**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №5

по дисциплине «**Технологии программирования**»

Автор: Кошкин Михаил Сергеевич

Факультет: ИТиП

Группа: М32051



Санкт-Петербург 2022

Описание:

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

Из созданного приложения выделяются три 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 (на выбор студента).

Сервисы доступа к котикам и доступа к 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 в каждую).

Внимание: недопустимо передавать через RabbitMQ/Kafka JPA сущности. Рекомендуется создать отдельные оберточные классы.

Рекомендуется выделить модуль с JPA сущностями в отдельный подключаемый модуль.

Исходный код:

Удален файл kotiki-java/api/pom.xml

Удален файл kotiki-java/api/src/main/java/org/itmo/Application.java

Удален файл kotiki-java/api/src/test/java/org/itmo/services/CatServiceTest.java

Удален файл kotiki-java/api/src/test/java/org/itmo/services/OwnerServiceTest.java

Добавлен файл kotiki-java/cats-access/pom.xml:

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.6.5</version>

<relativePath/>

</parent>

<modelVersion>4.0.0</modelVersion>

<artifactId>cats-access</artifactId>

<properties>

<maven.compiler.source>11</maven.compiler.source>

<maven.compiler.target>11</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>org.itmo</groupId>

<artifactId>domain-jpa</artifactId>

<version>1.0</version>

<scope>compile</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>5.8.2</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

<version>2.6.7</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

<version>2.6.7</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<version>2.6.7</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<version>2.6.7</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-amqp</artifactId>

<version>2.6.7</version>

</dependency>

<dependency>

<groupId>org.apache.tomcat.embed</groupId>

<artifactId>tomcat-embed-jasper</artifactId>

<version>10.1.0-M12</version>

</dependency>

<dependency>

<groupId>org.glassfish</groupId>

<artifactId>javax.faces</artifactId>

<version>2.4.0</version>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.24</version>

<scope>provided</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.0</version>

</plugin>

<plugin>

<artifactId>maven-failsafe-plugin</artifactId>

<version>2.22.0</version>

</plugin>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<version>2.6.6</version>

</plugin>

</plugins>

</build>

</project>

Добавлен файл kotiki-java/cats-access/src/main/java/org/itmo/Application.java:

package org.itmo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

Добавлен файл kotiki-java/cats-access/src/main/java/org/itmo/configs/CatsRabbitConfig.java:

package org.itmo.configs;

import org.springframework.amqp.core.Queue;

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

import org.springframework.amqp.support.converter.Jackson2JsonMessageConverter;

import org.springframework.amqp.support.converter.MessageConverter;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@EnableRabbit

@Configuration

public class CatsRabbitConfig {

public static final String CATS\_ACCESS\_QUEUE = "cats-access-queue";

public static final String CATS\_FULL\_ACCESS\_QUEUE = "cats-full-access-queue";

public static final String CATS\_SAVE\_QUEUE = "cats-save-queue";

public static final String CATS\_DELETE\_QUEUE = "cats-delete-queue";

public static final String CATS\_EXIST\_QUEUE = "cats-exist-queue";

@Bean

public Queue catsAccessQueue() {

return new Queue(CATS\_ACCESS\_QUEUE);

}

@Bean

public Queue catsFullAccessQueue() {

return new Queue(CATS\_FULL\_ACCESS\_QUEUE);

}

@Bean

public Queue catsSaveQueue() {

return new Queue(CATS\_SAVE\_QUEUE);

}

@Bean

public Queue catsDeleteQueue() {

return new Queue(CATS\_DELETE\_QUEUE);

}

@Bean

public Queue catsExistQueue() {

return new Queue(CATS\_EXIST\_QUEUE);

}

@Bean

public MessageConverter messageConverter() {

return new Jackson2JsonMessageConverter();

}

}

Изменен файл kotiki-java/dal/src/main/java/org/itmo/dao/CatDao.java -> kotiki-java/cats-access/src/main/java/org/itmo/dao/CatDao.java:

package org.itmo.dao;

+import org.itmo.entities.Cat;

import org.itmo.enums.Color;

-import org.itmo.models.Cat;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

Добавлен файл kotiki-java/cats-access/src/main/java/org/itmo/listeners/CatListener.java:

package org.itmo.listeners;

import lombok.RequiredArgsConstructor;

import org.itmo.dto.CatDto;

import org.itmo.dto.rabbit.BreedDto;

import org.itmo.dto.rabbit.ColorDto;

import org.itmo.dto.rabbit.OwnerIdDto;

import org.itmo.services.AbstractCatService;

import org.itmo.utils.CatConverter;

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

import org.springframework.stereotype.Component;

import java.util.List;

import java.util.stream.Collectors;

import static org.itmo.configs.CatsRabbitConfig.\*;

@Component

@RequiredArgsConstructor

public class CatListener {

private final AbstractCatService service;

private final CatConverter converter;

@RabbitListener(queues = CATS\_FULL\_ACCESS\_QUEUE)

public List<CatDto> getAll(OwnerIdDto ownerId) {

return (ownerId.getId() == null

? service.getAll()

: service.getByOwnerId(ownerId.getId()))

.stream()

.map(converter::toDto)

.collect(Collectors.toList());

}

@RabbitListener(queues = CATS\_ACCESS\_QUEUE)

public CatDto get(int id) {

return converter.toDto(service.get(id));

}

@RabbitListener(queues = CATS\_ACCESS\_QUEUE)

public List<CatDto> getByBreed(BreedDto breedDto) {

return (breedDto.getOwnerId() == null

? service.getByBreed(breedDto.getBreed())

: service.getByBreedAndOwnerId(breedDto.getBreed(), breedDto.getOwnerId()))

.stream()

.map(converter::toDto)

.collect(Collectors.toList());

}

@RabbitListener(queues = CATS\_ACCESS\_QUEUE)

public List<CatDto> getByColor(ColorDto colorDto) {

return (colorDto.getOwnerId() == null

? service.getByColor(colorDto.getColor())

: service.getByColorAndOwnerId(colorDto.getColor(), colorDto.getOwnerId()))

.stream()

.map(converter::toDto)

.collect(Collectors.toList());

}

@RabbitListener(queues = CATS\_SAVE\_QUEUE)

public void save(CatDto cat) {

service.save(converter.toModel(cat));

}

@RabbitListener(queues = CATS\_DELETE\_QUEUE)

public void delete(int id) {

service.delete(id);

}

@RabbitListener(queues = CATS\_EXIST\_QUEUE)

public boolean exists(int id) {

return service.exists(id);

}

}

Изменен файл kotiki-java/api/src/main/java/org/itmo/services/AbstractCatService.java -> kotiki-java/cats-access/src/main/java/org/itmo/services/AbstractCatService.java:

package org.itmo.services;

+import org.itmo.entities.Cat;

import org.itmo.enums.Color;

-import org.itmo.models.Cat;

import java.util.List;

Изменен файл kotiki-java/api/src/main/java/org/itmo/services/CatService.java -> kotiki-java/cats-access/src/main/java/org/itmo/services/CatService.java:

import lombok.RequiredArgsConstructor;

import org.itmo.dao.CatDao;

+import org.itmo.entities.Cat;

import org.itmo.enums.Color;

-import org.itmo.models.Cat;

import org.itmo.tools.DalException;

import org.springframework.stereotype.Service;

Изменен файл kotiki-java/api/src/main/java/org/itmo/utils/CatConverter.java -> kotiki-java/cats-access/src/main/java/org/itmo/utils/CatConverter.java:

import lombok.RequiredArgsConstructor;

import org.itmo.dto.CatDto;

-import org.itmo.models.Cat;

-import org.itmo.services.AbstractCatService;

-import org.itmo.services.AbstractOwnerService;

+import org.itmo.entities.Cat;

+import org.itmo.services.CatService;

import org.springframework.stereotype.Component;

import java.util.stream.Collectors;

