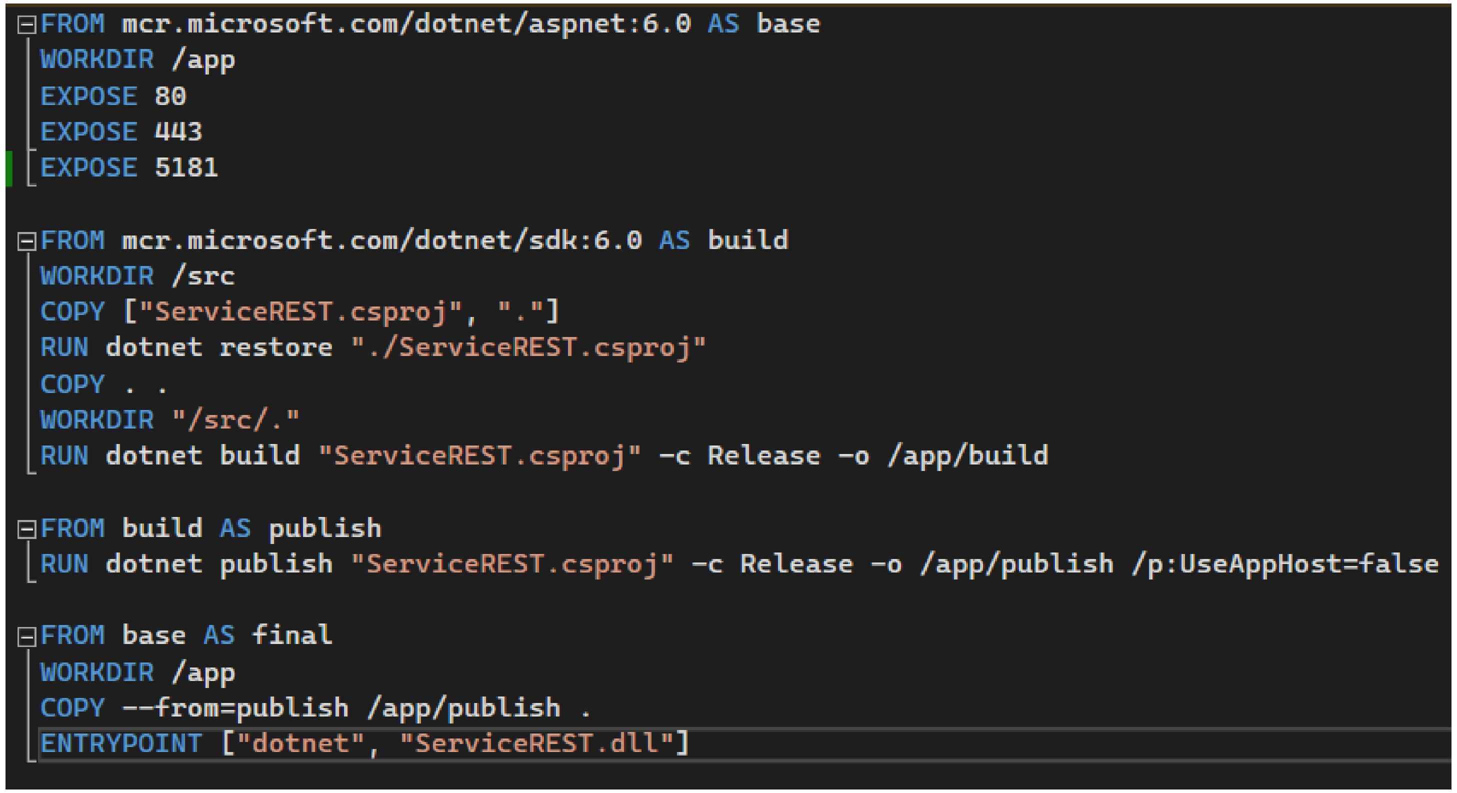
.NET MVC API + NestJS+MongoDB

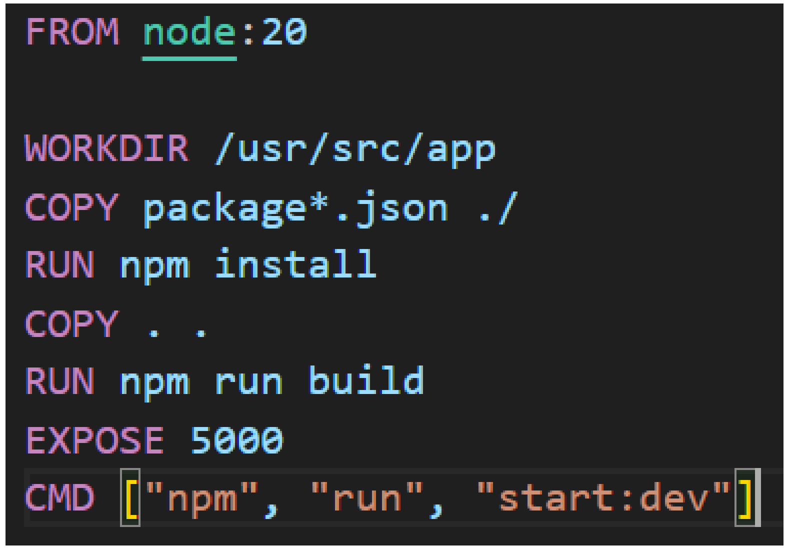
-Mina Petrović - 18332-

Docker

- Dockerfiles
- docker build –t imagename .



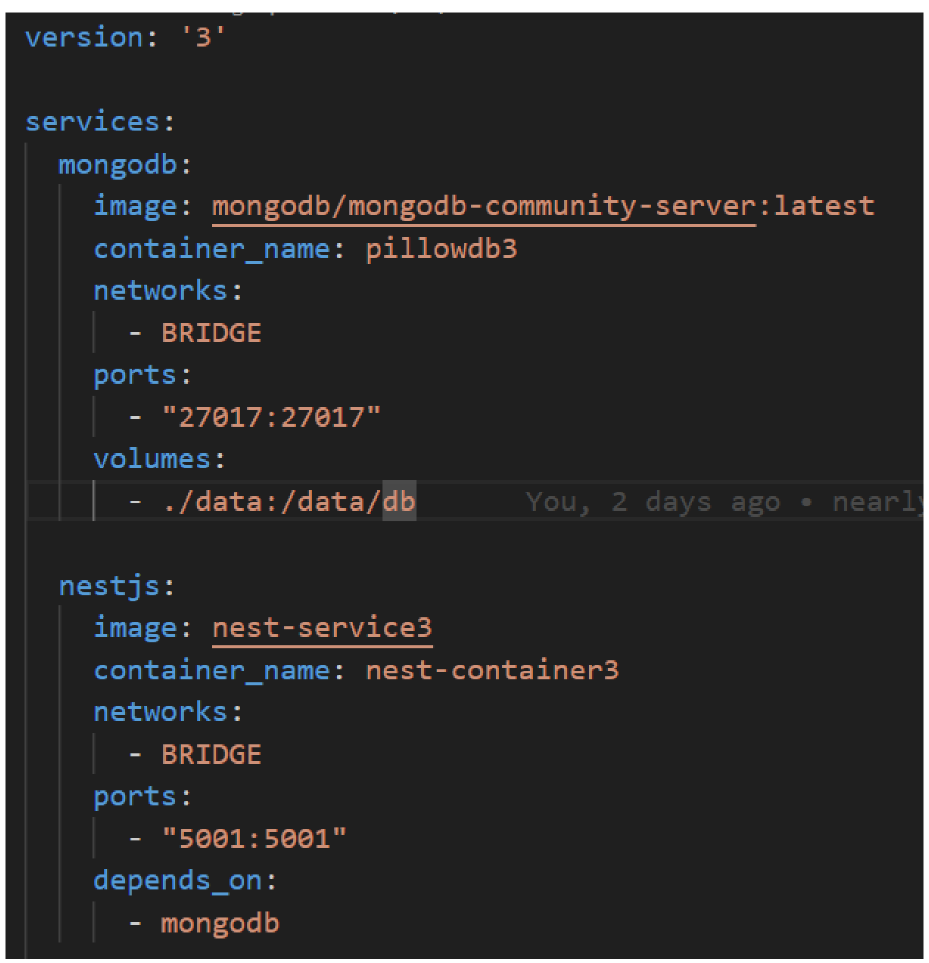
```
FROM mcr.microsoft.com/dotnet/aspnet:6.0 AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443
EXPOSE 5181

FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build
WORKDIR /src
COPY ["ServiceREST.csproj", "."]
RUN dotnet restore "./ServiceREST.csproj"
COPY . .
WORKDIR "/src/."
RUN dotnet build "ServiceREST.csproj" -c Release -o /app/build

FROM build AS publish
RUN dotnet publish "ServiceREST.csproj" -c Release -o /app/publish /p:UseAppHost=false

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "ServiceREST.dll"]
```



