**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«Национальный исследовательский университет ИТМО»**

Факультет информационных технологий и программирования

Лабораторная работа № 5

*Микросервисы*

**Выполнил студент группы M32051**

Писарева Юлия Игоревна

****

**Подпись:**

**Проверил:**

Бутенко Олег Романович

Санкт-Петербург

2022

**Текст задания**

Бизнесмен прочитал статью о том, что микросервисы это круто и попросил нас разбить программу на микросервисы.

Из созданного приложения выделяются 3 микросервиса:

* Микросервис доступа к котикам
* Микросервис доступа к владельцам
* Микросервис с внешними интерфейсами

Все они являются разными приложениями. Все созданные ранее эндпоинты и авторизация переезжает на третий микросервис.

Общение между микросервисами происходит посредством RabbitMQ/Kafka.

Сервисы доступа к котикам и владельцам могут либо быть подключены к одной БД, либо иметь разные БД. Во втором случае недопустимо делать один запрос на получение данных из двух БД, запроса должно быть два (по одному на каждую).

Внимание: недопустимо передавать через RabbitMQ/Kafka JPA сущности. Рекомендуется создать отдельные оберточные классы и выделить модуль с JPA в отдельный.

**Решение с комментариями**

package com.kotiki.microservices.cats.consumers;

import com.kotiki.core.entities.Cat;

import com.kotiki.dtos.CatDto;

import com.kotiki.dtos.OwnerDto;

import com.kotiki.infrastructure.services.InfrastructureCatService;

import com.kotiki.microservices.shared.messages.cats.\*;

import com.kotiki.microservices.shared.models.CreateCatModel;

import org.springframework.amqp.rabbit.annotation.RabbitListener;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.messaging.handler.annotation.Payload;

import org.springframework.stereotype.Component;

import java.time.LocalDateTime;

import java.util.List;

@Component

public class CatConsumer {

private final InfrastructureCatService catService;

@Autowired

public CatConsumer(InfrastructureCatService catService) {

this.catService = catService;

}

@RabbitListener(queues = CreateCat.NAME)

public void CreateCat(@Payload CreateCat message) {

CreateCatModel model = message.Model();

var cat = new Cat(model.getCatName(), LocalDateTime.now(), model.getBreed(), model.getColor());

catService.addToDatabase(cat);

}

@RabbitListener(queues = GetAllCats.NAME)

public List<CatDto> GetAllCats(@Payload GetAllCats message) {

return catService.getAll().stream().map(CatDto::new).toList();

}

@RabbitListener(queues = GetCatById.NAME)

public CatDto GetCatById(@Payload GetCatById message) { return new CatDto(catService.getById(message.Id())); }

@RabbitListener(queues = GetCatOwner.NAME)

public OwnerDto GetCatOwner(@Payload GetCatOwner message) {

var cat = catService.getById(message.Id());

return new OwnerDto(cat.getOwner());

}

@RabbitListener(queues = GetCatsFriends.NAME)

public List<CatDto> GetCatsFriends(@Payload GetCatsFriends message) {

var cat = catService.getById(message.Id());

return cat.getFriends().stream().map(CatDto::new).toList();

}

@RabbitListener(queues = GetCatsWithColor.NAME)

public List<CatDto> GetCatsWithColor(@Payload GetCatsWithColor message) {

return catService.getAll().stream().filter(c -> c.getColor() == message.Color()).map(CatDto::new).toList();

}

}

package com.kotiki.microservices.cats;

import com.kotiki.core.entities.Cat;

import com.kotiki.core.services.CatService;

import com.kotiki.dataAccess.daos.CatDao;

import com.kotiki.infrastructure.services.InfrastructureCatService;

import com.kotiki.microservices.shared.configurations.CatsQueueConfiguration;

import com.kotiki.microservices.shared.configurations.DataSourceConfiguration;

import org.springframework.amqp.rabbit.annotation.EnableRabbit;

import org.springframework.amqp.rabbit.connection.CachingConnectionFactory;

import org.springframework.amqp.rabbit.connection.ConnectionFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.domain.EntityScan;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Import;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

@SpringBootApplication

@EntityScan(basePackageClasses = {Cat.class})

@EnableJpaRepositories(basePackageClasses = {CatDao.class})

@Import({CatsQueueConfiguration.class, DataSourceConfiguration.class})