@@ -13,8 +12,7 @@

@RequiredArgsConstructor

public class CatConverter {

- private final AbstractCatService catService;

- private final AbstractOwnerService ownerService;

+ private final CatService catService;

public CatDto toDto(Cat cat) {

return CatDto.builder()

public CatDto toDto(Cat cat) {

.birthDate(cat.getBirthDate())

.breed(cat.getBreed())

.color(cat.getColor())

- .ownerId(cat.getOwner() == null ? null : cat.getOwner().getId())

.friendIds(cat.getFriends()

.stream()

.map(Cat::getId)

public Cat toModel(CatDto cat) {

.breed(cat.getBreed())

.color(cat.getColor());

- if (cat.getOwnerId() != null)

- catBuilder.owner(ownerService.get(cat.getOwnerId()));

+ /\*Cat dbCat = catService.get(cat.getId());

+ if (dbCat != null)

+ catBuilder.owner(dbCat.getOwner());\*/

if (cat.getFriendIds() != null)

catBuilder.friends(cat.getFriendIds()

Изменен файл kotiki-java/api/src/main/resources/application.properties -> kotiki-java/cats-access/src/main/resources/application.properties:

spring.jpa.properties.hibernate.event.merge.entity\_copy\_observer=allow

# Hibernate ddl auto (create, create-drop, validate, update)

spring.jpa.hibernate.ddl-auto = update

-server.port=9090

+server.port=9091

+

+spring.rabbitmq.host=localhost

+spring.rabbitmq.username=guest

+spring.rabbitmq.password=guest

+spring.rabbitmq.port=5672

Удален файл kotiki-java/dal/src/main/java/org/itmo/Main.java

Добавлен файл kotiki-java/domain-jpa/pom.xml:

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.6.5</version>

<relativePath/>

</parent>

<modelVersion>4.0.0</modelVersion>

<groupId>org.itmo</groupId>

<artifactId>domain-jpa</artifactId>

<version>1.0</version>

<dependencies>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.24</version>

<scope>compile</scope>

</dependency>

<dependency>

<groupId>org.postgresql</groupId>

<artifactId>postgresql</artifactId>

<version>42.3.4</version>

</dependency>

<dependency>

<groupId>org.itmo</groupId>

<artifactId>domain</artifactId>

<version>1.0</version>

</dependency>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>6.0.1.Final</version>

<scope>compile</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<version>5.8.2</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>5.8.2</version>

<scope>test</scope>

</dependency>

</dependencies>

<properties>

<maven.compiler.source>11</maven.compiler.source>

<maven.compiler.target>11</maven.compiler.target>

</properties>

</project>

Изменен файл kotiki-java/dal/src/main/java/org/itmo/models/Cat.java -> kotiki-java/domain-jpa/src/main/java/org/itmo/entities/Cat.java:

-package org.itmo.models;

+package org.itmo.entities;

import lombok.\*;

import org.hibernate.Hibernate;

public class Cat {

@ToString.Exclude

private Owner owner;

- @ManyToMany

+ @ManyToMany(fetch = FetchType.EAGER)

@Cascade({

CascadeType.PERSIST,

CascadeType.MERGE,

public class Cat {

@ToString.Exclude

private List<Cat> friends = new ArrayList<>();

- @ManyToMany

+ @ManyToMany(fetch = FetchType.EAGER)

@JoinTable(name = "cats\_cats",

joinColumns = @JoinColumn(name = "friend\_id"),

inverseJoinColumns = @JoinColumn(name = "cat\_id")

Изменен файл kotiki-java/dal/src/main/java/org/itmo/models/Owner.java -> kotiki-java/domain-jpa/src/main/java/org/itmo/entities/Owner.java:

-package org.itmo.models;

+package org.itmo.entities;

import lombok.\*;

import org.hibernate.Hibernate;

import org.hibernate.annotations.Cascade;

import org.hibernate.annotations.CascadeType;

-import org.itmo.models.auth.User;

+import org.itmo.entities.auth.User;

import javax.persistence.\*;

import java.time.LocalDate;

Изменен файл kotiki-java/dal/src/main/java/org/itmo/models/auth/Role.java -> kotiki-java/domain-jpa/src/main/java/org/itmo/entities/auth/Role.java:

-package org.itmo.models.auth;

+package org.itmo.entities.auth;

-import lombok.\*;

import org.hibernate.Hibernate;

+import lombok.\*;

import javax.persistence.\*;

import java.util.Objects;

Изменен файл kotiki-java/dal/src/main/java/org/itmo/models/auth/User.java -> kotiki-java/domain-jpa/src/main/java/org/itmo/entities/auth/User.java:

-package org.itmo.models.auth;

+package org.itmo.entities.auth;

-import lombok.\*;

import org.hibernate.Hibernate;

-import org.itmo.models.Owner;

+import lombok.\*;

+import org.itmo.entities.Owner;

import javax.persistence.\*;

import java.util.List;

public class User {

@NonNull

private String password;

- @ManyToMany

+ @ManyToMany(fetch = FetchType.EAGER)

@JoinTable(name = "users\_roles",

joinColumns = @JoinColumn(name = "user\_id"),

inverseJoinColumns = @JoinColumn(name = "role\_id"))

Изменен файл kotiki-java/dal/src/main/java/org/itmo/tools/DalException.java -> kotiki-java/domain-jpa/src/main/java/org/itmo/tools/DalException.java:

+<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

- <parent>

- <artifactId>kotiki-java</artifactId>

- <groupId>org.itmo</groupId>

- <version>1.0</version>

- </parent>

<modelVersion>4.0.0</modelVersion>

- <artifactId>dal</artifactId>

+ <groupId>org.itmo</groupId>

+ <artifactId>domain</artifactId>

+ <version>1.0</version>

+ <dependencies>

+ <dependency>

+ <groupId>org.projectlombok</groupId>

+ <artifactId>lombok</artifactId>

+ <version>1.18.24</version>

+ <scope>compile</scope>

+ </dependency>

+ </dependencies>

<properties>

<maven.compiler.source>11</maven.compiler.source>

<maven.compiler.target>11</maven.compiler.target>

</properties>

- <dependencies>

-

- <dependency>

- <groupId>org.postgresql</groupId>

- <artifactId>postgresql</artifactId>

- <version>42.3.3</version>

- </dependency>

-

- </dependencies>

-

</project>

Изменен файл kotiki-java/api/src/main/java/org/itmo/dto/CatDto.java -> kotiki-java/domain/src/main/java/org/itmo/dto/CatDto.java:

package org.itmo.dto;

+import lombok.AllArgsConstructor;

import lombok.Builder;

import lombok.Data;

+import lombok.NoArgsConstructor;

import org.itmo.enums.Color;

import java.time.LocalDate;

@@ -9,6 +11,8 @@

@Data

@Builder

+@NoArgsConstructor

+@AllArgsConstructor

public class CatDto {

private int id;

public class CatDto {

private Color color;

- private Integer ownerId;

-

private List<Integer> friendIds;

}

Изменен файл kotiki-java/api/src/main/java/org/itmo/dto/OwnerDto.java -> kotiki-java/domain/src/main/java/org/itmo/dto/OwnerDto.java:

package org.itmo.dto;

+import lombok.AllArgsConstructor;

import lombok.Builder;

import lombok.Data;

-

-import java.util.List;

+import lombok.NoArgsConstructor;

@Data

@Builder

+@NoArgsConstructor

+@AllArgsConstructor

public class OwnerDto {

private int id;

private String name;

-

- private List<Integer> catIds;

}

Изменен файл kotiki-java/api/src/main/java/org/itmo/dto/UserDto.java -> kotiki-java/domain/src/main/java/org/itmo/dto/UserDto.java:

package org.itmo.dto;

+import lombok.AllArgsConstructor;

import lombok.Builder;

import lombok.Data;

+import lombok.NoArgsConstructor;

import java.util.List;