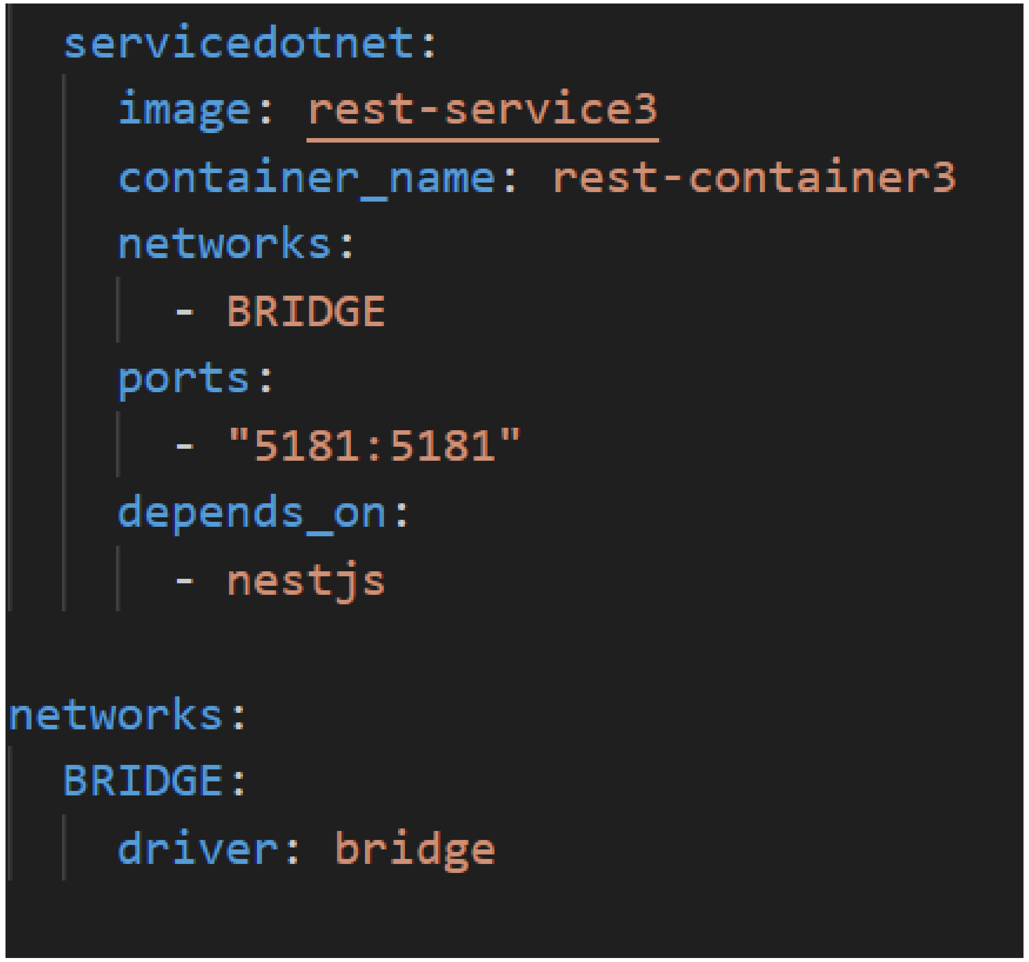
```
FROM node:20

WORKDIR /usr/src/app
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build
EXPOSE 5000
CMD ["npm", "run", "start:dev"]
```

Docker

docker-compose up/down



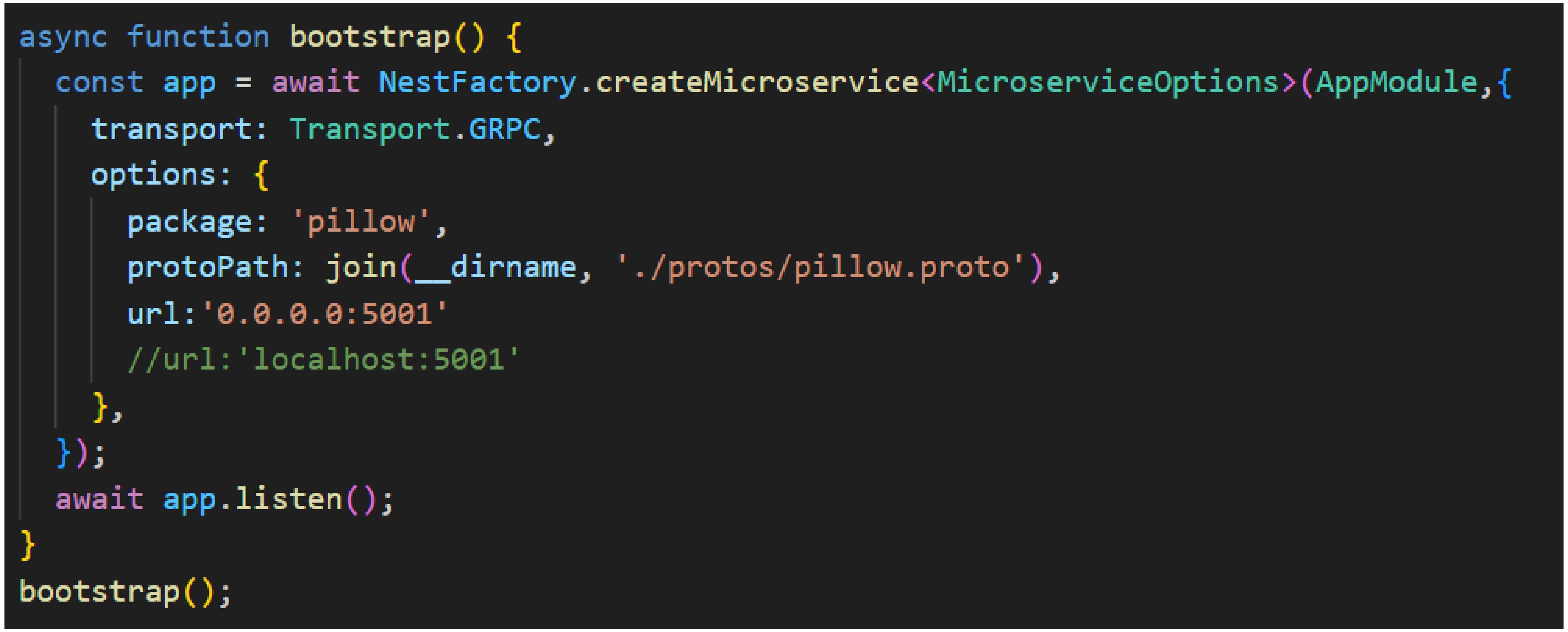
```
version: '3'

services:
  mongodb:
    image: mongodb/mongodb-community-server:latest
    container_name: pillowdb3
    networks:
      - BRIDGE
    ports:
      - "27017:27017"
    volumes:
      - ./data:/data/db        You, 2 days ago • nearly

  nestjs:
    image: nest-service3
    container_name: nest-container3
    networks:
      - BRIDGE
    ports:
      - "5001:5001"
    depends_on:
      - mongodb
```



```
  servicedotnet:
    image: rest-service3
    container_name: rest-container3
    networks:
      - BRIDGE
    ports:
      - "5181:5181"
    depends_on:
      - nestjs

networks:
  BRIDGE:
    driver: bridge
```

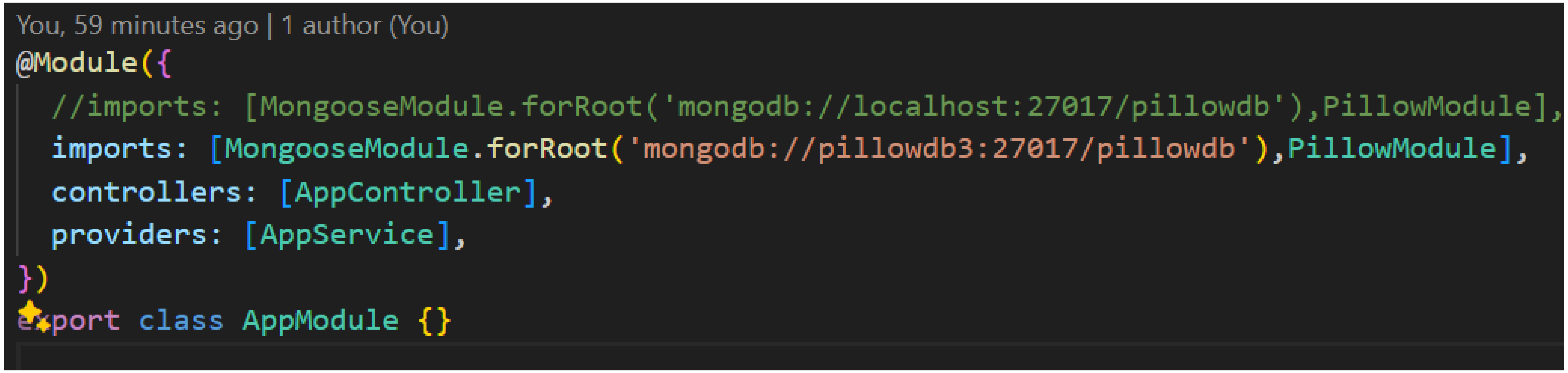
NestJS

- npm run start
- main.ts :