@EnableRabbit

public class Main {

public static void main(String[] args) {

var app = new SpringApplication(Main.class);

app.run(args);

}

@Bean

public CatService catService() { return new CatService(); }

@Bean

public InfrastructureCatService infrastructureCatService(CatDao catDao) {

return new InfrastructureCatService(catService(), catDao);

}

@Bean

public ConnectionFactory connectionFactory() { return new CachingConnectionFactory("localhost"); }

}

package com.kotiki.microservices.external.infrastructure.configurations;

import com.kotiki.microservices.external.infrastructure.services.UserService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;

import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

@Configuration

@EnableWebSecurity

public class SecurityConfiguration extends WebSecurityConfigurerAdapter {

@Autowired

private UserService userService;

@Autowired

private BCryptPasswordEncoder bCryptPasswordEncoder;

@Autowired

private void configureGlobal(AuthenticationManagerBuilder auth) throws Exception {

auth.userDetailsService(userService).passwordEncoder(bCryptPasswordEncoder);

}

@Override

protected void configure(HttpSecurity httpSecurity) throws Exception {

httpSecurity

.csrf().disable()

.authorizeRequests()

.antMatchers("/owner/\*\*").hasAnyAuthority("OWNER", "ADMIN")

.antMatchers("/cats/\*\*").hasAuthority("ADMIN")

.antMatchers("/admin/\*\*").hasAuthority("ADMIN")

.antMatchers("/register").permitAll()

.and()

.formLogin()

.loginProcessingUrl("/login")

.defaultSuccessUrl("/")

.permitAll()

.and()

.logout()

.logoutUrl("/logout")

.deleteCookies("JSESSIONID")

.permitAll();

}

}

package com.kotiki.microservices.external.presentation.controllers;

import com.kotiki.core.models.Color;

import com.kotiki.dtos.CatDto;

import com.kotiki.dtos.OwnerDto;

import com.kotiki.microservices.shared.messages.cats.\*;

import com.kotiki.microservices.shared.models.CreateCatModel;

import org.springframework.amqp.rabbit.core.RabbitTemplate;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("cats")

public class CatController {

private final RabbitTemplate rabbitTemplate;

@Autowired

public CatController(RabbitTemplate rabbitTemplate) { this.rabbitTemplate = rabbitTemplate; }

@GetMapping("all")

public List<CatDto> getAllCats() {

var message = new GetAllCats();

Object response = rabbitTemplate.convertSendAndReceive(GetAllCats.NAME, message);

return (List<CatDto>) response;

}

@GetMapping("{id}")

public CatDto getCatWithId(@PathVariable Long id) {

var message = new GetCatById(id);

Object response = rabbitTemplate.convertSendAndReceive(GetCatById.NAME, message);

return (CatDto) response;

}

@GetMapping("color")

public List<CatDto> getCatsWithColor(@RequestParam String color) {

var message = new GetCatsWithColor(Color.valueOf(color));

Object response = rabbitTemplate.convertSendAndReceive(GetCatsWithColor.NAME, message);

return (List<CatDto>) response;

}

@GetMapping("{id}/friends")

public List<CatDto> getCatsFriends(@PathVariable Long id) {

var message = new GetCatsFriends(id);

Object response = rabbitTemplate.convertSendAndReceive(GetCatsFriends.NAME, message);

return (List<CatDto>) response;

}

@GetMapping("{id}/owner")

public OwnerDto getOwner(@PathVariable Long id) {

var message = new GetCatOwner(id);

Object response = rabbitTemplate.convertSendAndReceive(GetCatOwner.NAME, message);

return (OwnerDto) response;

}

@PostMapping("create")

public void createCat(@RequestBody CreateCatModel model) {

var message = new CreateCat(model);

rabbitTemplate.convertAndSend(CreateCat.NAME, message);

}

}

package com.kotiki.microservices.external.presentation.controllers;

import com.kotiki.core.models.Color;

import com.kotiki.dtos.CatDto;

import com.kotiki.microservices.external.infrastructure.entities.User;

import com.kotiki.microservices.shared.messages.owners.GetUserCats;

import com.kotiki.microservices.shared.messages.owners.GetUserCatsWithBreed;

import com.kotiki.microservices.shared.messages.owners.GetUserCatsWithColor;

import org.springframework.amqp.rabbit.core.RabbitTemplate;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.core.Authentication;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("owner")

public class OwnerController {

private final RabbitTemplate rabbitTemplate;