@Data

@Builder

+@NoArgsConstructor

+@AllArgsConstructor

public class UserDto {

private int id;

Добавлен файл kotiki-java/domain/src/main/java/org/itmo/dto/rabbit/BreedDto.java:

package org.itmo.dto.rabbit;

import lombok.AllArgsConstructor;

import lombok.Builder;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@Builder

@NoArgsConstructor

@AllArgsConstructor

public class BreedDto {

private String breed;

private Integer ownerId;

}

Добавлен файл kotiki-java/domain/src/main/java/org/itmo/dto/rabbit/ColorDto.java:

package org.itmo.dto.rabbit;

import lombok.AllArgsConstructor;

import lombok.Builder;

import lombok.Data;

import lombok.NoArgsConstructor;

import org.itmo.enums.Color;

@Data

@Builder

@NoArgsConstructor

@AllArgsConstructor

public class ColorDto {

private Color color;

private Integer ownerId;

}

Добавлен файл kotiki-java/domain/src/main/java/org/itmo/dto/rabbit/OwnerIdDto.java:

package org.itmo.dto.rabbit;

import lombok.AllArgsConstructor;

import lombok.Builder;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@Builder

@NoArgsConstructor

@AllArgsConstructor

public class OwnerIdDto {

private Integer id;

}

Изменен файл kotiki-java/dal/src/main/java/org/itmo/enums/Color.java -> kotiki-java/domain/src/main/java/org/itmo/enums/Color.java:

+<?xml version="1.0" encoding="UTF-8"?>

+<project xmlns="http://maven.apache.org/POM/4.0.0"

+ xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

+ xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

+ <parent>

+ <groupId>org.springframework.boot</groupId>

+ <artifactId>spring-boot-starter-parent</artifactId>

+ <version>2.6.5</version>

+ <relativePath/>

+ </parent>

+ <modelVersion>4.0.0</modelVersion>

+

+ <groupId>org.itmo</groupId>

+ <artifactId>external-interafaces</artifactId>

+ <version>1.0</version>

+

+ <properties>

+ <maven.compiler.source>11</maven.compiler.source>

+ <maven.compiler.target>11</maven.compiler.target>

+ </properties>

+

+ <dependencies>

+

+ <dependency>

+ <groupId>org.junit.jupiter</groupId>

+ <artifactId>junit-jupiter-engine</artifactId>

+ <version>5.8.2</version>

+ </dependency>

+

+ <dependency>

+ <groupId>org.springframework.boot</groupId>

+ <artifactId>spring-boot-starter-data-jpa</artifactId>

+ <version>2.6.7</version>

+ </dependency>

+

+ <dependency>

+ <groupId>org.springframework.boot</groupId>

+ <artifactId>spring-boot-starter-web</artifactId>

+ <version>2.6.7</version>

+ </dependency>

+

+ <dependency>

+ <groupId>org.springframework.boot</groupId>

+ <artifactId>spring-boot-devtools</artifactId>

+ <version>2.6.7</version>

+ <scope>runtime</scope>

+ </dependency>

+

+ <dependency>

+ <groupId>org.springframework.boot</groupId>

+ <artifactId>spring-boot-starter-test</artifactId>

+ <version>2.6.7</version>

+ <scope>test</scope>

+ </dependency>

+

+ <dependency>

+ <groupId>org.springframework.boot</groupId>

+ <artifactId>spring-boot-starter-security</artifactId>

+ <version>2.6.7</version>

+ </dependency>

+

+ <dependency>

+ <groupId>org.springframework.boot</groupId>

+ <artifactId>spring-boot-starter-amqp</artifactId>

+ <version>2.6.7</version>

+ </dependency>

+

+ <dependency>

+ <groupId>org.apache.tomcat.embed</groupId>

+ <artifactId>tomcat-embed-jasper</artifactId>

+ <version>10.1.0-M12</version>

+ </dependency>

+

+ <dependency>

+ <groupId>org.glassfish</groupId>

+ <artifactId>javax.faces</artifactId>

+ <version>2.4.0</version>

+ </dependency>

+

+ <dependency>

+ <groupId>org.projectlombok</groupId>

+ <artifactId>lombok</artifactId>

+ <version>1.18.24</version>

+ <scope>provided</scope>

+ </dependency>

+ <dependency>

+ <groupId>org.itmo</groupId>

+ <artifactId>domain-jpa</artifactId>

+ <version>1.0</version>

+ <scope>compile</scope>

+ </dependency>

+

+ </dependencies>

+

+ <build>

+ <plugins>

+ <plugin>

+ <artifactId>maven-surefire-plugin</artifactId>

+ <version>2.22.0</version>

+ </plugin>

+ <plugin>

+ <artifactId>maven-failsafe-plugin</artifactId>

+ <version>2.22.0</version>

+ </plugin>

+ <plugin>

+ <groupId>org.springframework.boot</groupId>

+ <artifactId>spring-boot-maven-plugin</artifactId>

+ <version>2.6.6</version>

+ </plugin>

+ </plugins>

+ </build>

+

+</project>

Добавлен файл kotiki-java/external-interfaces/src/main/java/org/itmo/Application.java:

package org.itmo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

Добавлен файл kotiki-java/external-interfaces/src/main/java/org/itmo/configs/CatsRabbitConfig.java:

package org.itmo.configs;

import org.springframework.amqp.core.\*;

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

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

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

import org.springframework.amqp.support.converter.Jackson2JsonMessageConverter;

import org.springframework.amqp.support.converter.MessageConverter;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@EnableRabbit

@Configuration

public class CatsRabbitConfig {

public static final String CATS\_EXCHANGE = "cats-exchange";

public static final String CATS\_ACCESS\_QUEUE = "cats-access-queue";

public static final String CATS\_FULL\_ACCESS\_QUEUE = "cats-full-access-queue";

public static final String CATS\_SAVE\_QUEUE = "cats-save-queue";

public static final String CATS\_DELETE\_QUEUE = "cats-delete-queue";

public static final String CATS\_EXIST\_QUEUE = "cats-exist-queue";

public static final String CATS\_ACCESS\_ROUTING\_KEY = "cats-access-routing-key";

public static final String CATS\_FULL\_ACCESS\_ROUTING\_KEY = "cats-full-access-routing-key";

public static final String CATS\_SAVE\_ROUTING\_KEY = "cats-save-routing-key";

public static final String CATS\_DELETE\_ROUTING\_KEY = "cats-delete-routing-key";

public static final String CATS\_EXISTS\_ROUTING\_KEY = "cats-exist-routing-key";

@Bean

public TopicExchange catsExchange() {

return new TopicExchange(CATS\_EXCHANGE);

}

@Bean

public Queue catsAccessQueue() {

return new Queue(CATS\_ACCESS\_QUEUE);

}

@Bean

public Queue catsFullAccessQueue() {

return new Queue(CATS\_FULL\_ACCESS\_QUEUE);

}

@Bean

public Queue catsSaveQueue() {

return new Queue(CATS\_SAVE\_QUEUE);

}

@Bean

public Queue catsDeleteQueue() {

return new Queue(CATS\_DELETE\_QUEUE);

}

@Bean

public Queue catsExistQueue() {

return new Queue(CATS\_EXIST\_QUEUE);

}

@Bean

public Binding catsAccessBinding() {

return BindingBuilder.bind(catsAccessQueue()).to(catsExchange()).with(CATS\_ACCESS\_ROUTING\_KEY);

}

@Bean

public Binding catsFullAccessBinding() {

return BindingBuilder.bind(catsFullAccessQueue()).to(catsExchange()).with(CATS\_FULL\_ACCESS\_ROUTING\_KEY);

}

@Bean

public Binding catsSaveBinding() {

return BindingBuilder.bind(catsSaveQueue()).to(catsExchange()).with(CATS\_SAVE\_ROUTING\_KEY);