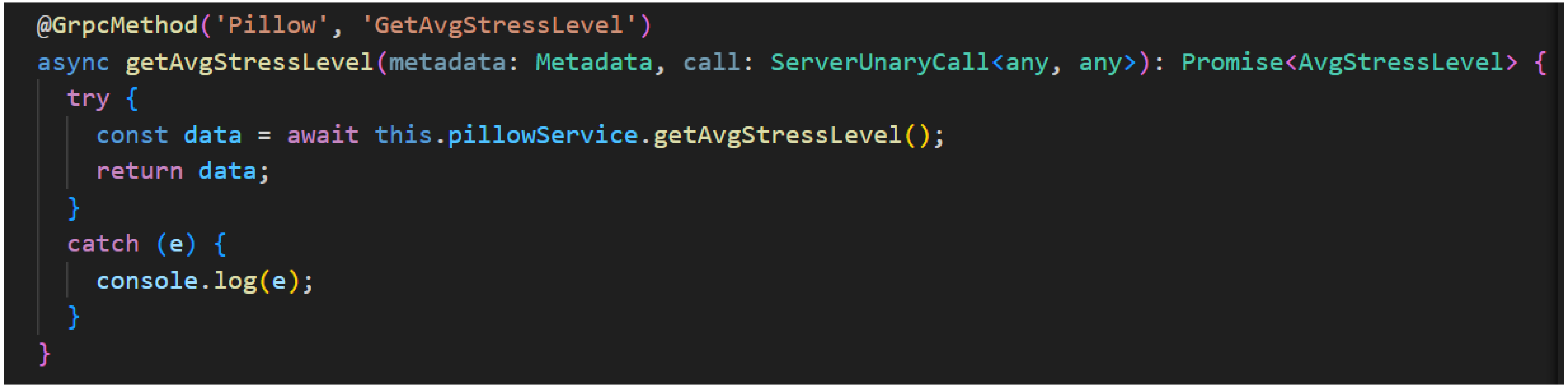


```
async function bootstrap() {
  const app = await NestFactory.createMicroservice<MicroserviceOptions>(AppModule,{
    transport: Transport.GRPC,
    options: {
      package: 'pillow',
      protoPath: join(__dirname, './protos/pillow.proto'),
      url:'0.0.0.0:5001'
      //url:'localhost:5001'
    },
  });
  await app.listen();
}
bootstrap();
```

NestJS

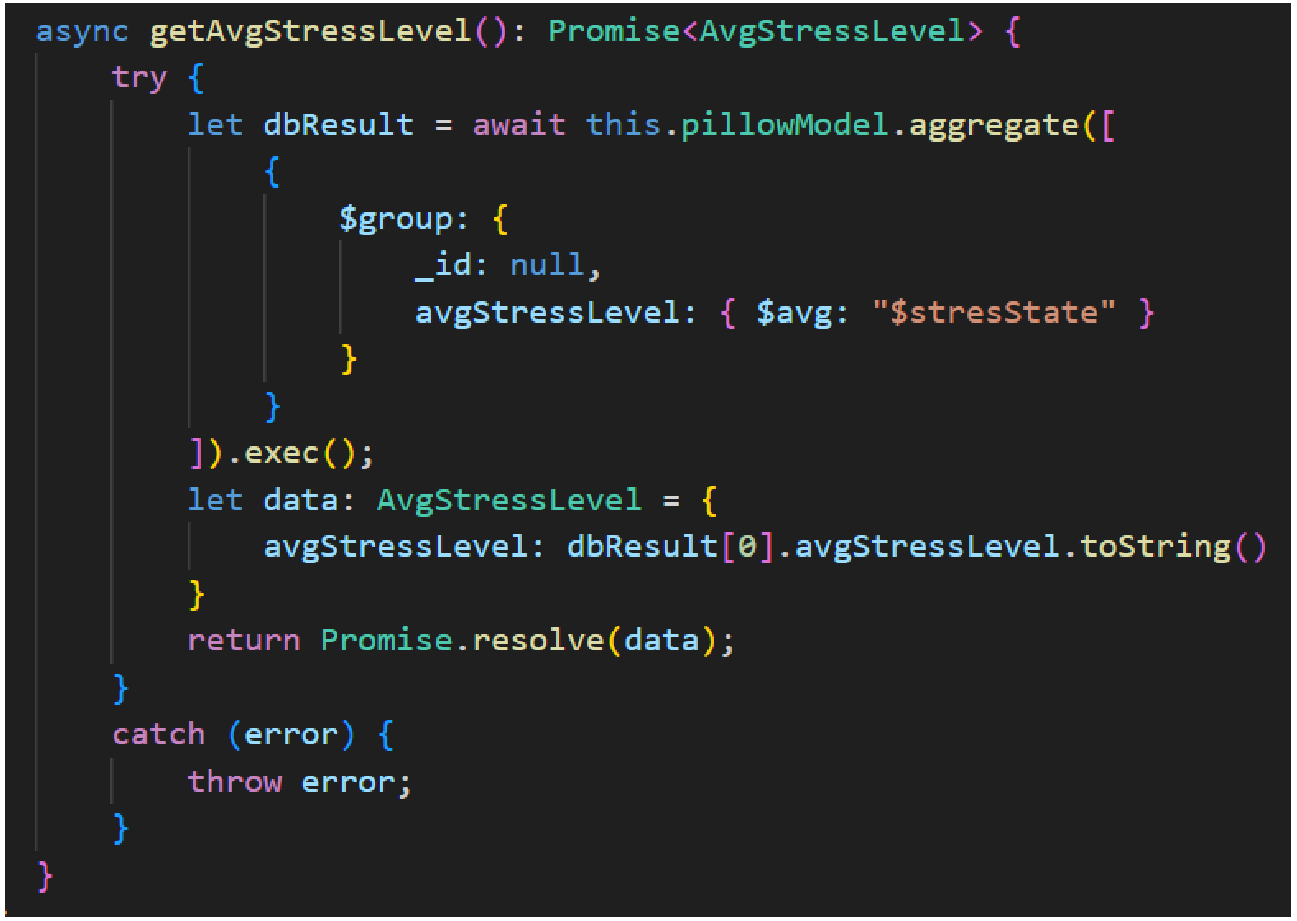


```
You, 59 minutes ago | 1 author (You)
@Module({
  //imports: [MongooseModule.forRoot('mongodb://localhost:27017/pillowdb'),PillowModule],
  imports: [MongooseModule.forRoot('mongodb://pillowdb3:27017/pillowdb'),PillowModule],
  controllers: [AppController],
  providers: [AppService],
})
export class AppModule {}
```



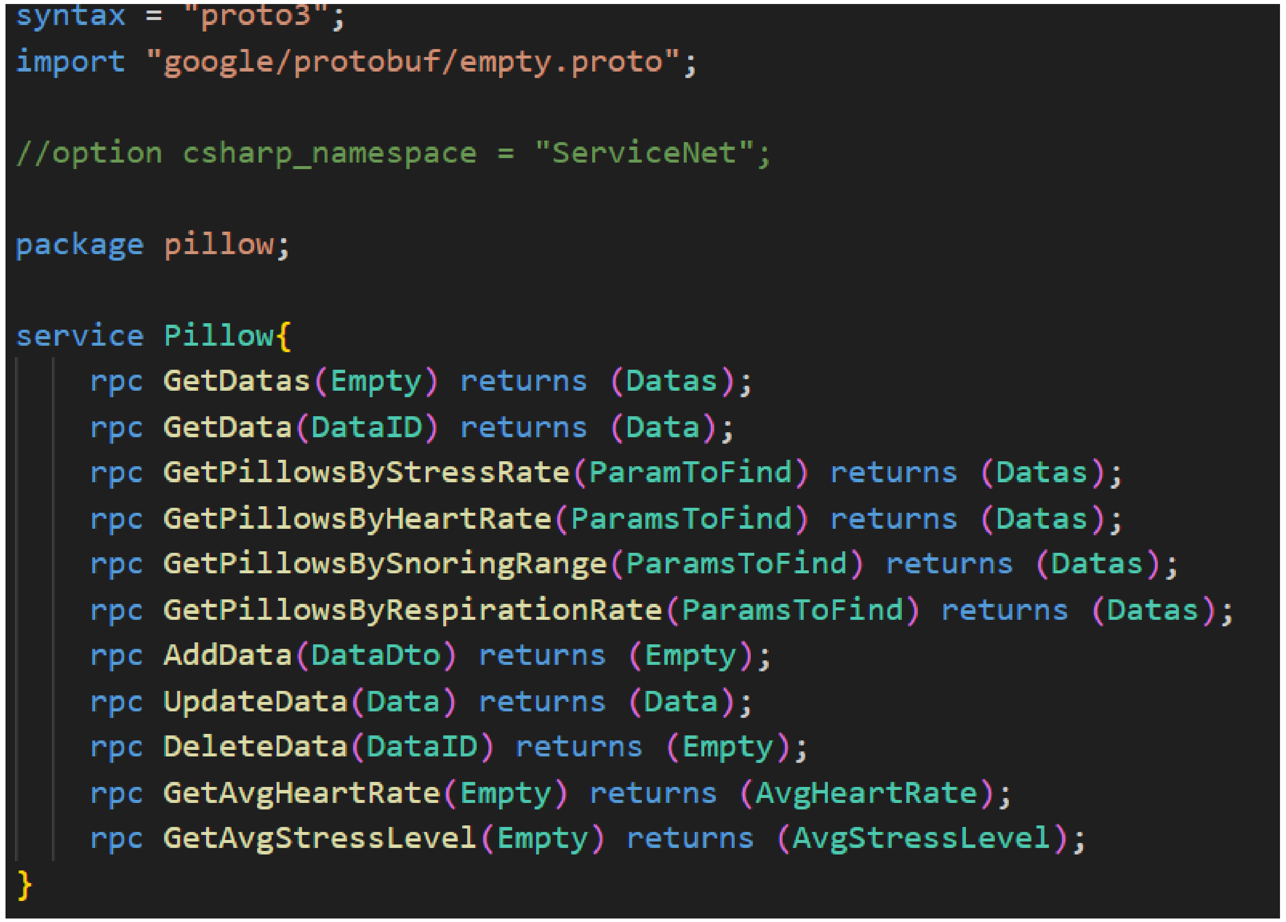
```
@GrpcMethod('Pillow', 'GetAvgStressLevel')
async getAvgStressLevel(metadata: Metadata, call: ServerUnaryCall<any, any>): Promise<AvgStressLevel> {
  try {
    const data = await this.pillowService.getAvgStressLevel();
    return data;
  }
  catch (e) {
    console.log(e);
  }
}
```