@Autowired

public OwnerController(RabbitTemplate rabbitTemplate) { this.rabbitTemplate = rabbitTemplate; }

@GetMapping("cats")

public List<CatDto> getCats(Authentication authentication) {

var user = (User) authentication.getPrincipal();

var owner = user.getOwner();

var message = new GetUserCats(owner.getId());

Object response = rabbitTemplate.convertSendAndReceive(GetUserCats.NAME, message);

return (List<CatDto>) response;

}

@GetMapping("cats/{color}")

public List<CatDto> getColoredCats(@PathVariable String color, Authentication authentication) {

var user = (User) authentication.getPrincipal();

var owner = user.getOwner();

var message = new GetUserCatsWithColor(owner.getId(), Color.valueOf(color));

Object response = rabbitTemplate.convertSendAndReceive(GetUserCatsWithColor.NAME, message);

return (List<CatDto>) response;

}

@GetMapping("cats/{breed}")

public List<CatDto> getCatsByBreed(@PathVariable String breed, Authentication authentication) {

var user = (User) authentication.getPrincipal();

var owner = user.getOwner();

var message = new GetUserCatsWithBreed(owner.getId(), breed);

Object response = rabbitTemplate.convertSendAndReceive(GetUserCatsWithBreed.NAME, message);

return (List<CatDto>) response;

}

}

package com.kotiki.microservices.external.presentation;

import com.kotiki.core.entities.Cat;

import com.kotiki.dataAccess.daos.CatDao;

import com.kotiki.dataAccess.daos.OwnerDao;

import com.kotiki.infrastructure.services.InfrastructureOwnerService;

import com.kotiki.microservices.external.infrastructure.configurations.SecurityConfiguration;

import com.kotiki.microservices.external.infrastructure.daos.UserDao;

import com.kotiki.microservices.external.infrastructure.entities.Role;

import com.kotiki.microservices.external.infrastructure.services.UserService;

import com.kotiki.microservices.external.presentation.controllers.CatController;

import com.kotiki.microservices.shared.configurations.CatsQueueConfiguration;

import com.kotiki.microservices.shared.configurations.DataSourceConfiguration;

import com.kotiki.microservices.shared.configurations.OwnersQueueConfiguration;

import org.springframework.amqp.core.AmqpAdmin;

import org.springframework.amqp.rabbit.annotation.EnableRabbit;

import org.springframework.amqp.rabbit.connection.CachingConnectionFactory;

import org.springframework.amqp.rabbit.connection.ConnectionFactory;

import org.springframework.amqp.rabbit.core.RabbitAdmin;

import org.springframework.amqp.rabbit.core.RabbitTemplate;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.domain.EntityScan;

import org.springframework.context.annotation.\*;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

import com.kotiki.core.services.OwnerService;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

@SpringBootApplication

@EntityScan(basePackageClasses = {Cat.class, Role.class})

@EnableJpaRepositories(basePackageClasses = {UserDao.class, CatDao.class})

@ComponentScan(basePackageClasses = {UserService.class, CatController.class})

@Import({

CatsQueueConfiguration.class,

OwnersQueueConfiguration.class,

DataSourceConfiguration.class,

SecurityConfiguration.class

})

@EnableRabbit

public class Main {

public static void main(String[] args) {

var app = new SpringApplication(Main.class);

app.run(args);

}

@Bean

public BCryptPasswordEncoder bCryptPasswordEncoder() { return new BCryptPasswordEncoder(); }

@Bean

public OwnerService ownerService() { return new OwnerService(); }

@Bean

public BCryptPasswordEncoder encoder() { return new BCryptPasswordEncoder(); }

@Bean

public ConnectionFactory connectionFactory() { return new CachingConnectionFactory("localhost"); }

@Bean

public AmqpAdmin amqpAdmin() { return new RabbitAdmin(connectionFactory()); }

@Bean

public RabbitTemplate rabbitTemplate() { return new RabbitTemplate(connectionFactory()); }

@Bean

public InfrastructureOwnerService infrastructureOwnerService(CatDao catDao, OwnerDao ownerDao) {

return new InfrastructureOwnerService(ownerService(), catDao, ownerDao);

