**ПРИЛОЖЕНИЕ Г**

Текст программы Г.1 – Текст 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">

<modelVersion>4.0.0</modelVersion>

<groupId>by.bsac.java\_programming\_learning\_platform</groupId>

<artifactId>Java programming learning platform</artifactId>

<version>1.0</version>

<parent>

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

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

<version>2.1.4.RELEASE</version>

</parent>

<dependencies>

<!-- Spring web starter -->

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

<optional>true</optional>

</dependency>

<!-- JPA Data (We are going to use Repositories, Entities, Hibernate, etc...) -->

<dependency>

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

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

</dependency>

<!-- Use MySQL Connector-J -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

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

<artifactId>spring-security-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.springframework.session</groupId>

<artifactId>spring-session-jdbc</artifactId>

</dependency>

<dependency>

<groupId>org.flywaydb</groupId>

<artifactId>flyway-core</artifactId>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

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

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

</plugin>

</plugins>

</build>

</project>

Текст программы Г.2 – Текст application.properties

spring.jpa.hibernate.ddl-auto=validate

spring.datasource.url=jdbc:mysql://localhost:3306/learning\_platform?serverTimezone=UTC

spring.datasource.username=dbuser

spring.datasource.password=dbuser

spring.jpa.show-sql=false

spring.jpa.generate-ddl=false

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL55Dialect

spring.http.log-request-details=true

spring.freemarker.expose-request-attributes=true

upload.path=c:/temp/uploads

upload.path.page.content=c:/temp/uploads/content

spring.mail.host=smtp.yandex.ru

spring.mail.username=babanin-dima1@ya.ru

spring.mail.password=pejqqstjuprnnghb

spring.mail.port=465

spring.mail.protocol=smtps

mail.debug=false

recaptcha.secret=6LdRp6MUAAAAAG03Xn7yHp4WybJIX3hZkwqVIMco

spring.session.jdbc.initialize-schema=always

spring.session.jdbc.table-name=SPRING\_SESSION

Текст программы Г.3 – Исходный код Application.java

package by.bsac;

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);

}

}

Текст программы Г.4 – Исходный код MvcConfig.java

package by.bsac.config;

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

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.client.RestTemplate;

import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;

import org.springframework.web.servlet.config.annotation.ViewControllerRegistry;

import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration

public class MvcConfig implements WebMvcConfigurer {

@Value("${upload.path}")

private String uploadPath;

@Override

public void addViewControllers(ViewControllerRegistry registry) {

registry.addViewController("/").setViewName("index");

}

@Bean

public RestTemplate getRestTemplate() {

return new RestTemplate();

}

@Override

public void addResourceHandlers(ResourceHandlerRegistry registry) {

registry.addResourceHandler("/img/\*\*")

.addResourceLocations("file:///" + uploadPath + "/");

/\*registry.addResourceHandler("/static/\*\*")

.addResourceLocations("classpath:/static/");\*/

}

}

Текст программы Г.5 – Исходный код WebSecurityConfig.java

package by.bsac.config;

import by.bsac.service.UserService;

import org.springframework.context.annotation.Configuration;

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

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.password.PasswordEncoder;

@Configuration

@EnableWebSecurity

@EnableGlobalMethodSecurity(prePostEnabled = true)

public class WebSecurityConfig extends WebSecurityConfigurerAdapter {

private final UserService userService;

private final PasswordEncoder passwordEncoder;

public WebSecurityConfig(UserService userService, PasswordEncoder passwordEncoder) {

this.userService = userService;

this.passwordEncoder = passwordEncoder;

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http

.authorizeRequests()

.antMatchers("/", "/registration", "/activate/\*", "/img/\*", "/course/search").permitAll()

.anyRequest().authenticated()

.and()

.formLogin()

.loginPage("/login")

.permitAll()

.and()

.rememberMe()

.and()

.logout()

.permitAll();

}

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.userDetailsService(userService)

.passwordEncoder(passwordEncoder);

}

}

Текст программы Г.6 – Исходный код MailConfig.java

package by.bsac.config;

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

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.mail.javamail.JavaMailSender;

import org.springframework.mail.javamail.JavaMailSenderImpl;

import java.util.Properties;

@Configuration

public class MailConfig {

@Value("${spring.mail.host}")

private String host;

@Value("${spring.mail.username}")

private String username;

@Value("${spring.mail.password}")

private String password;

@Value("${spring.mail.port}")

private int port;

@Value("${spring.mail.protocol}")

private String protocol;

@Value("${mail.debug}")

private String debug;