NestJS

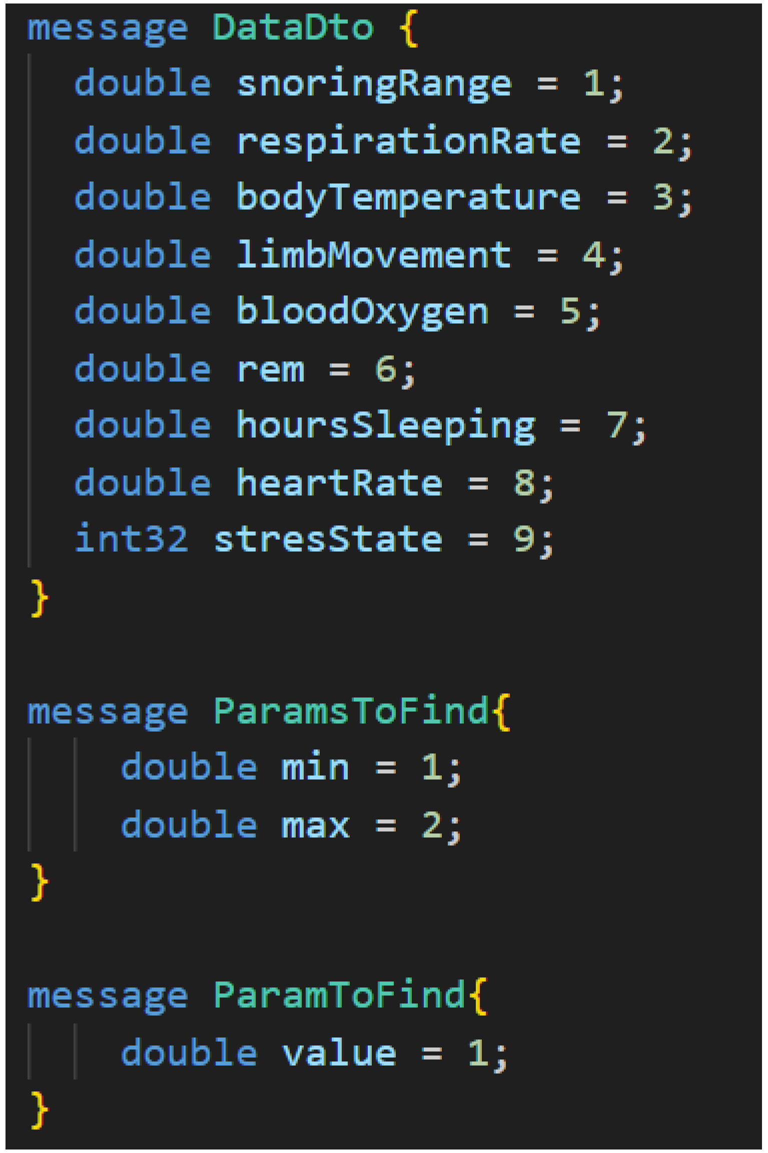


```
async getAvgStressLevel(): Promise<AvgStressLevel> {
    try {
        let dbResult = await this.pillowModel.aggregate([
            {
                $group: {
                    _id: null,
                    avgStressLevel: { $avg: "$stresState" }
                }
            }
        ]).exec();
        let data: AvgStressLevel = {
            avgStressLevel: dbResult[0].avgStressLevel.toString()
        }
        return Promise.resolve(data);
    }
    catch (error) {
        throw error;
    }
}
```

.proto file



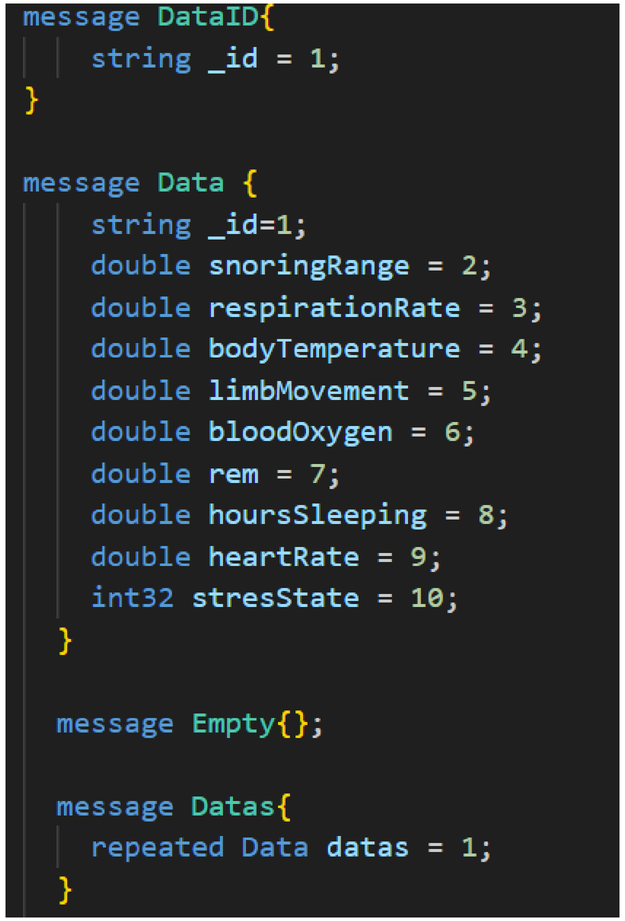
```
syntax = "proto3";
import "google/protobuf/empty.proto";

//option csharp_namespace = "ServiceNet";

package pillow;

service Pillow{
    rpc GetDatas(Empty) returns (Datas);
    rpc GetData(DataID) returns (Data);
    rpc GetPillowsByStressRate(ParamToFind) returns (Datas);
    rpc GetPillowsByHeartRate(ParamsToFind) returns (Datas);
    rpc GetPillowsBySnoringRange(ParamsToFind) returns (Datas);
    rpc GetPillowsByRespirationRate(ParamsToFind) returns (Datas);
    rpc AddData(DataDto) returns (Empty);
    rpc UpdateData(Data) returns (Data);
    rpc DeleteData(DataID) returns (Empty);
    rpc GetAvgHeartRate(Empty) returns (AvgHeartRate);
    rpc GetAvgStressLevel(Empty) returns (AvgStressLevel);
}
```



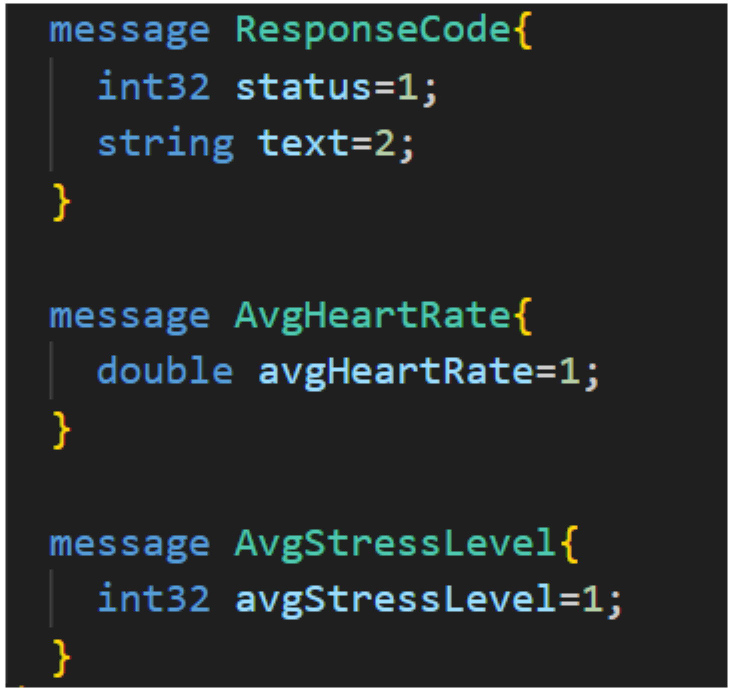
```
message DataDto {
  double snoringRange = 1;
  double respirationRate = 2;
  double bodyTemperature = 3;
  double limbMovement = 4;
  double bloodOxygen = 5;
  double rem = 6;
  double hoursSleeping = 7;
  double heartRate = 8;
  int32 stresState = 9;
}

message ParamsToFind{
    double min = 1;
    double max = 2;
}

message ParamToFind{
    double value = 1;
}
```

