CLOUDBEE ASSIGNMENT

GIVEN QUESTION:

All API referenced are gRPC APIs, not REST ones.

I want to board a train from London to France. The train ticket will cost $20.

Create API where you can submit a purchase for a ticket. Details included in the receipt are: From, To, User , price paid.

User should include first and last name, email address

The user is allocated a seat in the train. Assume the train has only 2 sections, section A and section B.

An API that shows the details of the receipt for the user

An API that lets you view the users and seat they are allocated by the requested section An API to remove a user from the train

An API to modify a user's seat , how to proceed this?

SOLUTION:

1) **PROTOBUF:**

```
syntax = "proto3";

message User {
  string first_name = 1;
  string last_name = 2;
  string email = 3;
}

message Ticket {
  string from = 1;
  string to = 2;
  User user = 3;
  float price = 4;
  string seat_section = 5;
}

service TrainService {
  rpc PurchaseTicket(Ticket) returns (Ticket);
```

```
  rpc GetReceipt(string) returns (Ticket);
  rpc GetUserSeats(string) returns (map<string, string>);
  rpc RemoveUser(string) returns (bool);
  rpc ModifyUserSeat(User) returns (bool);
}
```

2) IMPLEMENT GRPC (PYTHON):

3) IMPLEMENT SERVER-SIDE API (FLASK + PYTHON):

```
from flask import Flask, request, jsonify
from grpc import insecure_channel, StatusCode, RpcError
from your_grpc_module import train_pb2, train_pb2_grpc

app = Flask(__name__)

# gRPC connection
channel = insecure_channel('localhost:50051')  # Update with your gRPC server details
stub = train_pb2_grpc.TrainServiceStub(channel)


@app.route('/purchase_ticket', methods=['POST'])
def purchase_ticket():
    data = request.get_json()
    ticket = train_pb2.Ticket(
        from=data['from'],
        to=data['to'],
        user=train_pb2.User(
            first_name=data['user']['first_name'],
            last_name=data['user']['last_name'],
            email=data['user']['email']
        ),
        price=data['price']
    )

    try:
        response = stub.PurchaseTicket(ticket)
        return jsonify(response), 201
    except RpcError as e:
        if e.code() == StatusCode.ALREADY_EXISTS:
            return jsonify({'error': 'Ticket already purchased'}), 400
        else:
            return jsonify({'error': 'Internal server error'}), 500


@app.route('/get_receipt/<user_email>', methods=['GET'])
def get_receipt(user_email):
    try:
        response = stub.GetReceipt(user_email)
        return jsonify(response)
    except RpcError as e:
        if e.code() == StatusCode.NOT_FOUND:
```

```
            return jsonify({'error': 'User not found'}), 404
        else:
            return jsonify({'error': 'Internal server error'}), 500


@app.route('/get_user_seats/<section>', methods=['GET'])
def get_user_seats(section):
    try:
        response = stub.GetUserSeats(section)
        return jsonify(response)
    except RpcError as e:
        return jsonify({'error': 'Internal server error'}), 500


@app.route('/remove_user/<user_email>', methods=['DELETE'])
def remove_user(user_email):
    try:
        response = stub.RemoveUser(user_email)
        return jsonify({'success': response})
    except RpcError as e:
        if e.code() == StatusCode.NOT_FOUND:
            return jsonify({'error': 'User not found'}), 404
        else:
            return jsonify({'error': 'Internal server error'}), 500


@app.route('/modify_user_seat', methods=['PUT'])
def modify_user_seat():
    data = request.get_json()
    user = train_pb2.User(
        first_name=data['first_name'],
        last_name=data['last_name'],
        email=data['email']
    )

    try:
        response = stub.ModifyUserSeat(user)
        return jsonify({'success': response})
    except RpcError as e:
        if e.code() == StatusCode.NOT_FOUND:
            return jsonify({'error': 'User not found'}), 404
        else:
            return jsonify({'error': 'Internal server error'}), 500


if __name__ == '__main__':
    app.run(debug=True)
```

Then Run the flask server.