**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №4

по дисциплине «**Технологии программирования**»

Автор: Кошкин Михаил Сергеевич

Факультет: ИТиП

Группа: М32051



Санкт-Петербург 2022

Описание:

Владельцы недовольны, что информацию о котиках может получить кто угодно. В этой лабораторной мы добавим авторизацию к сервису.

Добавляется роль администратора. Он имеет доступ ко всем методам и может создавать новых пользователей. Пользователь связан с владельцем в соотношении 1:1.

Методы по получению информации и котиках и владельцах должны быть защищены Spring Security. Доступ к соответствующим endpoint’ам имеют только владельцы котиков и администраторы. Доступ к методам для фильтрации имеют все авторизованные пользователи, но на выходе получают только данные о своих котиках.

Внимание: эндпоинты, созданные на предыдущем этапе, не должны быть удалены.

Исходный код:

Удален файл kotiki-java/.github/workflows/maven.yml

Изменен файл kotiki-java/api/pom.xml:

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>kotiki-java</groupId>

- <version>1.0-SNAPSHOT</version>

+ <groupId>org.itmo</groupId>

+ <version>1.0</version>

</parent>

<modelVersion>4.0.0</modelVersion>

@@ -18,9 +18,9 @@

<dependencies>

<dependency>

- <groupId>kotiki-java</groupId>

+ <groupId>org.itmo</groupId>

<artifactId>dal</artifactId>

- <version>1.0-SNAPSHOT</version>

+ <version>1.0</version>

</dependency>

<dependency>

Удален файл kotiki-java/api/src/main/java/app/Application.java

Удален файл kotiki-java/api/src/main/java/app/BrowserLauncher.java

Удален файл kotiki-java/api/src/main/java/controllers/CatController.java

Удален файл kotiki-java/api/src/main/java/controllers/OwnerController.java

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

package org.itmo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import java.awt.\*;

import java.net.URI;

@SpringBootApplication

public class Application {

public static void main(String[] args) {

SpringApplication.run(Application.class, args);

launchBrowser();

}

private static void launchBrowser() {

try {

System.setProperty("java.awt.headless", "false");

Desktop.getDesktop().browse(new URI("http://localhost:9090/api/cat/get/all"));

} catch (Exception ignored) { }

}

}

Добавлен файл kotiki-java/api/src/main/java/org/itmo/configs/SecurityConfig.java:

package org.itmo.configs;

import lombok.RequiredArgsConstructor;

import org.itmo.services.auth.UserService;

import org.springframework.context.annotation.Bean;

import org.springframework.security.authentication.dao.DaoAuthenticationProvider;

import org.springframework.security.config.annotation.method.configuration.EnableGlobalMethodSecurity;

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;

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

@EnableWebSecurity

@EnableGlobalMethodSecurity(prePostEnabled = true)

@RequiredArgsConstructor

public class SecurityConfig extends WebSecurityConfigurerAdapter {

private final UserService userService;

@Override

protected void configure(HttpSecurity http) throws Exception {

http.httpBasic()

.and()

.authorizeRequests()

.antMatchers("/api/\*\*").authenticated()

.and()

.csrf().disable()

.formLogin();

}

@Bean

public PasswordEncoder passwordEncoder() {

return new BCryptPasswordEncoder();

}

@Bean

public DaoAuthenticationProvider authenticationProvider() {

var authenticationProvider = new DaoAuthenticationProvider();

authenticationProvider.setPasswordEncoder(passwordEncoder());

authenticationProvider.setUserDetailsService(userService);

return authenticationProvider;

}

}

Добавлен файл kotiki-java/api/src/main/java/org/itmo/controllers/AuthController.java:

package org.itmo.controllers;

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.\*;

import org.springframework.web.server.ResponseStatusException;

@RestController

@RequestMapping("/auth")

@RequiredArgsConstructor

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));

}

@PostMapping("/update")

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

public void update(@RequestBody UserDto user) {

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

}

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

public void delete(@PathVariable int id) {

if (userService.isCurrentUserNotAnAdmin()

&& userService.getCurrentId() != id)

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

userService.delete(id);

}

}

Добавлен файл kotiki-java/api/src/main/java/org/itmo/controllers/BaseController.java:

package org.itmo.controllers;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.server.ResponseStatusException;

public abstract class BaseController {

@ExceptionHandler(Exception.class)

public void handleAllExceptions(Exception e) {

throw new ResponseStatusException(HttpStatus.INTERNAL\_SERVER\_ERROR, "Internal server error", e);

