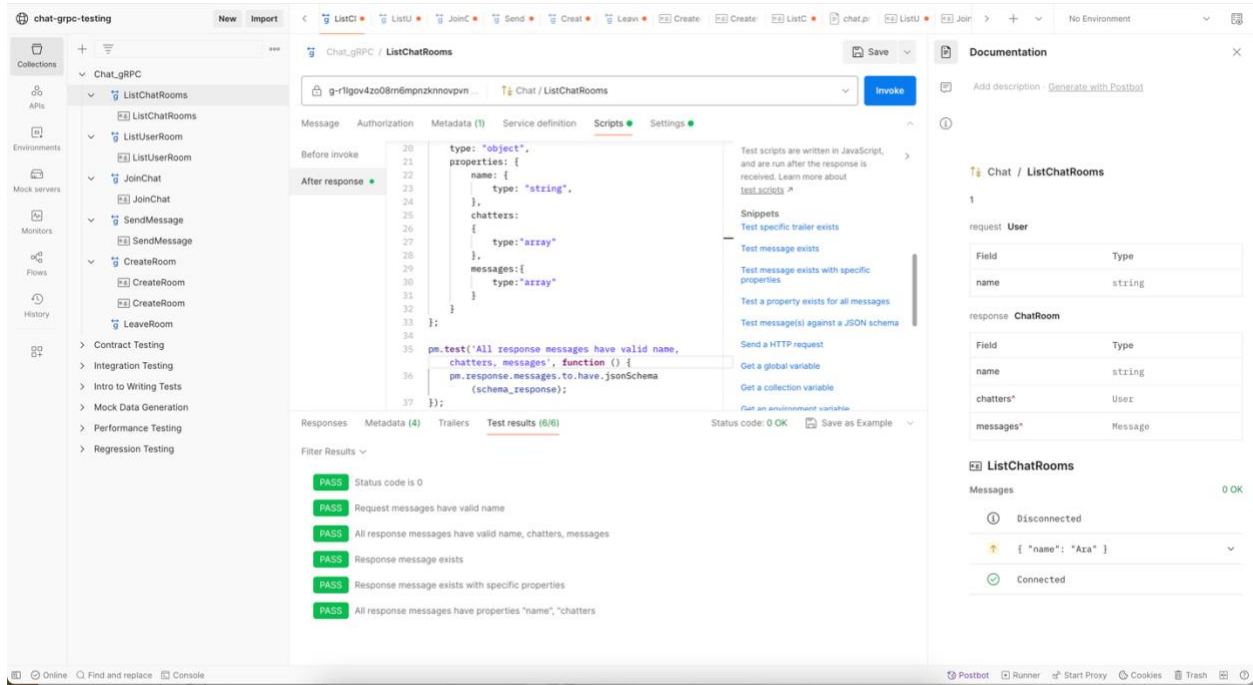
3. Select the method to test

4. Create request, update metadata and write tests in script (javascript)

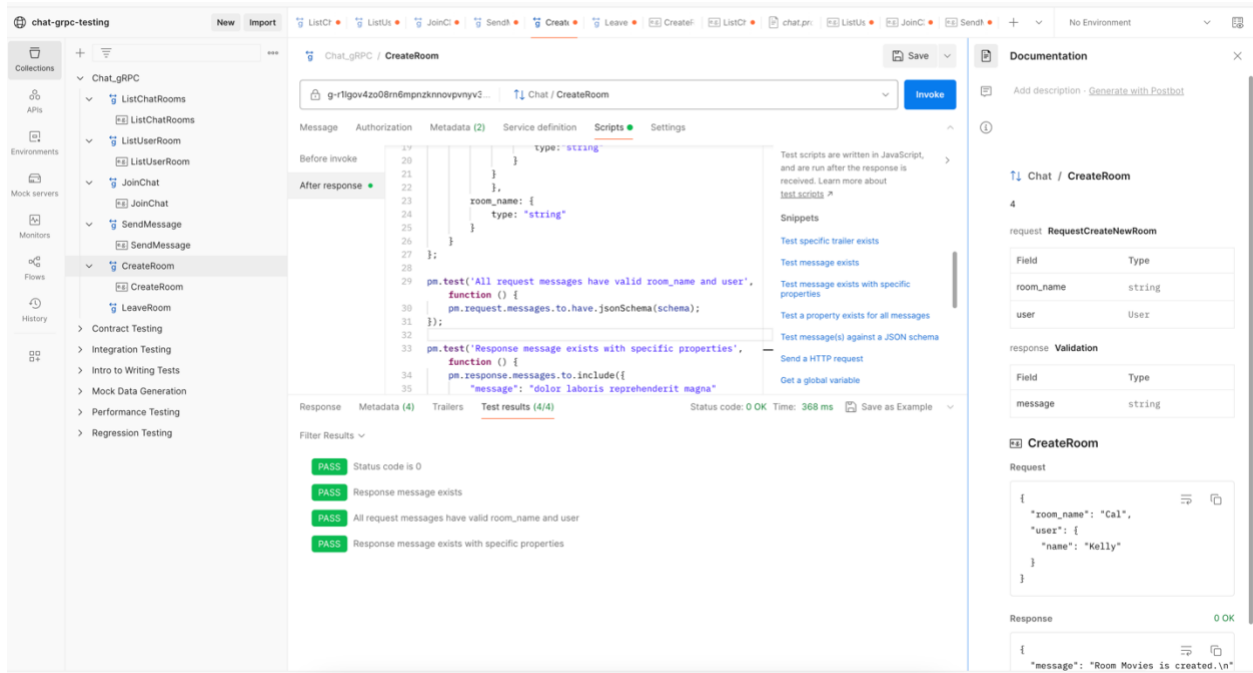
List of all tests for each method:

- To replicate the tests, choose the method you want to run the test and hit the button 'Invoke' (the blue button on the top right corner)
- Examples for the request and response are also included in the code, documentation for those were auto generated by Postman.

1. Chat list tests:
All 7 tests passed.



2. Create room tests:
All 4 tests passed.



5. Send message tests:

All 4 tests passed.

The screenshot shows the 'chat-grpc-testing' interface with the 'SendMessage' test selected. The test results show four passed tests:

- PASS: Status code is 0
- PASS: Response message exists with specific properties
- PASS: request messages have valid structure
- PASS: Response message exists

The test script is as follows:

```
27   room_name: {
28     type: "string"
29   };
30 };
31 };
32
33 pm.test("Request messages have valid structure", function () {
34   pm.request.messages.to.have.jsonSchema(schema);
35 });
36
37 pm.test("Response message exists", function () {
38   pm.response.to.have.message({
39     empty: {},
40     message: "magna consectetur"
41   });
42 });
43
44
```

The response is:

```
{
  "room_name": "magna consectetur",
  "event": "OneOf"
}
```

The status code is 0 OK, and the time is 395 ms.

6. Leave chat tests:

All 4 tests passed.

The screenshot shows the 'chat-grpc-testing' interface with the 'LeaveRoom' test selected. The test results show four passed tests:

- PASS: Status code is 0
- PASS: Response message exists
- PASS: All request messages have valid room_name and user_name
- PASS: Response message exists with specific properties

The test script is as follows:

```
12   room_name: {
13     type: "string",
14   },
15   user_name: {
16     type: "string",
17   }
18 };
19 };
20
21 pm.test("All request messages have valid room_name and
22 user_name", function () {
23   pm.request.messages.to.have.jsonSchema(schema);
24 });
25
26 pm.test("Response message exists with specific properties",
27 function () {
28   pm.response.messages.to.include({
29     room_name: "magna consectetur",
30     user_name: "magna consectetur"
31   });
32 });
33
```

The response is:

```
{
  "room_name": "magna consectetur",
  "user_name": "magna consectetur",
  "event": "OneOf"
}
```

The status code is 0 OK, and the time is 365 ms.