}

}

package com.kotiki.microservices.owners.consumers;

import com.kotiki.dtos.CatDto;

import com.kotiki.dtos.OwnerDto;

import com.kotiki.infrastructure.services.InfrastructureCatService;

import com.kotiki.infrastructure.services.InfrastructureOwnerService;

import com.kotiki.microservices.shared.messages.owners.\*;

import org.springframework.amqp.rabbit.annotation.RabbitListener;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.messaging.handler.annotation.Payload;

import org.springframework.stereotype.Component;

import java.util.List;

@Component

public class OwnerConsumer {

private final InfrastructureCatService catService;

private final InfrastructureOwnerService ownerService;

@Autowired

public OwnerConsumer(InfrastructureCatService catService, InfrastructureOwnerService ownerService) {

this.catService = catService;

this.ownerService = ownerService;

}

@RabbitListener(queues = GetUserCats.NAME)

public List<CatDto> GetUserCats(@Payload GetUserCats message) {

var owner = ownerService.getById(message.UserId());

return catService.getAll().stream().filter(c -> owner.equals(c.getOwner())).map(CatDto::new).toList();

}

@RabbitListener(queues = GetUserCatsWithBreed.NAME)

public List<CatDto> GetUserCatsWithBreed(@Payload GetUserCatsWithBreed message) {

var owner = ownerService.getById(message.UserId());

return catService.getAll().stream()

.filter(c -> owner.equals(c.getOwner()) && c.getBreed().equals(message.Breed()))

.map(CatDto::new)

.toList();

}

@RabbitListener(queues = GetUserCatsWithColor.NAME)

public List<CatDto> GetUserCatsWithColor(@Payload GetUserCatsWithColor message) {

var owner = ownerService.getById(message.UserId());

return catService.getAll().stream()

.filter(c -> owner.equals(c.getOwner()) && c.getColor().equals(message.Color()))

.map(CatDto::new)

.toList();

}

@RabbitListener(queues = GetAllOwners.NAME)

public List<OwnerDto> GetAllOwners(@Payload GetAllOwners message) {

return ownerService.getAll().stream().map(OwnerDto::new).toList();

}

@RabbitListener(queues = GetOwnerWithId.NAME)

public OwnerDto GetOwnerWithId(@Payload GetOwnerWithId message) {

return new OwnerDto(ownerService.getById(message.Id()));

}

}

package com.kotiki.microservices.owners;

import com.kotiki.core.entities.Cat;

import com.kotiki.core.services.CatService;

import com.kotiki.core.services.OwnerService;

import com.kotiki.dataAccess.daos.CatDao;

import com.kotiki.dataAccess.daos.OwnerDao;

import com.kotiki.infrastructure.services.InfrastructureCatService;

import com.kotiki.infrastructure.services.InfrastructureOwnerService;

import com.kotiki.microservices.shared.configurations.DataSourceConfiguration;

import com.kotiki.microservices.shared.configurations.OwnersQueueConfiguration;

import org.springframework.amqp.rabbit.annotation.EnableRabbit;

import org.springframework.amqp.rabbit.connection.CachingConnectionFactory;

import org.springframework.amqp.rabbit.connection.ConnectionFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.domain.EntityScan;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Import;

import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

@SpringBootApplication

@EntityScan(basePackageClasses = {Cat.class})

@EnableJpaRepositories(basePackageClasses = {CatDao.class})

@Import({OwnersQueueConfiguration.class, DataSourceConfiguration.class})

@EnableRabbit

public class Main {

public static void main(String[] args) {

var app = new SpringApplication(Main.class);

app.run(args);

}

@Bean

public OwnerService ownerService() { return new OwnerService(); }

@Bean

public CatService catService() { return new CatService(); }

@Bean

public InfrastructureCatService infrastructureCatService(CatDao catDao) {

return new InfrastructureCatService(catService(), catDao);

}

@Bean

public InfrastructureOwnerService infrastructureOwnerService(CatDao catDao, OwnerDao ownerDao) {

return new InfrastructureOwnerService(ownerService(), catDao, ownerDao);

}

@Bean

public ConnectionFactory connectionFactory() { return new CachingConnectionFactory("localhost"); }

}

package com.kotiki.microservices.shared.configurations;

import com.kotiki.microservices.shared.messages.cats.\*;

import org.springframework.amqp.core.Queue;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class CatsQueueConfiguration {