@Bean

public JavaMailSender getMailSender() {

JavaMailSenderImpl mailSender = new JavaMailSenderImpl();

mailSender.setHost(host);

mailSender.setUsername(username);

mailSender.setPassword(password);

mailSender.setPort(port);

Properties properties = mailSender.getJavaMailProperties();

properties.setProperty("mail.transport.protocol", protocol);

properties.setProperty("mail.debug", debug);

return mailSender;

}

}

Текст программы Г.7 – Исходный код User.java

package by.bsac.entity;

import by.bsac.entity.content.Course;

import org.springframework.security.core.GrantedAuthority;

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

import javax.persistence.\*;

import java.util.Collection;

import java.util.Objects;

import java.util.Set;

@Entity

public class User implements UserDetails {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String username;

private String password;

private String email;

private String activationCode;

private Boolean active;

@ElementCollection(targetClass = Role.class, fetch = FetchType.EAGER)

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

@Enumerated(EnumType.STRING)

private Set<Role> roles;

@OneToMany(mappedBy = "from", cascade = CascadeType.ALL, fetch = FetchType.LAZY)

private Set<Message> outgoingMessages;

@OneToMany(mappedBy = "to", cascade = CascadeType.ALL, fetch = FetchType.LAZY)

private Set<Message> incomingMessages;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "course\_student",

joinColumns = {@JoinColumn(name = "student\_id")},

inverseJoinColumns = {@JoinColumn(name = "course\_id")}

0 )

private Set<Course> courses;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "editor\_course",

joinColumns = {@JoinColumn(name = "teacher\_id")},

inverseJoinColumns = {@JoinColumn(name = "course\_id")}

)

private Set<Course> editableCourses;

@OneToMany(mappedBy = "user", cascade = CascadeType.ALL, fetch = FetchType.LAZY)

private Set<Grade> grades;

@Override

public Collection<? extends GrantedAuthority> getAuthorities() {

return getRoles();

}

@Override

public String getPassword() {

return password;

}

@Override

public String getUsername() {

return username;

}

@Override

public boolean isAccountNonExpired() {

return true;

}

@Override

public boolean isAccountNonLocked() {

return true;

}

@Override

public boolean isCredentialsNonExpired() {

return true;

}

@Override

public boolean isEnabled() {

return getActive();

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public void setUsername(String username) {

this.username = username;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getActivationCode() {

return activationCode;

}

public void setActivationCode(String activationCode) {

this.activationCode = activationCode;

}

public Boolean getActive() {

return active;

}

public void setActive(Boolean active) {

this.active = active;

}

public Set<Role> getRoles() {

return roles;

}

public void setRoles(Set<Role> roles) {

this.roles = roles;

}

public Set<Message> getOutgoingMessages() {

return outgoingMessages;

}

public void setOutgoingMessages(Set<Message> outgoingMessages) {

this.outgoingMessages = outgoingMessages;

}

public Set<Course> getCourses() {

return courses;

}

public void setCourses(Set<Course> courses) {

this.courses = courses;

}

public Set<Course> getEditableCourses() {

return editableCourses;

}

public void setEditableCourses(Set<Course> editableCourses) {

this.editableCourses = editableCourses;

}

public Set<Message> getIncomingMessages() {

return incomingMessages;

}

public void setIncomingMessages(Set<Message> incomingMessages) {

this.incomingMessages = incomingMessages;

}

public Set<Grade> getGrades() {

return grades;

}

public void setGrades(Set<Grade> grades) {

this.grades = grades;

}

public boolean isAdmin() {

return roles.contains(Role.ADMIN);

}

public boolean isTeacher() {

return roles.contains(Role.TEACHER);

}

public boolean isUser() {

return roles.contains(Role.USER);

}

@Override

public boolean equals(Object o) {

if (this == o) return true;

if (o == null || getClass() != o.getClass()) return false;

User user = (User) o;

return id.equals(user.id);

}

@Override

public int hashCode() {

return Objects.hash(id);

}

}

Текст программы Г.8 – Исходный код Role.java

package by.bsac.entity;

import org.springframework.security.core.GrantedAuthority;

public enum Role implements GrantedAuthority {

USER, TEACHER, ADMIN;

@Override

public String getAuthority() {

return name();

}

}

Текст программы Г.9 – Исходный код Message.java

package by.bsac.entity;

import javax.persistence.\*;

import java.time.LocalDateTime;

@Entity

public class Message {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String title;

private String text;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumn(name = "from\_user\_id")

private User from;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumn(name = "to\_user\_id")

private User to;

@OneToOne

@JoinColumn(name = "document\_id", referencedColumnName = "id")

private Document document;

private LocalDateTime date;