}

}

Добавлен файл kotiki-java/api/src/main/java/org/itmo/controllers/CatController.java:

package org.itmo.controllers;

import lombok.RequiredArgsConstructor;

import org.itmo.dto.CatDto;

import org.itmo.enums.Color;

import org.itmo.models.Cat;

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")

@RequiredArgsConstructor

public class CatController extends BaseController {

private final AbstractCatService catService;

private final CatConverter converter;

private final UserService userService;

@GetMapping("/get/all")

public List<CatDto> getAll() {

List<Cat> cats = 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);

if (userService.isCurrentUserNotAnAdmin()

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

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

return converter.toDto(cat);

}

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

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

List<Cat> cats = 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

.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));

}

@PostMapping("/update")

public void update(@RequestBody CatDto cat) {

if (userService.isCurrentUserNotAnAdmin()

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

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

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

throw new ResponseStatusException(HttpStatus.NOT\_FOUND);

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

}

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

public void delete(@PathVariable int id) {

Cat cat = catService.get(id);

if (userService.isCurrentUserNotAnAdmin()

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

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

catService.delete(id);

}

}

Добавлен файл kotiki-java/api/src/main/java/org/itmo/controllers/OwnerController.java:

package org.itmo.controllers;

import lombok.RequiredArgsConstructor;

import org.itmo.dto.OwnerDto;

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")

@RequiredArgsConstructor

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());

}

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

public OwnerDto get(@PathVariable int id) {

if (userService.isCurrentUserNotAnAdmin()

&& userService.getCurrentOwnerId() != id)

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

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

}

@PostMapping("/create")

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

public void create(@RequestBody OwnerDto owner) {

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

}

@PostMapping("/update")

public void update(@RequestBody OwnerDto owner) {

if (userService.isCurrentUserNotAnAdmin()

&& userService.getCurrentOwnerId() != owner.getId())

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

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

throw new ResponseStatusException(HttpStatus.NOT\_FOUND);

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

}

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

public void delete(@PathVariable int id) {

if (userService.isCurrentUserNotAnAdmin()

&& userService.getCurrentOwnerId() != id)

throw new ResponseStatusException(HttpStatus.FORBIDDEN);

ownerService.delete(id);

}

}

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

-package dto;

+package org.itmo.dto;

-import enums.Color;

-import lombok.\*;

+import lombok.Builder;

+import lombok.Data;

+import org.itmo.enums.Color;

import java.time.LocalDate;

import java.util.List;

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

-package dto;

+package org.itmo.dto;

-import lombok.\*;

+import lombok.Builder;

+import lombok.Data;

import java.util.List;

Добавлен файл kotiki-java/api/src/main/java/org/itmo/dto/UserDto.java:

package org.itmo.dto;

import lombok.Builder;

import lombok.Data;

import java.util.List;

@Data

@Builder

public class UserDto {

private int id;

private String username;

private String password;

private Integer ownerId;

private List<String> roles;

}

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

-package services;

+package org.itmo.services;

-import enums.Color;

-import models.Cat;

+import org.itmo.enums.Color;

+import org.itmo.models.Cat;

import java.util.List;

public interface AbstractCatService {

List<Cat> getByColor(Color color);

+ List<Cat> getByOwnerId(int ownerId);

+

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

+

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

+

void save(Cat cat);

void delete(int id);

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

-package services;

+package org.itmo.services;

-import models.Owner;

+import org.itmo.models.Owner;

import java.util.List;

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

-package services;

+package org.itmo.services;

-import dao.CatDao;

-import enums.Color;

-import models.Cat;

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

-import org.springframework.stereotype.Component;

+import lombok.RequiredArgsConstructor;

+import org.itmo.dao.CatDao;

+import org.itmo.enums.Color;

+import org.itmo.models.Cat;

+import org.itmo.tools.DalException;

import org.springframework.stereotype.Service;

-import tools.DalException;

import java.util.List;

import java.util.Optional;

@Service

+@RequiredArgsConstructor

public class CatService implements AbstractCatService {

private final CatDao dao;

- @Autowired

- public CatService(CatDao dao) {

- this.dao = dao;

- }

-

public List<Cat> getAll() {

return dao.findAll();

}

public List<Cat> getByColor(Color color) {

return dao.findByColor(color);

}

+ public List<Cat> getByOwnerId(int ownerId) {

+ return dao.findByOwnerId(ownerId);