}

@Bean

public Binding catsDeleteBinding() {

return BindingBuilder.bind(catsDeleteQueue()).to(catsExchange()).with(CATS\_DELETE\_ROUTING\_KEY);

}

@Bean

public Binding catsExistBinding() {

return BindingBuilder.bind(catsExistQueue()).to(catsExchange()).with(CATS\_EXISTS\_ROUTING\_KEY);

}

@Bean

public MessageConverter messageConverter() {

return new Jackson2JsonMessageConverter();

}

@Bean

public AmqpTemplate template(ConnectionFactory connectionFactory) {

var rabbitTemplate = new RabbitTemplate(connectionFactory);

rabbitTemplate.setMessageConverter(messageConverter());

return rabbitTemplate;

}

}

Добавлен файл kotiki-java/external-interfaces/src/main/java/org/itmo/configs/OwnersRabbitConfig.java:

package org.itmo.configs;

import org.springframework.amqp.core.\*;

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

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@EnableRabbit

@Configuration

public class OwnersRabbitConfig {

public static final String OWNERS\_EXCHANGE = "owners-exchange";

public static final String OWNERS\_ACCESS\_QUEUE = "owners-access-queue";

public static final String OWNERS\_FULL\_ACCESS\_QUEUE = "owners-full-access-queue";

public static final String OWNERS\_CAT\_ACCESS\_QUEUE = "owners-cat-access-queue";

public static final String OWNERS\_SAVE\_QUEUE = "owners-save-queue";

public static final String OWNERS\_DELETE\_QUEUE = "owners-delete-queue";

public static final String OWNERS\_EXIST\_QUEUE = "owners-exist-queue";

public static final String USERS\_GET\_QUEUE = "users-get-queue";

public static final String USERS\_SAVE\_QUEUE = "users-save-queue";

public static final String USERS\_DELETE\_QUEUE = "users-delete-queue";

public static final String OWNERS\_ACCESS\_ROUTING\_KEY = "owners-access-routing-key";

public static final String OWNERS\_FULL\_ACCESS\_ROUTING\_KEY = "owners-full-access-routing-key";

public static final String OWNERS\_CAT\_ACCESS\_ROUTING\_KEY = "owners-cat-routing-key";

public static final String OWNERS\_SAVE\_ROUTING\_KEY = "owners-save-routing-key";

public static final String OWNERS\_DELETE\_ROUTING\_KEY = "owners-delete-routing-key";

public static final String OWNERS\_EXIST\_ROUTING\_KEY = "owners-exist-routing-key";

public static final String USERS\_GET\_ROUTING\_KEY = "users-get-routing-key";

public static final String USERS\_SAVE\_ROUTING\_KEY = "users-save-routing-key";

public static final String USERS\_DELETE\_ROUTING\_KEY = "users-delete-routing-key";

@Bean

public TopicExchange ownersExchange() {

return new TopicExchange(OWNERS\_EXCHANGE);

}

@Bean

public Queue ownersAccessQueue() {

return new Queue(OWNERS\_ACCESS\_QUEUE);

}

@Bean

public Queue ownersFullAccessQueue() {

return new Queue(OWNERS\_FULL\_ACCESS\_QUEUE);

}

@Bean

public Queue ownersCatAccessQueue() {

return new Queue(OWNERS\_CAT\_ACCESS\_QUEUE);

}

@Bean

public Queue ownersSaveQueue() {

return new Queue(OWNERS\_SAVE\_QUEUE);

}

@Bean

public Queue ownersDeleteQueue() {

return new Queue(OWNERS\_DELETE\_QUEUE);

}

@Bean

public Queue ownersExistQueue() {

return new Queue(OWNERS\_EXIST\_QUEUE);

}

@Bean

public Queue usersGetQueue() {

return new Queue(USERS\_GET\_QUEUE);

}

@Bean

public Queue usersSaveQueue() {

return new Queue(USERS\_SAVE\_QUEUE);

}

@Bean

public Queue usersDeleteQueue() {

return new Queue(USERS\_DELETE\_QUEUE);

}

@Bean

public Binding ownersAccessBinding() {

return BindingBuilder.bind(ownersAccessQueue()).to(ownersExchange()).with(OWNERS\_ACCESS\_ROUTING\_KEY);

}

@Bean

public Binding ownersFullAccessBinding() {

return BindingBuilder.bind(ownersFullAccessQueue()).to(ownersExchange()).with(OWNERS\_FULL\_ACCESS\_ROUTING\_KEY);

}

@Bean

public Binding ownersCatAccessBinding() {

return BindingBuilder.bind(ownersCatAccessQueue()).to(ownersExchange()).with(OWNERS\_CAT\_ACCESS\_ROUTING\_KEY);

}

@Bean

public Binding ownersSaveBinding() {

return BindingBuilder.bind(ownersSaveQueue()).to(ownersExchange()).with(OWNERS\_SAVE\_ROUTING\_KEY);

}

@Bean

public Binding ownersDeleteBinding() {

return BindingBuilder.bind(ownersDeleteQueue()).to(ownersExchange()).with(OWNERS\_DELETE\_ROUTING\_KEY);

}

@Bean

public Binding ownersExistBinding() {

return BindingBuilder.bind(ownersExistQueue()).to(ownersExchange()).with(OWNERS\_EXIST\_ROUTING\_KEY);

}

@Bean

public Binding usersGetBinding() {

return BindingBuilder.bind(usersGetQueue()).to(ownersExchange()).with(USERS\_GET\_ROUTING\_KEY);

}

@Bean

public Binding usersSaveBinding() {

return BindingBuilder.bind(usersSaveQueue()).to(ownersExchange()).with(USERS\_SAVE\_ROUTING\_KEY);

}

@Bean

public Binding usersDeleteBinding() {

return BindingBuilder.bind(usersDeleteQueue()).to(ownersExchange()).with(USERS\_DELETE\_ROUTING\_KEY);

}

}

Изменен файл kotiki-java/api/src/main/java/org/itmo/configs/SecurityConfig.java -> kotiki-java/external-interfaces/src/main/java/org/itmo/configs/SecurityConfig.java:

import lombok.RequiredArgsConstructor;

import org.itmo.dto.UserDto;

import org.itmo.services.auth.UserService;

-import org.itmo.utils.UserConverter;

import org.springframework.http.HttpStatus;

import org.springframework.security.access.prepost.PreAuthorize;

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

@@ -15,18 +14,17 @@

public class AuthController extends BaseController {

private final UserService userService;

- private final UserConverter userConverter;

@PostMapping("/create")

@PreAuthorize("hasRole('ADMIN')")

public void create(@RequestBody UserDto user) {

- userService.save(userConverter.toModel(user));

+ userService.save(user);

}

@PostMapping("/update")

@PreAuthorize("hasRole('ADMIN')")

public void update(@RequestBody UserDto user) {

- userService.save(userConverter.toModel(user));

+ userService.save(user);

}

@PostMapping("/delete/{id}")

Изменен файл kotiki-java/api/src/main/java/org/itmo/controllers/BaseController.java -> kotiki-java/external-interfaces/src/main/java/org/itmo/controllers/BaseController.java:

import lombok.RequiredArgsConstructor;

import org.itmo.dto.CatDto;

import org.itmo.enums.Color;

-import org.itmo.models.Cat;

+import org.itmo.services.AbstractOwnerService;

import org.itmo.services.auth.UserService;

import org.springframework.http.HttpStatus;

import org.springframework.security.access.prepost.PreAuthorize;

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

import org.springframework.web.server.ResponseStatusException;

import org.itmo.services.AbstractCatService;

-import org.itmo.utils.CatConverter;

import java.util.List;

-import java.util.stream.Collectors;

@RestController

@RequestMapping("/api/cat")

@@ -21,83 +19,68 @@

public class CatController extends BaseController {

private final AbstractCatService catService;

- private final CatConverter converter;

+ private final AbstractOwnerService ownerService;

private final UserService userService;