}

Текст программы Г.10 – Исходный код Grade.java

package by.bsac.entity;

import by.bsac.entity.test.Practice;

import by.bsac.entity.test.Test;

import javax.persistence.\*;

import java.time.LocalDate;

@Entity

public class Grade {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumn(name = "user\_id")

private User user;

private Short grade;

private LocalDate date;

@OneToOne

@JoinColumn(name = "test\_id")

private Test test;

@OneToOne

@JoinColumn(name = "practice\_id")

private Practice practice;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public User getUser() {

return user;

}

public void setUser(User user) {

this.user = user;

}

public Short getGrade() {

return grade;

}

public void setGrade(Short grade) {

this.grade = grade;

}

public LocalDate getDate() {

return date;

}

public void setDate(LocalDate date) {

this.date = date;

}

public Test getTest() {

return test;

}

public void setTest(Test test) {

this.test = test;

}

public Practice getPractice() {

return practice;

}

public void setPractice(Practice practice) {

this.practice = practice;

}

}

Текст программы Г.11 – Исходный код Document.java

package by.bsac.entity;

import by.bsac.entity.test.Practice;

import javax.persistence.\*;

@Entity

public class Document {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String filename;

@Enumerated(EnumType.STRING)

private DocumentType type;

@OneToOne(mappedBy = "document")

private Message message;

@OneToOne(mappedBy = "document")

private Practice practice;

}

Текст программы Г.12 – Исходный код Course.java

package by.bsac.entity.content;

import by.bsac.entity.User;

import javax.persistence.\*;

import java.util.Comparator;

import java.util.List;

import java.util.Set;

@Entity

public class Course {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String name;

private String description;

private String imgName;

@OneToMany(mappedBy = "course", cascade = CascadeType.ALL, fetch = FetchType.LAZY)

private List<Topic> topics;

private Boolean active;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "course\_student",

joinColumns = {@JoinColumn(name = "course\_id")},

inverseJoinColumns = {@JoinColumn(name = "student\_id")}

)

private Set<User> students;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "editor\_course",

joinColumns = {@JoinColumn(name = "course\_id")},

inverseJoinColumns = {@JoinColumn(name = "teacher\_id")}

)

private Set<User> editors;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public List<Topic> getTopics() {

if (!topics.isEmpty())

topics.sort(Comparator.comparing(Topic::getPriority));

return topics;

}

public void setTopics(List<Topic> topics) {

this.topics = topics;

}

public Boolean getActive() {

return active;

}

public void setActive(Boolean active) {

this.active = active;

}

public Set<User> getStudents() {

return students;

}

public void setStudents(Set<User> students) {

this.students = students;

}

public Set<User> getEditors() {

return editors;

}

public void setEditors(Set<User> editors) {

this.editors = editors;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public String getImgName() {

return imgName;

}

public void setImgName(String imgName) {

this.imgName = imgName;

}

public Integer getLastPriorityPage() {

List<Topic> topics = getTopics();

if (!topics.isEmpty()) {

return topics.get(topics.size() - 1).getPriority();

}

return 0;

}

}

Текст программы Г.13 – Исходный код Page.java

package by.bsac.entity.content;

import by.bsac.service.Tools;

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

import javax.persistence.\*;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.IOException;

@Entity

public class Page {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String name;

private String path;

private Integer priority;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumn(name = "topic\_id")

private Topic topic;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getPath() {

return path;

}

public void setPath(String path) {

this.path = path;

}

public Integer getPriority() {

return priority;

}

public void setPriority(Integer priority) {

this.priority = priority;

}

public Topic getTopic() {

return topic;

}

public void setTopic(Topic topic) {

this.topic = topic;

}

public String getContent(Tools tools) {

return tools.readContentPage(getPath());

}

}

Текст программы Г.14 – Исходный код Topic.java

package by.bsac.entity.content;

import javax.persistence.\*;

import java.util.Comparator;

import java.util.List;

@Entity

public class Topic {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String name;

private Integer priority;

@OneToMany(mappedBy = "topic", cascade = CascadeType.ALL, fetch = FetchType.LAZY)

private List<Page> pages;

@ManyToOne(fetch = FetchType.LAZY)

@JoinColumn(name = "course\_id")

private Course course;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Integer getPriority() {

return priority;

}

public void setPriority(Integer priority) {

this.priority = priority;

}

public List<Page> getPages() {

if (!pages.isEmpty())

pages.sort(Comparator.comparing(Page::getPriority));

return pages;

}

public void setPages(List<Page> pages) {

this.pages = pages;

}

public Course getCourse() {

return course;

}

public void setCourse(Course course) {

this.course = course;

}

public Integer getLastPriorityPage() {

List<Page> pages = getPages();

if (!pages.isEmpty()) return pages.get(pages.size() - 1).getPriority();

return 0;