+ }

+

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

+ return dao.findByBreedAndOwnerId(breed, ownerId);

+ }

+

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

+ return dao.findByColorAndOwnerId(color, ownerId);

+ }

+

public void save(Cat cat) {

dao.save(cat);

}

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

-package services;

+package org.itmo.services;

-import dao.OwnerDao;

-import models.Cat;

-import models.Owner;

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

+import lombok.RequiredArgsConstructor;

+import org.itmo.dao.OwnerDao;

+import org.itmo.models.Owner;

+import org.itmo.tools.DalException;

import org.springframework.stereotype.Service;

-import tools.DalException;

import java.util.List;

import java.util.Optional;

@Service

+@RequiredArgsConstructor

public class OwnerService implements AbstractOwnerService {

private final OwnerDao dao;

- @Autowired

- public OwnerService(OwnerDao dao) {

- this.dao = dao;

- }

-

public List<Owner> getAll() {

return dao.findAll();

}

Добавлен файл kotiki-java/api/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 org.springframework.stereotype.Service;

@Service

@RequiredArgsConstructor

public class RoleService {

private final RoleDao roleDao;

public Role getByName(String name) {

return roleDao.getByName(name);

}

}

Добавлен файл kotiki-java/api/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.springframework.security.core.GrantedAuthority;

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

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

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.core.userdetails.UsernameNotFoundException;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import java.util.Collection;

import java.util.stream.Collectors;

@Service

@RequiredArgsConstructor

public class UserService implements UserDetailsService {

private final UserDao dao;

@Transactional

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

User user = dao.getByUsername(username);

if (user == null)

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

return new org.springframework.security.core.userdetails.User(

user.getUsername(),

user.getPassword(),

mapRolesToAuthorities(user.getRoles()));

}

public void save(User user) {

dao.save(user);

}

public void delete(int id) {

dao.deleteById(id);

}

public boolean isCurrentUserNotAnAdmin() {

return SecurityContextHolder

.getContext()

.getAuthentication()

.getAuthorities()

.stream()

.noneMatch(a -> a.getAuthority().equals("ROLE\_ADMIN"));

}

public int getCurrentOwnerId() {

return dao

.getByUsername(getCurrentUsername())

.getOwner()

.getId();

}

public int getCurrentId() {

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

}

public String getCurrentUsername() {

Object principal = SecurityContextHolder

.getContext()

.getAuthentication()

.getPrincipal();

return principal instanceof UserDetails

? ((UserDetails)principal).getUsername()

: principal.toString();

}

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

return roles

.stream()

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

.collect(Collectors.toList());

}

}

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

-package utils;

+package org.itmo.utils;

-import dto.CatDto;

-import models.Cat;

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

+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.springframework.stereotype.Component;

-import services.CatService;

-import services.OwnerService;

import java.util.stream.Collectors;

@Component

+@RequiredArgsConstructor

public class CatConverter {

- private final CatService catService;

- private final OwnerService ownerService;

-

- @Autowired

- public CatConverter(CatService catService, OwnerService ownerService) {

- this.catService = catService;

- this.ownerService = ownerService;

- }

+ private final AbstractCatService catService;

+ private final AbstractOwnerService ownerService;

public CatDto toDto(Cat cat) {

return CatDto.builder()

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

-package utils;

+package org.itmo.utils;

-import dto.OwnerDto;

-import models.Cat;

-import models.Owner;

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

+import lombok.RequiredArgsConstructor;

+import org.itmo.dto.OwnerDto;

+import org.itmo.models.Cat;

+import org.itmo.models.Owner;

import org.springframework.stereotype.Component;

-import services.CatService;

+import org.itmo.services.AbstractCatService;

import java.util.stream.Collectors;

@Component

+@RequiredArgsConstructor

public class OwnerConverter {

- private final CatService catService;

-

- @Autowired

- public OwnerConverter(CatService catService) {

- this.catService = catService;

- }

+ private final AbstractCatService catService;

public OwnerDto toDto(Owner owner) {

return OwnerDto.builder()

Добавлен файл kotiki-java/api/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.services.AbstractOwnerService;

import org.itmo.services.auth.RoleService;

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

import org.springframework.stereotype.Component;

import java.util.stream.Collectors;

@Component

@RequiredArgsConstructor

public class UserConverter {

private final AbstractOwnerService ownerService;

private final RoleService roleService;

private final PasswordEncoder passwordEncoder;

public User toModel(UserDto user) {

User.UserBuilder userBuilder = User.builder()