@GetMapping("/get/all")

public List<CatDto> getAll() {

- List<Cat> cats = userService

+ return userService

.isCurrentUserNotAnAdmin()

? catService.getByOwnerId(userService.getCurrentOwnerId())

: catService.getAll();

-

- return cats

- .stream()

- .map(converter::toDto)

- .collect(Collectors.toList());

}

@GetMapping("/get/{id}")

public CatDto get(@PathVariable int id) {

- Cat cat = catService.get(id);

+ CatDto cat = catService.get(id);

if (userService.isCurrentUserNotAnAdmin()

- && cat.getOwner().getId() != userService.getCurrentOwnerId())

+ && ownerService.getByCatId(cat.getId()).getId() != userService.getCurrentOwnerId())

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

- return converter.toDto(cat);

+ return cat;

}

@GetMapping("/get/breed/{breed}")

public List<CatDto> getByBreed(@PathVariable String breed) {

- List<Cat> cats = userService

+ return userService

.isCurrentUserNotAnAdmin()

? catService.getByBreedAndOwnerId(breed, userService.getCurrentOwnerId())

: catService.getByBreed(breed);

-

- return cats

- .stream()

- .map(converter::toDto)

- .collect(Collectors.toList());

}

@GetMapping("/get/color/{color}")

public List<CatDto> getByColor(@PathVariable String color) {

var enumColor = Color.valueOf(color.toUpperCase());

- List<Cat> cats = userService

+ return userService

.isCurrentUserNotAnAdmin()

? catService.getByColorAndOwnerId(enumColor, userService.getCurrentOwnerId())

: catService.getByColor(enumColor);

-

- return cats

- .stream()

- .map(converter::toDto)

- .collect(Collectors.toList());

}

@PostMapping("/create")

@PreAuthorize("hasRole('ADMIN')")

public void create(@RequestBody CatDto cat) {

- catService.save(converter.toModel(cat));

+ catService.save(cat);

}

@PostMapping("/update")

public void update(@RequestBody CatDto cat) {

if (userService.isCurrentUserNotAnAdmin()

- && cat.getOwnerId() != userService.getCurrentOwnerId())

+ && ownerService.getByCatId(cat.getId()).getId() != userService.getCurrentOwnerId())

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

if (!catService.exists(cat.getId()))

throw new ResponseStatusException(HttpStatus.NOT\_FOUND);

- catService.save(converter.toModel(cat));

+ catService.save(cat);

}

@PostMapping("/delete/{id}")

public void delete(@PathVariable int id) {

- Cat cat = catService.get(id);

+ CatDto cat = catService.get(id);

if (userService.isCurrentUserNotAnAdmin()

- && cat.getOwner().getId() != userService.getCurrentOwnerId())

+ && ownerService.getByCatId(cat.getId()).getId() != userService.getCurrentOwnerId())

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

catService.delete(id);

Изменен файл kotiki-java/api/src/main/java/org/itmo/controllers/OwnerController.java -> kotiki-java/external-interfaces/src/main/java/org/itmo/controllers/OwnerController.java:

import lombok.RequiredArgsConstructor;

import org.itmo.dto.OwnerDto;

+import org.itmo.services.AbstractOwnerService;

import org.itmo.services.auth.UserService;

import org.springframework.http.HttpStatus;

import org.springframework.security.access.prepost.PreAuthorize;

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

import org.springframework.web.server.ResponseStatusException;

-import org.itmo.services.AbstractOwnerService;

-import org.itmo.utils.OwnerConverter;

import java.util.List;

-import java.util.stream.Collectors;

@RestController

@RequestMapping("/api/owner")

@@ -19,17 +17,12 @@

public class OwnerController extends BaseController {

private final AbstractOwnerService ownerService;

- private final OwnerConverter converter;

private final UserService userService;

@GetMapping("/get/all")

@PreAuthorize("hasRole('ADMIN')")

public List<OwnerDto> getAll() {

- return ownerService

- .getAll()

- .stream()

- .map(converter::toDto)

- .collect(Collectors.toList());

+ return ownerService.getAll();

}

@GetMapping("/get/{id}")

public OwnerDto get(@PathVariable int id) {

&& userService.getCurrentOwnerId() != id)

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

- return converter.toDto(ownerService.get(id));

+ return ownerService.get(id);

}

@PostMapping("/create")

@PreAuthorize("hasRole('ADMIN')")

public void create(@RequestBody OwnerDto owner) {

- ownerService.save(converter.toModel(owner));

+ ownerService.save(owner);

}

@PostMapping("/update")

public void update(@RequestBody OwnerDto owner) {

if (!ownerService.exists(owner.getId()))

throw new ResponseStatusException(HttpStatus.NOT\_FOUND);

- ownerService.save(converter.toModel(owner));

+ ownerService.save(owner);

}

@PostMapping("/delete/{id}")

Добавлен файл kotiki-java/external-interfaces/src/main/java/org/itmo/services/AbstractCatService.java:

package org.itmo.services;

import org.itmo.dto.CatDto;

import org.itmo.enums.Color;

import java.util.List;

public interface AbstractCatService {

List<CatDto> getAll();

CatDto get(int id);

List<CatDto> getByBreed(String breed);

List<CatDto> getByColor(Color color);

List<CatDto> getByOwnerId(int ownerId);

List<CatDto> getByBreedAndOwnerId(String breed, int ownerId);

List<CatDto> getByColorAndOwnerId(Color color, int ownerId);

void save(CatDto cat);

void delete(int id);

boolean exists(int id);

}

Добавлен файл kotiki-java/external-interfaces/src/main/java/org/itmo/services/AbstractOwnerService.java:

package org.itmo.services;

import org.itmo.dto.OwnerDto;

import java.util.List;

public interface AbstractOwnerService {

List<OwnerDto> getAll();

OwnerDto get(int id);

OwnerDto getByCatId(int catId);

void save(OwnerDto owner);

void delete(int id);

boolean exists(int id);

}

Добавлен файл kotiki-java/external-interfaces/src/main/java/org/itmo/services/CatService.java:

package org.itmo.services;

import lombok.RequiredArgsConstructor;

import org.itmo.dto.CatDto;

import org.itmo.dto.rabbit.BreedDto;

import org.itmo.dto.rabbit.ColorDto;

import org.itmo.dto.rabbit.OwnerIdDto;

import org.itmo.enums.Color;

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

import org.springframework.core.ParameterizedTypeReference;

import org.springframework.stereotype.Service;

import java.util.List;

import static org.itmo.configs.CatsRabbitConfig.\*;

@Service

@RequiredArgsConstructor

@SuppressWarnings("unchecked")

public class CatService implements AbstractCatService {

private final RabbitTemplate template;

public List<CatDto> getAll() {

return (List<CatDto>)template.convertSendAndReceive(

CATS\_EXCHANGE,

CATS\_FULL\_ACCESS\_ROUTING\_KEY,

new OwnerIdDto());

}

public CatDto get(int id) {

return template.convertSendAndReceiveAsType(

CATS\_EXCHANGE,

CATS\_ACCESS\_ROUTING\_KEY,

id,

ParameterizedTypeReference.forType(CatDto.class));

}

public List<CatDto> getByBreed(String breed) {

return (List<CatDto>)template.convertSendAndReceive(

CATS\_EXCHANGE,

CATS\_ACCESS\_ROUTING\_KEY,

new BreedDto(breed, null));

}

public List<CatDto> getByColor(Color color) {

return (List<CatDto>)template.convertSendAndReceive(

CATS\_EXCHANGE,

CATS\_ACCESS\_ROUTING\_KEY,

new ColorDto(color, null));

}

public List<CatDto> getByOwnerId(int ownerId) {

return (List<CatDto>)template.convertSendAndReceive(

CATS\_EXCHANGE,

CATS\_ACCESS\_ROUTING\_KEY,

new OwnerIdDto(ownerId));