@Bean

public Queue queueCreateCat() { return new Queue(CreateCat.NAME); }

@Bean

public Queue queueGetAllCats() { return new Queue(GetAllCats.NAME); }

@Bean

public Queue queueGetCatById() { return new Queue(GetCatById.NAME); }

@Bean

public Queue queueGetCatOwner() { return new Queue(GetCatOwner.NAME); }

@Bean

public Queue queueGetCatsFriends() { return new Queue(GetCatsFriends.NAME); }

@Bean

public Queue queueGetCatsWithColor() { return new Queue(GetCatsWithColor.NAME); }

}

package com.kotiki.microservices.shared.configurations;

import org.springframework.boot.jdbc.DataSourceBuilder;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import javax.sql.DataSource;

@Configuration

public class DataSourceConfiguration {

@Bean

public DataSource datasource() {

return DataSourceBuilder.create()

.driverClassName("org.postgresql.Driver")

.url("jdbc:postgresql://localhost:5432/kotiki-java")

.username("postgres")

.password("1234567890")

.build();

}

}

package com.kotiki.microservices.shared.configurations;

import com.kotiki.microservices.shared.messages.owners.\*;

import org.springframework.amqp.core.Queue;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class OwnersQueueConfiguration {

@Bean

public Queue queueGetUserCats() { return new Queue(GetUserCats.NAME); }

@Bean

public Queue queueGetUserCatsWithBreed() { return new Queue(GetUserCatsWithBreed.NAME); }

@Bean

public Queue queueGetUserCatsWithColor() { return new Queue(GetUserCatsWithColor.NAME); }

@Bean

public Queue queueGetAllOwners() { return new Queue(GetAllOwners.NAME); }

@Bean

public Queue queueGetOwnerWithId() { return new Queue(GetOwnerWithId.NAME); }

}

package com.kotiki.microservices.shared.messages.cats;

import com.kotiki.microservices.shared.models.CreateCatModel;

import java.io.Serializable;

public record CreateCat(CreateCatModel Model) implements Serializable {

public static final String NAME = "CreateCat";

}

package com.kotiki.microservices.shared.messages.cats;

import java.io.Serializable;

public record GetAllCats() implements Serializable {

public static final String NAME = "GetAllCats";

}

package com.kotiki.microservices.shared.messages.cats;

import java.io.Serializable;

public record GetCatById(Long Id) implements Serializable {

public static final String NAME = "GetCatById";

}

package com.kotiki.microservices.shared.messages.cats;

import java.io.Serializable;

public record GetCatOwner(Long Id) implements Serializable {

public static final String NAME = "GetCatOwner";

}

package com.kotiki.microservices.shared.messages.cats;

import java.io.Serializable;

public record GetCatsFriends(Long Id) implements Serializable {

public static final String NAME = "GetCatsFriends";

}

package com.kotiki.microservices.shared.messages.cats;

import com.kotiki.core.models.Color;

import java.io.Serializable;

public record GetCatsWithColor(Color Color) implements Serializable {

public static final String NAME = "GetCatsWithColor";

}

package com.kotiki.microservices.shared.messages.owners;

import java.io.Serializable;

public record GetAllOwners() implements Serializable {

public static final String NAME = "GetAllOwners";

}

package com.kotiki.microservices.shared.messages.owners;

import java.io.Serializable;

public record GetOwnerWithId(Long Id) implements Serializable {

public static final String NAME = "GetOwnerWithId";

}

package com.kotiki.microservices.shared.messages.owners;

import java.io.Serializable;

public record GetUserCats(Long UserId) implements Serializable {

public static final String NAME = "GetUserCats";

}

package com.kotiki.microservices.shared.messages.owners;

import java.io.Serializable;

public record GetUserCatsWithBreed(Long UserId, String Breed) implements Serializable {

public static final String NAME = "GetUserCatsWithBreed";

}

package com.kotiki.microservices.shared.messages.owners;

import com.kotiki.core.models.Color;

import java.io.Serializable;

public record GetUserCatsWithColor(Long UserId, Color Color) implements Serializable {

public static final String NAME = "GetUserCatsWithColor";

}

package com.kotiki.microservices.shared.models;

import com.kotiki.core.models.Color;

import java.io.Serializable;

public class CreateCatModel implements Serializable {

private String catName;

private String breed;

private Color color;

public String getCatName() { return catName; }

public String getBreed() { return breed; }

public Color getColor() { return color; }

}