.id(user.getId())

.username(user.getUsername())

.password(passwordEncoder.encode(user.getPassword()));

if (user.getOwnerId() != null)

userBuilder.owner(ownerService.get(user.getOwnerId()));

if (user.getRoles() != null)

userBuilder.roles(user.getRoles()

.stream()

.map(roleService::getByName)

.collect(Collectors.toList()));

return userBuilder.build();

}

}

Удален файл kotiki-java/api/src/main/java/tools/ControllerException.java

Изменен файл kotiki-java/api/src/test/java/services/CatServiceTest.java -> kotiki-java/api/src/test/java/org/itmo/services/CatServiceTest.java:

-package services;

+package org.itmo.services;

-import dao.CatDao;

-import models.Cat;

+import org.itmo.dao.CatDao;

+import org.itmo.models.Cat;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

@@ -13,10 +13,10 @@

public class CatServiceTest {

- private CatService service;

+ private AbstractCatService service;

private CatDao dao;

- @BeforeEach

+ @BeforeEach()

public void setup() {

dao = mock(CatDao.class);

service = new CatService(dao);

Изменен файл kotiki-java/api/src/test/java/services/OwnerServiceTest.java -> kotiki-java/api/src/test/java/org/itmo/services/OwnerServiceTest.java:

-package services;

+package org.itmo.services;

-import dao.OwnerDao;

-import models.Cat;

-import models.Owner;

+import org.itmo.dao.OwnerDao;

+import org.itmo.models.Cat;

+import org.itmo.models.Owner;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

@@ -15,10 +15,10 @@

public class OwnerServiceTest {

- private OwnerService service;

+ private AbstractOwnerService service;

private OwnerDao dao;

- @BeforeEach

+ @BeforeEach()

public void setup() {

dao = mock(OwnerDao.class);

service = new OwnerService(dao);

Изменен файл kotiki-java/dal/pom.xml:

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>kotiki-java</groupId>

- <version>1.0-SNAPSHOT</version>

+ <groupId>org.itmo</groupId>

+ <version>1.0</version>

</parent>

<modelVersion>4.0.0</modelVersion>

Удален файл kotiki-java/dal/src/main/java/dao/CatDao.java

Изменен файл kotiki-java/dal/src/main/java/Main.java -> kotiki-java/dal/src/main/java/org/itmo/Main.java:

+package org.itmo;

+

public class Main {

public static void main(String[] args) {

Добавлен файл kotiki-java/dal/src/main/java/org/itmo/dao/CatDao.java:

package org.itmo.dao;

import org.itmo.enums.Color;

import org.itmo.models.Cat;

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

import org.springframework.stereotype.Repository;

import java.util.List;

@Repository

public interface CatDao extends JpaRepository<Cat, Integer> {

List<Cat> findByBreed(String breed);

List<Cat> findByColor(Color color);

List<Cat> findByOwnerId(int ownerId);

List<Cat> findByBreedAndOwnerId(String breed, int ownerId);

List<Cat> findByColorAndOwnerId(Color color, int ownerId);

}

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

-package dao;

+package org.itmo.dao;

-import models.Owner;

+import org.itmo.models.Owner;

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

import org.springframework.stereotype.Repository;

Добавлен файл kotiki-java/dal/src/main/java/org/itmo/dao/auth/RoleDao.java:

package org.itmo.dao.auth;

import org.itmo.models.auth.Role;

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

public interface RoleDao extends JpaRepository<Role, Integer> {

Role getByName(String name);

}

Добавлен файл kotiki-java/dal/src/main/java/org/itmo/dao/auth/UserDao.java:

package org.itmo.dao.auth;

import org.itmo.models.auth.User;

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

public interface UserDao extends JpaRepository<User, Integer> {

User getByUsername(String username);

}

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

-package enums;

+package org.itmo.enums;

public enum Color {

BLACK,

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

-package models;

+package org.itmo.models;

-import enums.Color;

import lombok.\*;

import org.hibernate.Hibernate;

-import org.hibernate.annotations.CascadeType;

import org.hibernate.annotations.Cascade;

-import tools.DalException;

+import org.hibernate.annotations.CascadeType;

+import org.itmo.enums.Color;

+import org.itmo.tools.DalException;

import javax.persistence.\*;

import java.time.LocalDate;

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

-package models;