.proto file

Kompajliranje proto fajla u nest.js-u:
./node_modules/.bin/proto-loader-gen-types --longs=String --enums=String --defaults --oneofs --grpcLib=@grpc/grpc-js --outDir=src/protos/ src/protos/pillow.proto



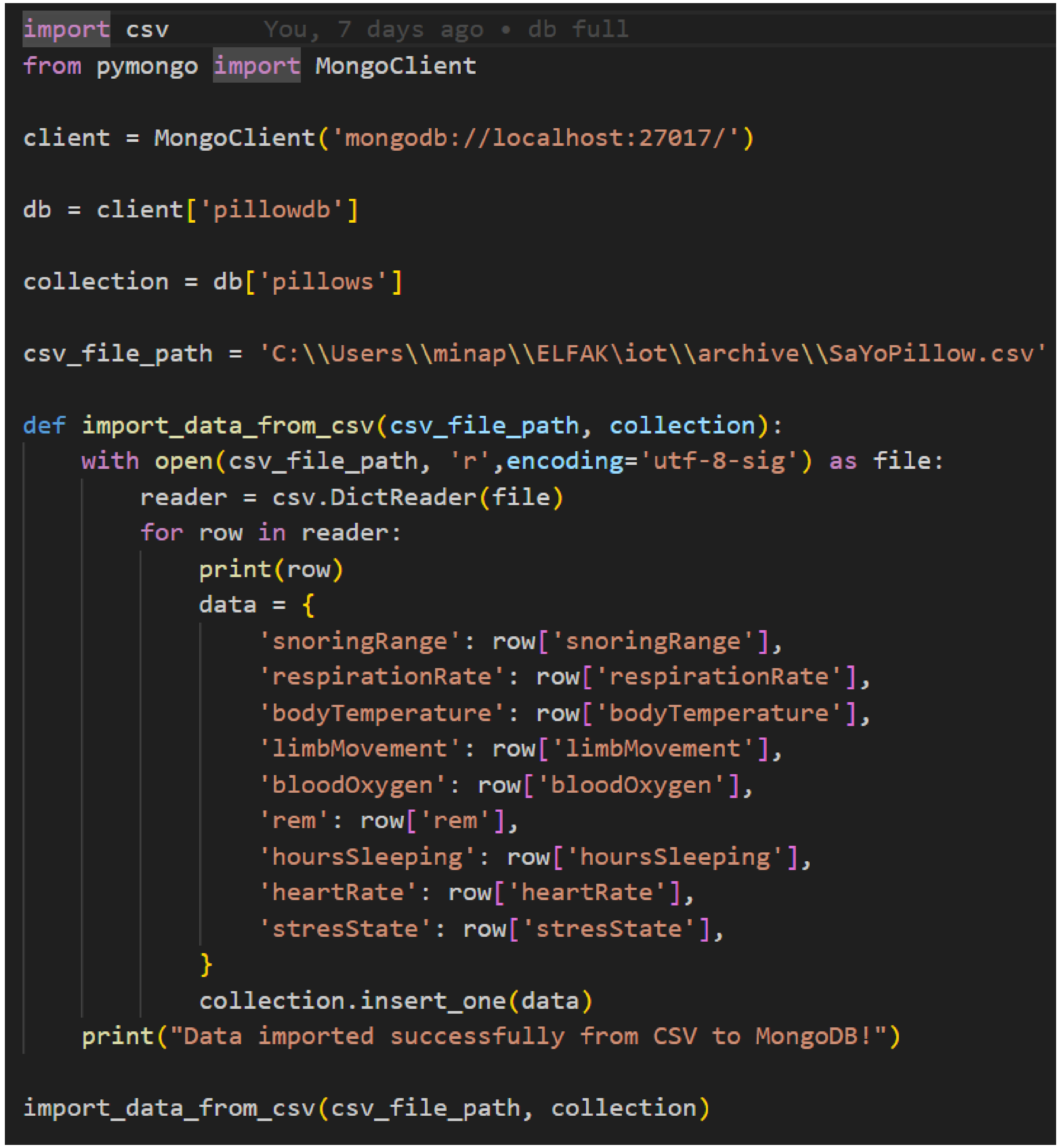
```
message DataID{
    string _id = 1;
}

message Data {
    string _id=1;
    double snoringRange = 2;
    double respirationRate = 3;
    double bodyTemperature = 4;
    double limbMovement = 5;
    double bloodOxygen = 6;
    double rem = 7;
    double hoursSleeping = 8;
    double heartRate = 9;
    int32 stresState = 10;
  }

  message Empty{};

  message Datas{
    repeated Data datas = 1;
  }
```



```
message ResponseCode{
  int32 status=1;
  string text=2;
}

message AvgHeartRate{
  double avgHeartRate=1;
}

message AvgStressLevel{
  int32 avgStressLevel=1;
}
```

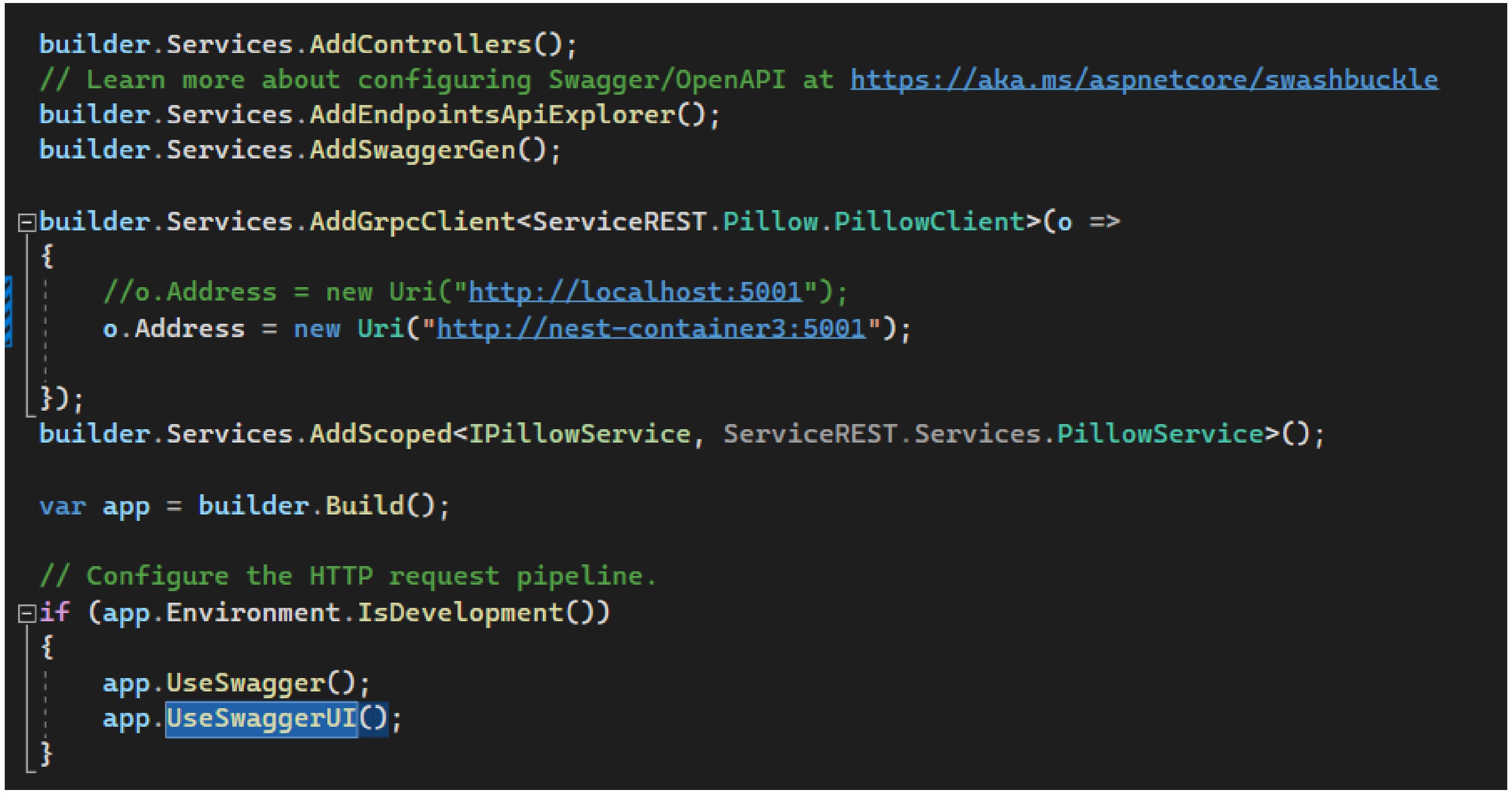
MongoDB

- Iz .csv u mongoDB korišćenjem pymongo biblioteke
- Kolekcija "pillows"
- MongoDB pokrenut na docker kontejneru