}

public List<CatDto> getByBreedAndOwnerId(String breed, int ownerId) {

return (List<CatDto>)template.convertSendAndReceive(

CATS\_EXCHANGE,

CATS\_ACCESS\_ROUTING\_KEY,

new BreedDto(breed, ownerId));

}

public List<CatDto> getByColorAndOwnerId(Color color, int ownerId) {

return (List<CatDto>)template.convertSendAndReceive(

CATS\_EXCHANGE,

CATS\_ACCESS\_ROUTING\_KEY,

new ColorDto(color, null));

}

public void save(CatDto cat) {

template.convertAndSend(

CATS\_EXCHANGE,

CATS\_SAVE\_ROUTING\_KEY,

cat);

}

public void delete(int id) {

template.convertAndSend(

CATS\_EXCHANGE,

CATS\_DELETE\_ROUTING\_KEY,

id);

}

public boolean exists(int id) {

return Boolean.TRUE.equals(template.convertSendAndReceiveAsType(

CATS\_EXCHANGE,

CATS\_EXISTS\_ROUTING\_KEY,

id,

ParameterizedTypeReference.forType(boolean.class)));

}

}

Добавлен файл kotiki-java/external-interfaces/src/main/java/org/itmo/services/OwnerService.java:

package org.itmo.services;

import lombok.RequiredArgsConstructor;

import org.itmo.dto.OwnerDto;

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

import org.springframework.core.ParameterizedTypeReference;

import org.springframework.stereotype.Service;

import java.util.List;

import static org.itmo.configs.OwnersRabbitConfig.\*;

@Service

@RequiredArgsConstructor

@SuppressWarnings("unchecked")

public class OwnerService implements AbstractOwnerService {

private final RabbitTemplate template;

public List<OwnerDto> getAll() {

return (List<OwnerDto>)template.convertSendAndReceive(

OWNERS\_EXCHANGE,

OWNERS\_FULL\_ACCESS\_ROUTING\_KEY,

"message");

}

public OwnerDto get(int id) {

return template.convertSendAndReceiveAsType(

OWNERS\_EXCHANGE,

OWNERS\_ACCESS\_ROUTING\_KEY,

id,

ParameterizedTypeReference.forType(OwnerDto.class));

}

public OwnerDto getByCatId(int catId) {

return template.convertSendAndReceiveAsType(

OWNERS\_EXCHANGE,

OWNERS\_CAT\_ACCESS\_ROUTING\_KEY,

catId,

ParameterizedTypeReference.forType(OwnerDto.class));

}

public void save(OwnerDto owner) {

template.convertAndSend(

OWNERS\_EXCHANGE,

OWNERS\_SAVE\_ROUTING\_KEY,

owner);

}

public void delete(int id) {

template.convertAndSend(

OWNERS\_EXCHANGE,

OWNERS\_DELETE\_ROUTING\_KEY,

id);

}

public boolean exists(int id) {

return Boolean.TRUE.equals(template.convertSendAndReceiveAsType(

OWNERS\_EXCHANGE,

OWNERS\_DELETE\_ROUTING\_KEY,

id,

ParameterizedTypeReference.forType(boolean.class)));

}

}

Изменен файл kotiki-java/api/src/main/java/org/itmo/services/auth/UserService.java -> kotiki-java/external-interfaces/src/main/java/org/itmo/services/auth/UserService.java:

package org.itmo.services.auth;

import lombok.RequiredArgsConstructor;

-import org.itmo.dao.auth.UserDao;

-import org.itmo.models.auth.Role;

-import org.itmo.models.auth.User;

+import org.itmo.dto.UserDto;

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

+import org.springframework.core.ParameterizedTypeReference;

import org.springframework.security.core.GrantedAuthority;

import org.springframework.security.core.authority.SimpleGrantedAuthority;

import org.springframework.security.core.context.SecurityContextHolder;

@@ -16,15 +16,17 @@

import java.util.Collection;

import java.util.stream.Collectors;

+import static org.itmo.configs.OwnersRabbitConfig.\*;

+

@Service

@RequiredArgsConstructor

public class UserService implements UserDetailsService {

- private final UserDao dao;

+ private final RabbitTemplate template;

@Transactional

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

- User user = dao.getByUsername(username);

+ UserDto user = get(username);

if (user == null)

throw new UsernameNotFoundException("User " + username + " was not found");

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundEx

mapRolesToAuthorities(user.getRoles()));

}

- public void save(User user) {

- dao.save(user);

+ public void save(UserDto user) {

+ template.convertAndSend(

+ OWNERS\_EXCHANGE,

+ USERS\_SAVE\_ROUTING\_KEY,

+ user);

}

public void delete(int id) {

- dao.deleteById(id);

+ template.convertAndSend(

+ OWNERS\_EXCHANGE,

+ USERS\_DELETE\_ROUTING\_KEY,

+ id);

}

public boolean isCurrentUserNotAnAdmin() {

public boolean isCurrentUserNotAnAdmin() {

}

public int getCurrentOwnerId() {

- return dao

- .getByUsername(getCurrentUsername())

- .getOwner()

- .getId();

+ return get(getCurrentUsername())

+ .getOwnerId();

}

public int getCurrentId() {

- return dao.getByUsername(getCurrentUsername()).getId();

+ return get(getCurrentUsername()).getId();

}

public String getCurrentUsername() {

public String getCurrentUsername() {

: principal.toString();

}

- private Collection<? extends GrantedAuthority> mapRolesToAuthorities(Collection<Role> roles) {

+ private UserDto get(String username) {

+ return template.convertSendAndReceiveAsType(

+ OWNERS\_EXCHANGE,

+ USERS\_GET\_ROUTING\_KEY,

+ username,

+ ParameterizedTypeReference.forType(UserDto.class));

+ }

+

+ private Collection<? extends GrantedAuthority> mapRolesToAuthorities(Collection<String> roles) {

return roles

.stream()

- .map(r -> new SimpleGrantedAuthority(r.getName()))

+ .map(SimpleGrantedAuthority::new)

.collect(Collectors.toList());

}

}

Добавлен файл kotiki-java/external-interfaces/src/main/resources/application.properties:

spring.datasource.url=jdbc:postgresql://localhost:5432/postgres

spring.datasource.username=postgres

spring.datasource.password=koshkin

spring.jpa.show-sql=true

## Hibernate Properties

# The SQL dialect makes Hibernate generate better SQL for the chosen database

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.PostgreSQLDialect

spring.jpa.properties.hibernate.event.merge.entity\_copy\_observer=allow

# Hibernate ddl auto (create, create-drop, validate, update)

spring.jpa.hibernate.ddl-auto = update

server.port=9090

spring.rabbitmq.host=localhost

spring.rabbitmq.username=guest

spring.rabbitmq.password=guest

spring.rabbitmq.port=5672

Изменен файл kotiki-java/dal/src/test/java/org/itmo/models/CatTest.java -> kotiki-java/external-interfaces/src/test/java/entities/CatTest.java:

-package org.itmo.models;

+package entities;

+import org.itmo.entities.Cat;

import org.itmo.enums.Color;

import org.itmo.tools.DalException;

import org.junit.jupiter.api.Test;

Изменен файл kotiki-java/dal/src/test/java/org/itmo/models/OwnerTest.java -> kotiki-java/external-interfaces/src/test/java/entities/OwnerTest.java:

-package org.itmo.models;

+package entities;

+import org.itmo.entities.Owner;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

Добавлен файл kotiki-java/owners-access/pom.xml:

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.6.5</version>

<relativePath/>

</parent>

<modelVersion>4.0.0</modelVersion>

<artifactId>owners-access</artifactId>

<properties>

<maven.compiler.source>11</maven.compiler.source>

<maven.compiler.target>11</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>org.itmo</groupId>

<artifactId>domain-jpa</artifactId>

<version>1.0</version>

<scope>compile</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>5.8.2</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

<version>2.6.7</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

<version>2.6.7</version>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<version>2.6.7</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<version>2.6.7</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-amqp</artifactId>