+package org.itmo.models;

import lombok.\*;

import org.hibernate.Hibernate;

-import org.hibernate.annotations.CascadeType;

import org.hibernate.annotations.Cascade;

+import org.hibernate.annotations.CascadeType;

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

import javax.persistence.\*;

import java.time.LocalDate;

public class Owner {

@Column(name = "birth\_date")

private LocalDate birthDate;

- @OneToMany()

+ @OneToMany

@Cascade({

CascadeType.PERSIST,

CascadeType.MERGE,

public class Owner {

@ToString.Exclude

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

+ @OneToOne(mappedBy = "owner")

+ @ToString.Exclude

+ private User user;

+

@Override

public boolean equals(Object o) {

if (this == o)

Добавлен файл kotiki-java/dal/src/main/java/org/itmo/models/auth/Role.java:

package org.itmo.models.auth;

import lombok.\*;

import org.hibernate.Hibernate;

import javax.persistence.\*;

import java.util.Objects;

@Entity

@Table(name = "roles")

@NoArgsConstructor

@AllArgsConstructor

@Getter

@Setter

@ToString

@Builder

public class Role {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(nullable = false)

private int id;

@Column(nullable = false)

@NonNull

private String name;

@Override

public boolean equals(Object o) {

if (this == o)

return true;

if (o == null || Hibernate.getClass(this) != Hibernate.getClass(o))

return false;

var role = (Role)o;

return Objects.equals(id, role.id);

}

@Override

public int hashCode() {

return getClass().hashCode();

}

}

Добавлен файл kotiki-java/dal/src/main/java/org/itmo/models/auth/User.java:

package org.itmo.models.auth;

import lombok.\*;

import org.hibernate.Hibernate;

import org.itmo.models.Owner;

import javax.persistence.\*;

import java.util.List;

import java.util.Objects;

@Entity

@Table(name = "users")

@NoArgsConstructor

@AllArgsConstructor

@Getter

@Setter

@ToString

@Builder

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(nullable = false)

private int id;

@Column(nullable = false, unique = true)

@NonNull

private String username;

@Column(nullable = false)

@NonNull

private String password;

@ManyToMany

@JoinTable(name = "users\_roles",

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

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

@ToString.Exclude

private List<Role> roles;

@OneToOne(cascade = CascadeType.ALL)

@JoinColumn(name = "owner\_id")

@ToString.Exclude

private Owner owner;

@Override

public boolean equals(Object o) {

if (this == o)

return true;

if (o == null || Hibernate.getClass(this) != Hibernate.getClass(o))

return false;

var user = (User)o;

return Objects.equals(id, user.id);

}

@Override

public int hashCode() {

return getClass().hashCode();

}

}

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

-package tools;

+package org.itmo.tools;

public class DalException extends RuntimeException {

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

-package models;

+package org.itmo.models;

-import enums.Color;

+import org.itmo.enums.Color;

+import org.itmo.tools.DalException;

import org.junit.jupiter.api.Test;

-import tools.DalException;

import java.lang.reflect.Field;

import java.util.List;

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

-package models;

+package org.itmo.models;

import org.junit.jupiter.api.Test;

-import static org.junit.jupiter.api.Assertions.\*;

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

+import static org.junit.jupiter.api.Assertions.assertThrows;

public class OwnerTest {

Изменен файл kotiki-java/pom.xml:

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

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

<version>2.6.5</version>

- <relativePath/> <!-- lookup parent from repository -->

+ <relativePath/>

</parent>

<modelVersion>4.0.0</modelVersion>

- <groupId>kotiki-java</groupId>

+ <groupId>org.itmo</groupId>

<artifactId>kotiki-java</artifactId>

<packaging>pom</packaging>

- <version>1.0-SNAPSHOT</version>

+ <version>1.0</version>

<modules>

<module>dal</module>

<module>api</module>

@@ -35,29 +35,35 @@

<dependency>

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

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

- <version>2.6.5</version>

+ <version>2.6.6</version>

</dependency>

<dependency>

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

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

- <version>2.6.5</version>

+ <version>2.6.6</version>

</dependency>

<dependency>

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

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

- <version>2.6.5</version>

+ <version>2.6.6</version>

<scope>runtime</scope>

</dependency>

<dependency>

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

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

- <version>2.6.5</version>

+ <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>

@@ -92,7 +98,7 @@

<plugin>

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

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

- <version>2.6.5</version>

+ <version>2.6.6</version>

</plugin>

</plugins>

</build>