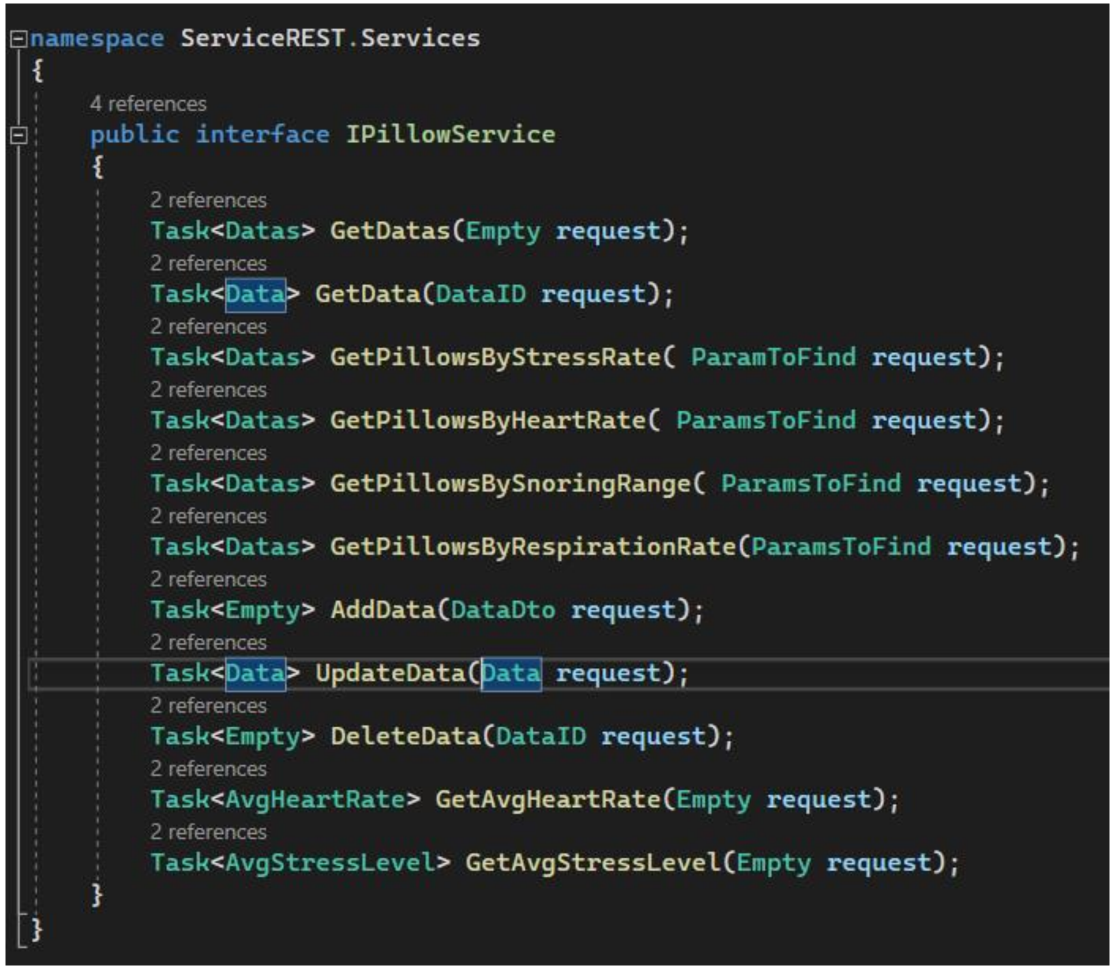
```
import csv        You, 7 days ago • db full
from pymongo import MongoClient

client = MongoClient('mongodb://localhost:27017/')

db = client['pillowdb']

collection = db['pillows']

csv_file_path = 'C:\\Users\\minap\\ELFAK\iot\\archive\\SaYoPillow.csv'

def import_data_from_csv(csv_file_path, collection):
    with open(csv_file_path, 'r',encoding='utf-8-sig') as file:
        reader = csv.DictReader(file)
        for row in reader:
            print(row)
            data = {
                'snoringRange': row['snoringRange'],
                'respirationRate': row['respirationRate'],
                'bodyTemperature': row['bodyTemperature'],
                'limbMovement': row['limbMovement'],
                'bloodOxygen': row['bloodOxygen'],
                'rem': row['rem'],
                'hoursSleeping': row['hoursSleeping'],
                'heartRate': row['heartRate'],
                'stresState': row['stresState'],
            }
            collection.insert_one(data)
    print("Data imported successfully from CSV to MongoDB!")

import_data_from_csv(csv_file_path, collection)
```

.NET MVC API

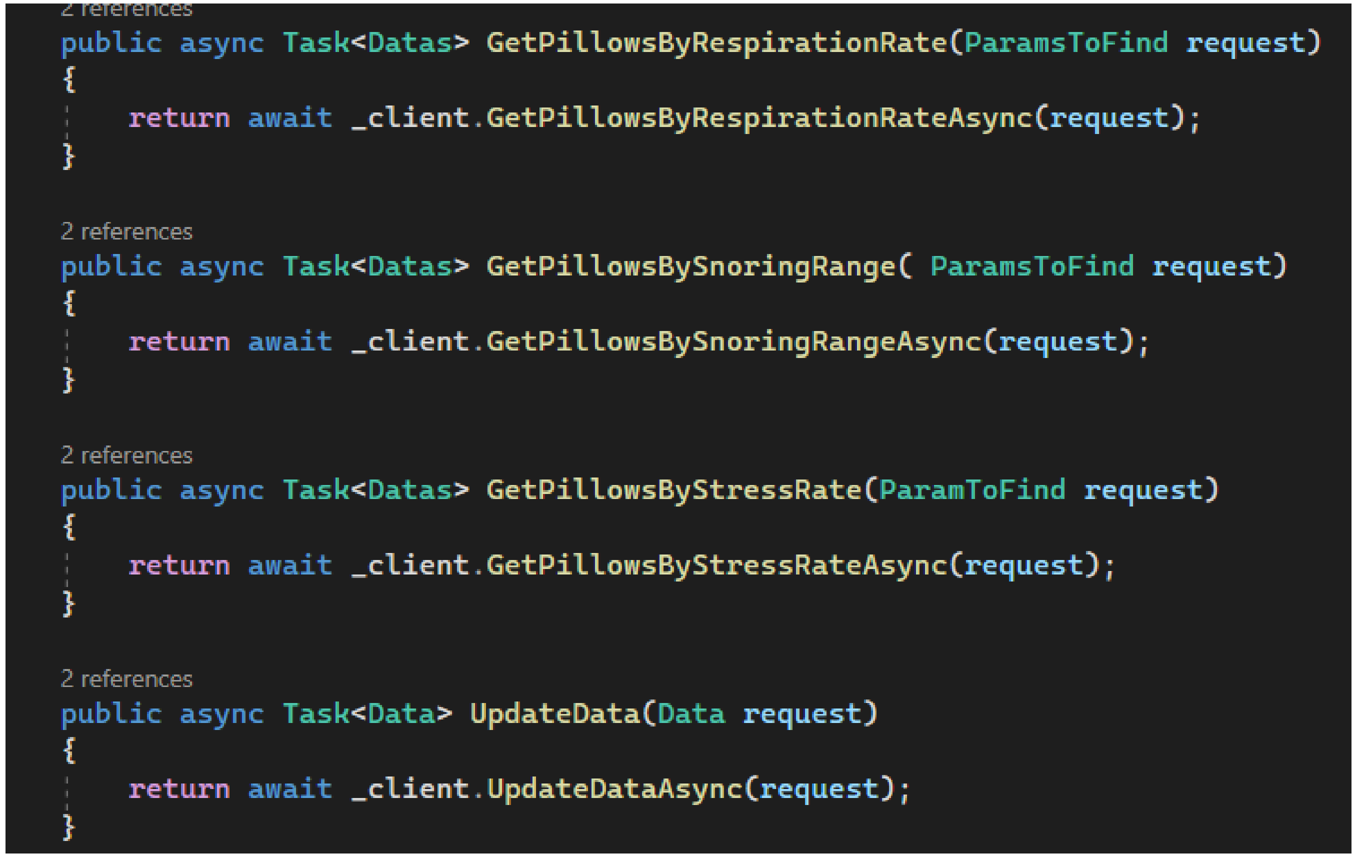


```
builder.Services.AddControllers();
// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

builder.Services.AddGrpcClient<ServiceREST.Pillow.PillowClient>(o =>
{
    //o.Address = new Uri("http://localhost:5001");
    o.Address = new Uri("http://nest-container3:5001");

});
builder.Services.AddScoped<IPillowService, ServiceREST.Services.PillowService>();

var app = builder.Build();

// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}
```