<version>2.6.7</version>

</dependency>

<dependency>

<groupId>org.apache.tomcat.embed</groupId>

<artifactId>tomcat-embed-jasper</artifactId>

<version>10.1.0-M12</version>

</dependency>

<dependency>

<groupId>org.glassfish</groupId>

<artifactId>javax.faces</artifactId>

<version>2.4.0</version>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.24</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-crypto</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.0</version>

</plugin>

<plugin>

<artifactId>maven-failsafe-plugin</artifactId>

<version>2.22.0</version>

</plugin>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<version>2.6.6</version>

</plugin>

</plugins>

</build>

</project>

Добавлен файл kotiki-java/owners-access/src/main/java/org/itmo/Application.java:

package org.itmo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

}

}

Добавлен файл kotiki-java/owners-access/src/main/java/org/itmo/configs/OwnersRabbitConfig.java:

package org.itmo.configs;

import org.springframework.amqp.core.Queue;

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

import org.springframework.amqp.support.converter.Jackson2JsonMessageConverter;

import org.springframework.amqp.support.converter.MessageConverter;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@EnableRabbit

@Configuration

public class OwnersRabbitConfig {

public static final String OWNERS\_FULL\_ACCESS\_QUEUE = "owners-full-access-queue";

public static final String OWNERS\_ACCESS\_QUEUE = "owners-access-queue";

public static final String OWNERS\_CAT\_ACCESS\_QUEUE = "owners-cat-access-queue";

public static final String OWNERS\_SAVE\_QUEUE = "owners-save-queue";

public static final String OWNERS\_DELETE\_QUEUE = "owners-delete-queue";

public static final String OWNERS\_EXIST\_QUEUE = "owners-exist-queue";

public static final String USERS\_GET\_QUEUE = "users-get-queue";

public static final String USERS\_SAVE\_QUEUE = "users-save-queue";

public static final String USERS\_DELETE\_QUEUE = "users-delete-queue";

@Bean

public Queue ownersAccessQueue() {

return new Queue(OWNERS\_ACCESS\_QUEUE);

}

@Bean

public Queue ownersFullAccessQueue() {

return new Queue(OWNERS\_FULL\_ACCESS\_QUEUE);

}

@Bean Queue ownersCatAccessQueue() {

return new Queue(OWNERS\_CAT\_ACCESS\_QUEUE);

}

@Bean

public Queue ownersSaveQueue() {

return new Queue(OWNERS\_SAVE\_QUEUE);

}

@Bean

public Queue ownersDeleteQueue() {

return new Queue(OWNERS\_DELETE\_QUEUE);

}

@Bean

public Queue ownersExistQueue() {

return new Queue(OWNERS\_EXIST\_QUEUE);

}

@Bean

public Queue usersGetQueue() {

return new Queue(USERS\_GET\_QUEUE);

}

@Bean

public Queue usersSaveQueue() {

return new Queue(USERS\_SAVE\_QUEUE);

}

@Bean

public Queue usersDeleteQueue() {

return new Queue(USERS\_DELETE\_QUEUE);

}

@Bean

public MessageConverter messageConverter() {

return new Jackson2JsonMessageConverter();

}

}

Добавлен файл kotiki-java/owners-access/src/main/java/org/itmo/configs/PasswordEncoderProvider.java:

package org.itmo.configs;

import org.springframework.context.annotation.Bean;

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

import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.stereotype.Component;

@Component

public class PasswordEncoderProvider {

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

}

Изменен файл kotiki-java/dal/src/main/java/org/itmo/dao/OwnerDao.java -> kotiki-java/owners-access/src/main/java/org/itmo/dao/OwnerDao.java:

package org.itmo.dao;

-import org.itmo.models.Owner;

+import org.itmo.entities.Owner;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

Изменен файл kotiki-java/dal/src/main/java/org/itmo/dao/auth/RoleDao.java -> kotiki-java/owners-access/src/main/java/org/itmo/dao/auth/RoleDao.java:

package org.itmo.dao.auth;

-import org.itmo.models.auth.Role;

+import org.itmo.entities.auth.Role;

import org.springframework.data.jpa.repository.JpaRepository;

public interface RoleDao extends JpaRepository<Role, Integer> {

Изменен файл kotiki-java/dal/src/main/java/org/itmo/dao/auth/UserDao.java -> kotiki-java/owners-access/src/main/java/org/itmo/dao/auth/UserDao.java:

package org.itmo.dao.auth;

-import org.itmo.models.auth.User;

+import org.itmo.entities.auth.User;

import org.springframework.data.jpa.repository.JpaRepository;

public interface UserDao extends JpaRepository<User, Integer> {

Добавлен файл kotiki-java/owners-access/src/main/java/org/itmo/listeners/OwnerListener.java:

package org.itmo.listeners;

import lombok.RequiredArgsConstructor;

import org.itmo.dto.OwnerDto;

import org.itmo.services.AbstractOwnerService;

import org.itmo.utils.OwnerConverter;

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

import org.springframework.stereotype.Component;

import java.util.List;

import java.util.stream.Collectors;

import static org.itmo.configs.OwnersRabbitConfig.\*;

@Component

@RequiredArgsConstructor

public class OwnerListener {

private final AbstractOwnerService service;

private final OwnerConverter converter;

@RabbitListener(queues = OWNERS\_FULL\_ACCESS\_QUEUE)

public List<OwnerDto> getAll(String ignoredMessage) {

return service.getAll()

.stream()

.map(converter::toDto)

.collect(Collectors.toList());

}

@RabbitListener(queues = OWNERS\_CAT\_ACCESS\_QUEUE)

public OwnerDto getByCatId(int catId) {

return converter.toDto(service.getByCatId(catId));

}

@RabbitListener(queues = OWNERS\_ACCESS\_QUEUE)

public OwnerDto get(int id) {

return converter.toDto(service.get(id));

}

@RabbitListener(queues = OWNERS\_SAVE\_QUEUE)

public void save(OwnerDto owner) {

service.save(converter.toModel(owner));

}

@RabbitListener(queues = OWNERS\_DELETE\_QUEUE)

public void delete(int id) {

service.delete(id);

}

}

Добавлен файл kotiki-java/owners-access/src/main/java/org/itmo/listeners/UserListener.java:

package org.itmo.listeners;

import lombok.RequiredArgsConstructor;

import org.itmo.dto.UserDto;

import org.itmo.services.auth.UserService;

import org.itmo.utils.UserConverter;

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

import org.springframework.stereotype.Component;

import static org.itmo.configs.OwnersRabbitConfig.\*;

@Component

@RequiredArgsConstructor

public class UserListener {

private final UserService service;

private final UserConverter converter;

@RabbitListener(queues = USERS\_GET\_QUEUE)

public UserDto get(String username) {

return converter.toDto(service.get(username));

}

@RabbitListener(queues = USERS\_SAVE\_QUEUE)

public void save(UserDto user) {

service.save(converter.toModel(user));

}

@RabbitListener(queues = USERS\_DELETE\_QUEUE)

public void delete(int id) {

service.delete(id);

}

}

Изменен файл kotiki-java/api/src/main/java/org/itmo/services/AbstractOwnerService.java -> kotiki-java/owners-access/src/main/java/org/itmo/services/AbstractOwnerService.java:

package org.itmo.services;

-import org.itmo.models.Owner;

+import org.itmo.entities.Owner;

import java.util.List;

public interface AbstractOwnerService {

Owner get(int id);

+ Owner getByCatId(int catId);

+

void save(Owner owner);

void delete(int id);

Изменен файл kotiki-java/api/src/main/java/org/itmo/services/OwnerService.java -> kotiki-java/owners-access/src/main/java/org/itmo/services/OwnerService.java:

package org.itmo.services;

-import lombok.RequiredArgsConstructor;

import org.itmo.dao.OwnerDao;

-import org.itmo.models.Owner;