.NET MVC API - Service

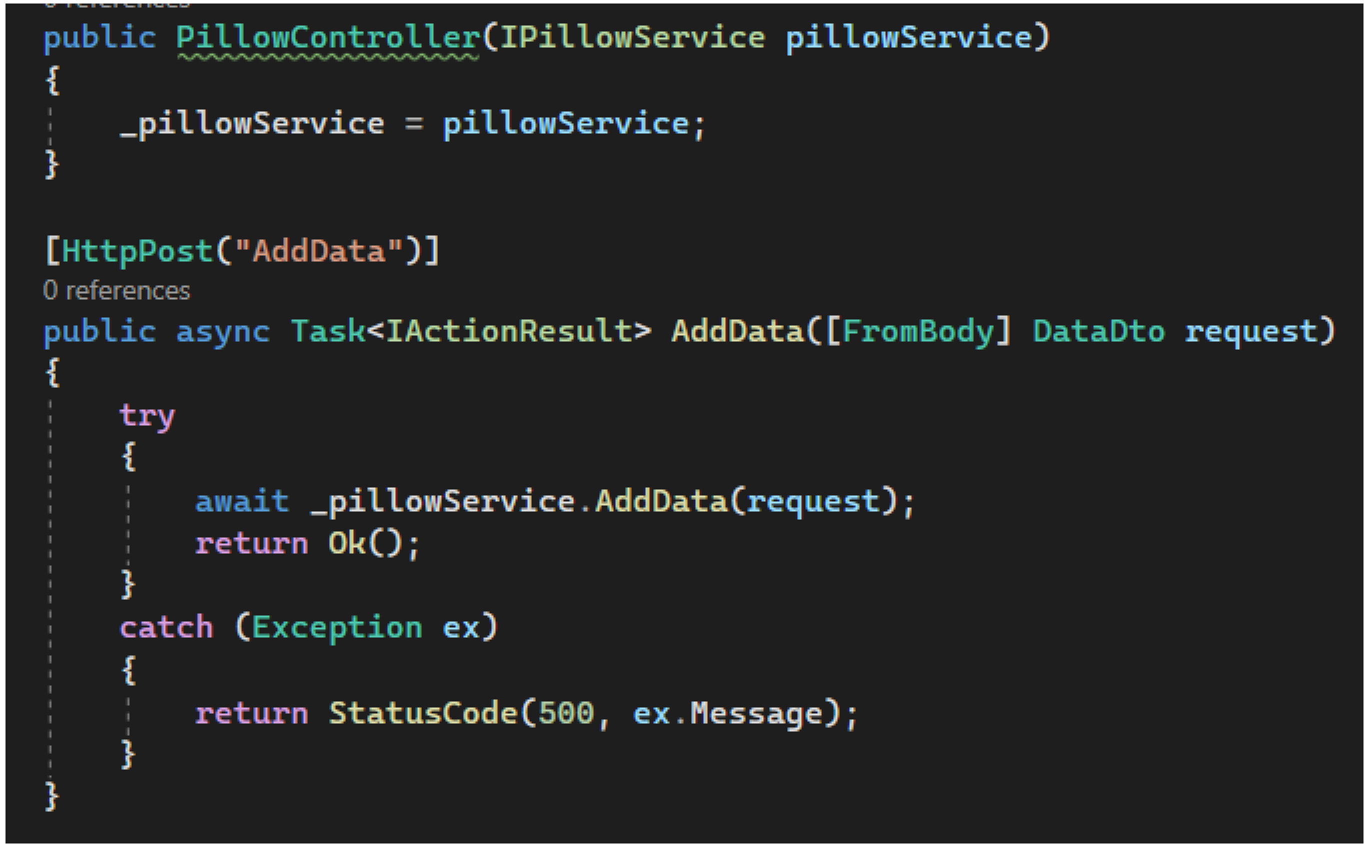


```
namespace ServiceREST.Services
{
    4 references
    public interface IPillowService
    {
        2 references
        Task<Datas> GetDatas(Empty request);
        2 references
        Task<Data> GetData(DataID request);
        2 references
        Task<Datas> GetPillowsByStressRate( ParamToFind request);
        2 references
        Task<Datas> GetPillowsByHeartRate( ParamsToFind request);
        2 references
        Task<Datas> GetPillowsBySnoringRange( ParamsToFind request);
        2 references
        Task<Datas> GetPillowsByRespirationRate(ParamsToFind request);
        2 references
        Task<Empty> AddData(DataDto request);
        2 references
        Task<Data> UpdateData(Data request);
        2 references
        Task<Empty> DeleteData(DataID request);
        2 references
        Task<AvgHeartRate> GetAvgHeartRate(Empty request);
        2 references
        Task<AvgStressLevel> GetAvgStressLevel(Empty request);
    }
}
```



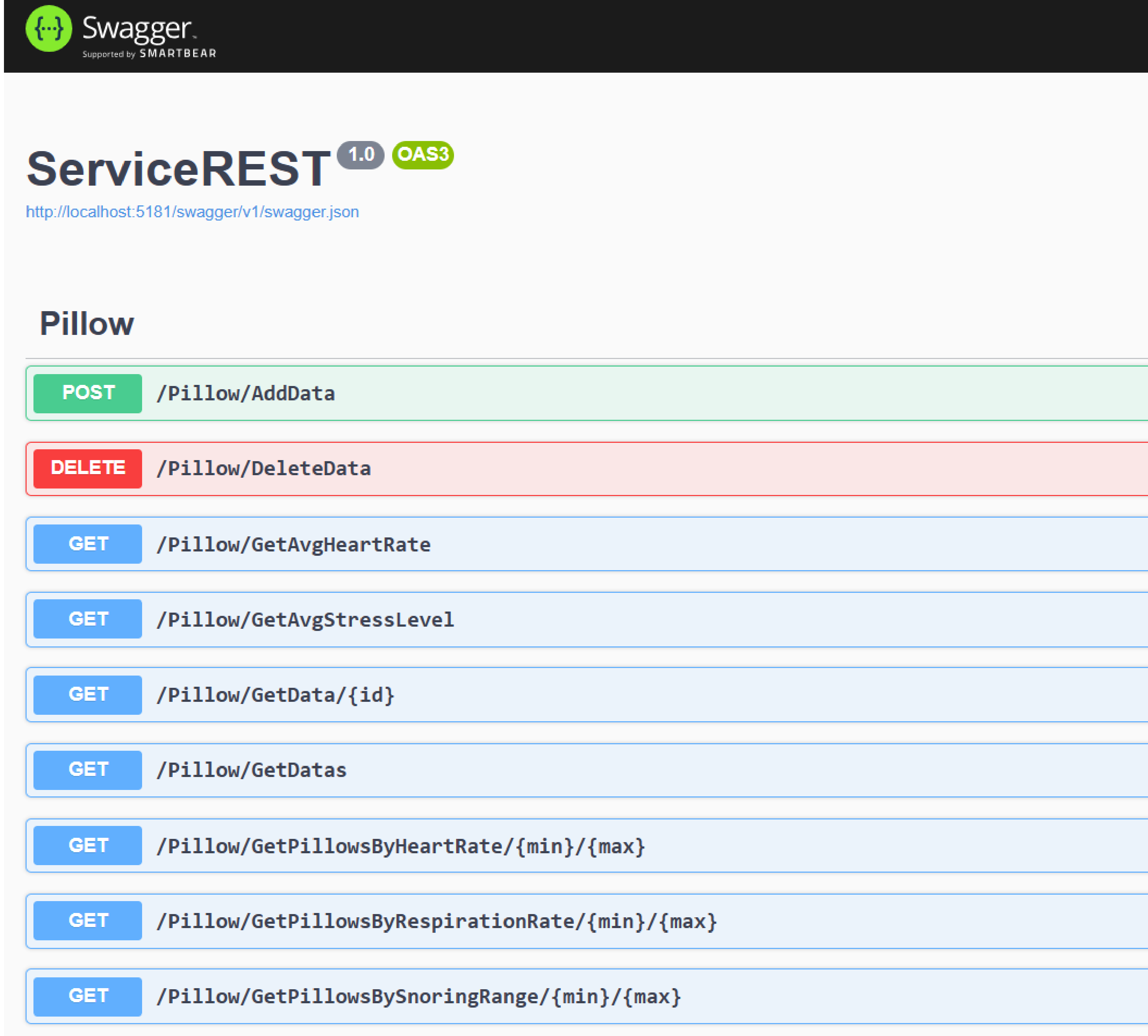
```
2 references
public async Task<Datas> GetPillowsByRespirationRate(ParamsToFind request)
{
    return await _client.GetPillowsByRespirationRateAsync(request);
}

2 references
public async Task<Datas> GetPillowsBySnoringRange( ParamsToFind request)
{
    return await _client.GetPillowsBySnoringRangeAsync(request);
}

2 references
public async Task<Datas> GetPillowsByStressRate(ParamToFind request)
{
    return await _client.GetPillowsByStressRateAsync(request);
}

2 references
public async Task<Data> UpdateData(Data request)
{
    return await _client.UpdateDataAsync(request);
}
```

.NET MVC API - Controller



```
public PillowController(IPillowService pillowService)
{
    _pillowService = pillowService;
}

[HttpPost("AddData")]
0 references
public async Task<IActionResult> AddData([FromBody] DataDto request)
{
    try
    {
        await _pillowService.AddData(request);
        return Ok();
    }
    catch (Exception ex)
    {
        return StatusCode(500, ex.Message);
    }
}
```

Swagger (OpenAPI)



ServiceREST 1.0 OAS3

http://localhost:5181/swagger/v1/swagger.json

Pillow

POST /Pillow/AddData

DELETE /Pillow/DeleteData

GET /Pillow/GetAvgHeartRate

GET /Pillow/GetAvgStressLevel

GET /Pillow/GetData/{id}

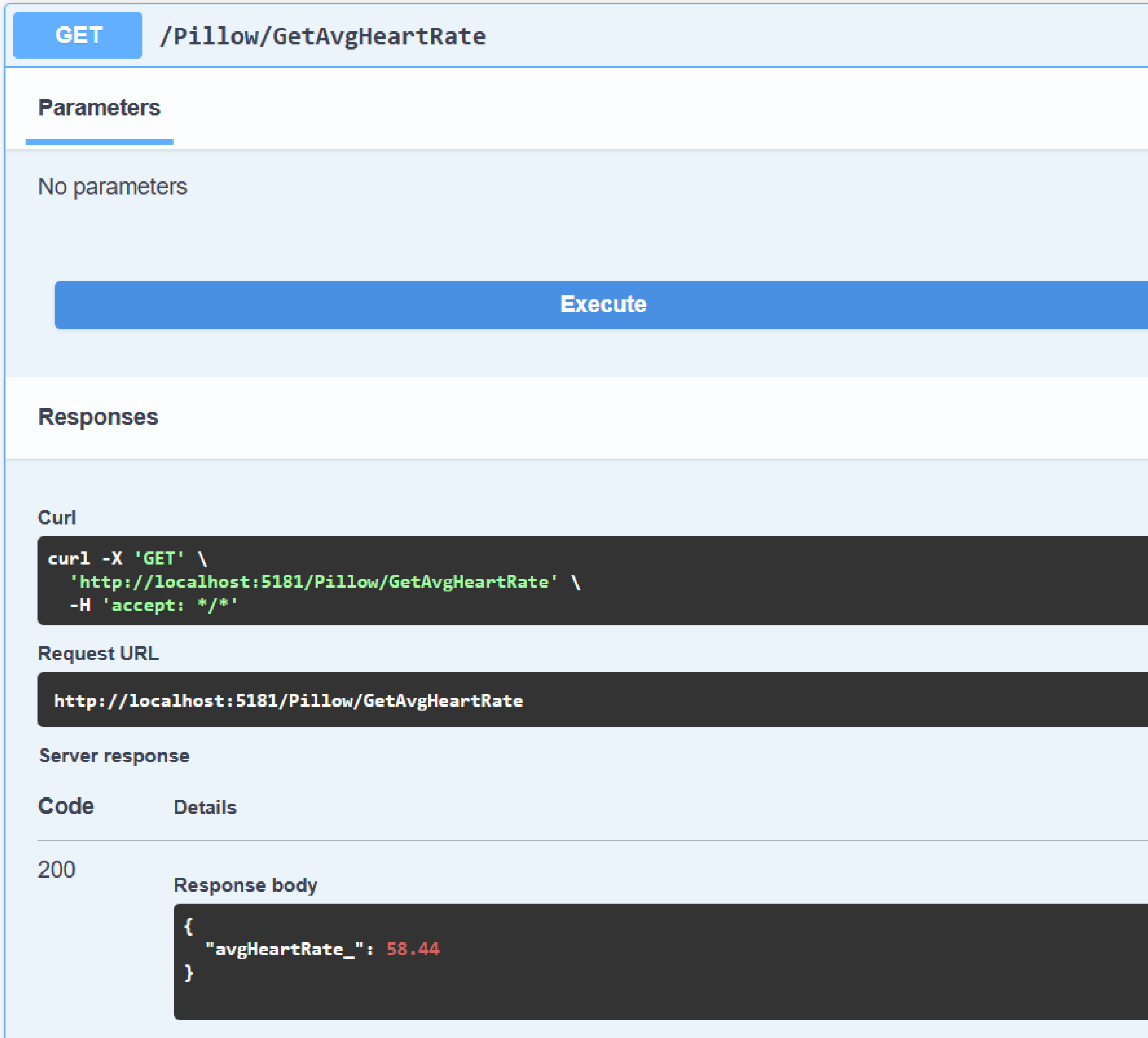
GET /Pillow/GetDatas

GET /Pillow/GetPillowsByHeartRate/{min}/{max}

GET /Pillow/GetPillowsByRespirationRate/{min}/{max}

GET /Pillow/GetPillowsBySnoringRange/{min}/{max}

Swagger



/Pillow/GetAvgHeartRate

No parameters

Curl

```
curl -X 'GET' \
  'http://localhost:5181/Pillow/GetAvgHeartRate' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:5181/Pillow/GetAvgHeartRate
```

Server response

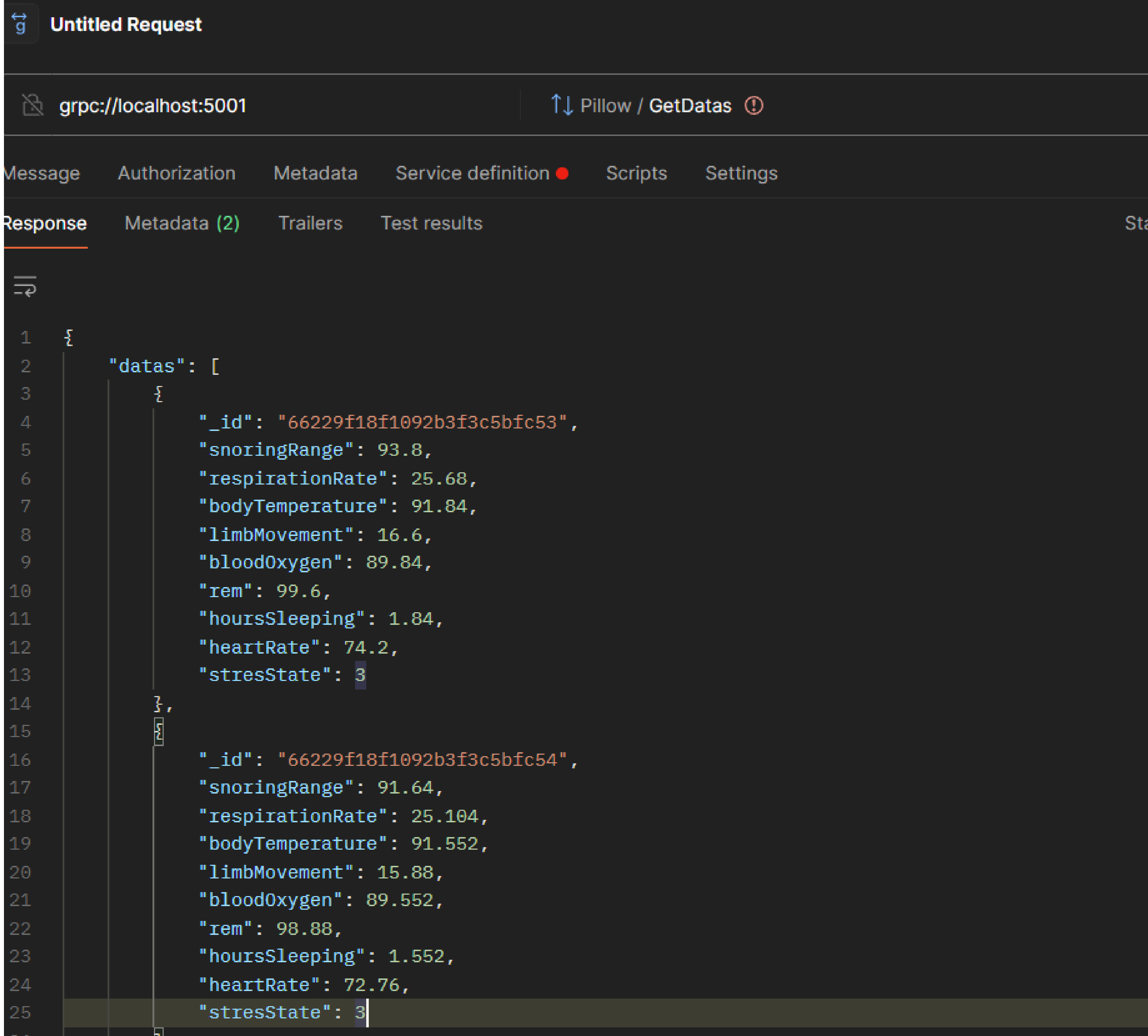
Code Details

200

Response body

```
{
  "avgHeartRate_": 58.44
}
```

Postman test



```
{
    "datas": [
        {
            "_id": "66229f18f1092b3f3c5bfc53",
            "snoringRange": 93.8,
            "respirationRate": 25.68,
            "bodyTemperature": 91.84,
            "limbMovement": 16.6,
            "bloodOxygen": 89.84,
            "rem": 99.6,
            "hoursSleeping": 1.84,
            "heartRate": 74.2,
            "stresState": 3
        },
        {
            "_id": "66229f18f1092b3f3c5bfc54",
            "snoringRange": 91.64,
            "respirationRate": 25.104,
            "bodyTemperature": 91.552,
            "limbMovement": 15.88,
            "bloodOxygen": 89.552,
            "rem": 98.88,
            "hoursSleeping": 1.552,
            "heartRate": 72.76,
            "stresState": 3
```