+import lombok.RequiredArgsConstructor;

+import org.itmo.entities.Owner;

import org.itmo.tools.DalException;

import org.springframework.stereotype.Service;

public Owner get(int id) {

return result.get();

}

+ public Owner getByCatId(int catId) {

+ return getAll()

+ .stream()

+ .filter(o -> o.getCats().stream().anyMatch(c -> c.getId() == catId))

+ .findFirst()

+ .orElse(null);

+ }

+

public void save(Owner owner) {

dao.save(owner);

}

Изменен файл kotiki-java/api/src/main/java/org/itmo/services/auth/RoleService.java -> kotiki-java/owners-access/src/main/java/org/itmo/services/auth/RoleService.java:

package org.itmo.services.auth;

-import lombok.RequiredArgsConstructor;

import org.itmo.dao.auth.RoleDao;

-import org.itmo.models.auth.Role;

+import lombok.RequiredArgsConstructor;

+import org.itmo.entities.auth.Role;

import org.springframework.stereotype.Service;

@Service

Добавлен файл kotiki-java/owners-access/src/main/java/org/itmo/services/auth/UserService.java:

package org.itmo.services.auth;

import org.itmo.dao.auth.UserDao;

import lombok.RequiredArgsConstructor;

import org.itmo.entities.auth.User;

import org.springframework.stereotype.Service;

@Service

@RequiredArgsConstructor

public class UserService {

private final UserDao dao;

public User get(String username) {

return dao.getByUsername(username);

}

public void save(User user) {

dao.save(user);

}

public void delete(int id) {

dao.deleteById(id);

}

}

Изменен файл kotiki-java/api/src/main/java/org/itmo/utils/OwnerConverter.java -> kotiki-java/owners-access/src/main/java/org/itmo/utils/OwnerConverter.java:

import lombok.RequiredArgsConstructor;

import org.itmo.dto.OwnerDto;

-import org.itmo.models.Cat;

-import org.itmo.models.Owner;

+import org.itmo.entities.Owner;

import org.springframework.stereotype.Component;

-import org.itmo.services.AbstractCatService;

-

-import java.util.stream.Collectors;

@Component

@RequiredArgsConstructor

public class OwnerConverter {

- private final AbstractCatService catService;

-

public OwnerDto toDto(Owner owner) {

return OwnerDto.builder()

.id(owner.getId())

.name(owner.getName())

- .catIds(owner.getCats()

- .stream()

- .map(Cat::getId)

- .collect(Collectors.toList()))

.build();

}

public Owner toModel(OwnerDto owner) {

.id(owner.getId())

.name(owner.getName());

- if (owner.getCatIds() != null)

- ownerBuilder.cats(owner.getCatIds()

- .stream()

- .map(catService::get)

- .collect(Collectors.toList()));

-

return ownerBuilder.build();

}

}

Изменен файл kotiki-java/api/src/main/java/org/itmo/utils/UserConverter.java -> kotiki-java/owners-access/src/main/java/org/itmo/utils/UserConverter.java:

package org.itmo.utils;

-import lombok.RequiredArgsConstructor;

-import org.itmo.dto.UserDto;

-import org.itmo.models.auth.User;

+import org.itmo.entities.auth.Role;

import org.itmo.services.AbstractOwnerService;

import org.itmo.services.auth.RoleService;

+import lombok.RequiredArgsConstructor;

+import org.itmo.dto.UserDto;

+import org.itmo.entities.auth.User;

import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.stereotype.Component;

public class UserConverter {

private final RoleService roleService;

private final PasswordEncoder passwordEncoder;

+ public UserDto toDto(User user) {

+ return UserDto.builder()

+ .id(user.getId())

+ .username(user.getUsername())

+ .password(user.getPassword())

+ .ownerId(user.getOwner() != null

+ ? user.getOwner().getId()

+ : null)

+ .roles(user.getRoles()

+ .stream()

+ .map(Role::getName)

+ .collect(Collectors.toList()))

+ .build();

+ }

+

public User toModel(UserDto user) {

User.UserBuilder userBuilder = User.builder()

.id(user.getId())

Добавлен файл kotiki-java/owners-access/src/main/resources/application.properties:

spring.datasource.url=jdbc:postgresql://localhost:5432/postgres

spring.datasource.username=postgres

spring.datasource.password=koshkin

spring.jpa.show-sql=true

## Hibernate Properties

# The SQL dialect makes Hibernate generate better SQL for the chosen database

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.PostgreSQLDialect

spring.jpa.properties.hibernate.event.merge.entity\_copy\_observer=allow

# Hibernate ddl auto (create, create-drop, validate, update)

spring.jpa.hibernate.ddl-auto = update

server.port=9092

spring.rabbitmq.host=localhost

spring.rabbitmq.username=guest

spring.rabbitmq.password=guest

spring.rabbitmq.port=5672

Изменен файл kotiki-java/pom.xml:

+<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

- <parent>

- <groupId>org.springframework.boot</groupId>

- <artifactId>spring-boot-starter-parent</artifactId>

- <version>2.6.5</version>

- <relativePath/>

- </parent>

<modelVersion>4.0.0</modelVersion>

<groupId>org.itmo</groupId>

@@ -14,93 +9,16 @@

<packaging>pom</packaging>

<version>1.0</version>

<modules>

- <module>dal</module>

- <module>api</module>

+ <module>domain</module>

+ <module>domain-jpa</module>

+ <module>cats-access</module>

+ <module>owners-access</module>

+ <module>external-interfaces</module>

</modules>

<properties>

<maven.compiler.source>11</maven.compiler.source>

<maven.compiler.target>11</maven.compiler.target>

- <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

- <dependencies>

-

- <dependency>

- <groupId>org.junit.jupiter</groupId>

- <artifactId>junit-jupiter-engine</artifactId>

- <version>5.8.2</version>

- </dependency>

-

- <dependency>

- <groupId>org.springframework.boot</groupId>

- <artifactId>spring-boot-starter-data-jpa</artifactId>

- <version>2.6.6</version>

- </dependency>

-

- <dependency>

- <groupId>org.springframework.boot</groupId>

- <artifactId>spring-boot-starter-web</artifactId>

- <version>2.6.6</version>

- </dependency>

-

- <dependency>

- <groupId>org.springframework.boot</groupId>

- <artifactId>spring-boot-devtools</artifactId>

- <version>2.6.6</version>

- <scope>runtime</scope>

- </dependency>

-

- <dependency>

- <groupId>org.springframework.boot</groupId>

- <artifactId>spring-boot-starter-test</artifactId>

- <version>2.6.6</version>

- <scope>test</scope>

- </dependency>

-

- <dependency>

- <groupId>org.springframework.boot</groupId>

- <artifactId>spring-boot-starter-security</artifactId>

- <version>2.6.6</version>

- </dependency>

-

- <dependency>

- <groupId>org.apache.tomcat.embed</groupId>

- <artifactId>tomcat-embed-jasper</artifactId>

- <version>10.1.0-M12</version>

- </dependency>

-

- <dependency>

- <groupId>org.glassfish</groupId>

- <artifactId>javax.faces</artifactId>

- <version>2.4.0</version>

- </dependency>

-

- <dependency>

- <groupId>org.projectlombok</groupId>

- <artifactId>lombok</artifactId>

- <version>1.18.22</version>

- <scope>provided</scope>

- </dependency>

-

- </dependencies>

-

- <build>

- <plugins>

- <plugin>

- <artifactId>maven-surefire-plugin</artifactId>

- <version>2.22.0</version>

- </plugin>

- <plugin>

- <artifactId>maven-failsafe-plugin</artifactId>

- <version>2.22.0</version>

- </plugin>

- <plugin>

- <groupId>org.springframework.boot</groupId>

- <artifactId>spring-boot-maven-plugin</artifactId>

- <version>2.6.6</version>

- </plugin>

- </plugins>

- </build>

-

</project>