}

}

Текст программы Г.15 – Исходный код Answer.java

package by.bsac.entity.test;

import javax.persistence.\*;

@Entity

public class Answer {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String content;

private Boolean isRight;

@ManyToOne(fetch = FetchType.LAZY)

@JoinTable(name = "question\_id")

private Question question;

}

Текст программы Г.16 – Исходный код Practice.java

package by.bsac.entity.test;

import by.bsac.entity.Document;

import by.bsac.entity.Grade;

import javax.persistence.\*;

@Entity

public class Practice {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String name;

private String content;

@OneToOne

@JoinColumn(name = "document\_id")

private Document document;

@OneToOne(mappedBy = "practice")

private Grade grade;

}

Текст программы Г.17 – Исходный код Question.java

package by.bsac.entity.test;

import javax.persistence.\*;

import java.util.Set;

@Entity

public class Question {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String content;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "question\_teg",

joinColumns = {@JoinColumn(name = "tag\_id")},

inverseJoinColumns = {@JoinColumn(name = "question\_id")}

)

private Set<Tag> tags;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "test\_question",

joinColumns = {@JoinColumn(name = "test\_id")},

inverseJoinColumns = {@JoinColumn(name = "question\_id")}

)

private Set<Test> tests;

@OneToMany(mappedBy = "question", cascade = CascadeType.ALL, fetch = FetchType.LAZY)

private Set<Answer> answers;

}

Текст программы Г.18 – Исходный код Tag.java

package by.bsac.entity.test;

import javax.persistence.\*;

import java.util.Set;

@Entity

public class Tag {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String name;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "question\_teg",

joinColumns = {@JoinColumn(name = "question\_id")},

inverseJoinColumns = {@JoinColumn(name = "tag\_id")}

)

private Set<Question> questions;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "test\_tag",

joinColumns = {@JoinColumn(name = "test\_id")},

inverseJoinColumns = {@JoinColumn(name = "tag\_id")}

)

private Set<Test> tests;

}

Текст программы Г.19 – Исходный код Test.java

package by.bsac.entity.test;

import by.bsac.entity.Grade;

import javax.persistence.\*;

import java.util.Set;

@Entity

public class Test {

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

private Long id;

private String name;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "test\_tag",

joinColumns = {@JoinColumn(name = "tag\_id")},

inverseJoinColumns = {@JoinColumn(name = "test\_id")}

)

private Set<Tag> tags;

@ManyToMany(fetch = FetchType.LAZY)

@JoinTable(

name = "test\_question",

joinColumns = {@JoinColumn(name = "question\_id")},

inverseJoinColumns = {@JoinColumn(name = "test\_id")}

)

private Set<Question> questions;

@OneToOne(mappedBy = "test")

private Grade grade;

}

Текст программы Г.20 – Исходный код CaptchaResponseDto.java

package by.bsac.entity.dto;

import com.fasterxml.jackson.annotation.JsonAlias;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

import java.util.Set;

@JsonIgnoreProperties(ignoreUnknown = true)

public class CaptchaResponseDto {

private boolean success;

@JsonAlias("error-codes")

private Set<String> errorCodes;

public boolean isSuccess() {

return success;

}

public void setSuccess(boolean success) {

this.success = success;

}

public Set<String> getErrorCodes() {

return errorCodes;

}

public void setErrorCodes(Set<String> errorCodes) {

this.errorCodes = errorCodes;

}

}

Текст программы Г.21 – Исходный код AdminController.java

package by.bsac.controller;

import by.bsac.entity.Role;

import by.bsac.entity.User;

import by.bsac.service.UserService;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

import java.util.Map;

@Controller

@PreAuthorize("hasAuthority('ADMIN')")

@RequestMapping("/admin")

public class AdminController {

private final UserService userService;

public AdminController(UserService userService) {

this.userService = userService;

}

@GetMapping("/userList")

public String userList(Model model) {

model.addAttribute("userList", userService.findAll());

return "userList";

}

@PostMapping("/user/{user}")

public String editUser(

@PathVariable User user,

@RequestParam String action,

@RequestParam Map<String, String> form,

Model model

) {

if (action.equals("delete")) {

userService.deleteUser(user);

return "redirect:/admin/userList";

} else if (action.equals("save")) {

userService.save(user, form);

return "redirect:/admin/userList";

} else {

model.addAttribute("userList", userService.findAll());

model.addAttribute("userEditId", user.getId());

model.addAttribute("roles", Role.values());

return "userList";

}

}

}

Текст программы Г.22 – Исходный код ConstructorController.java

package by.bsac.controller;

import by.bsac.dao.CourseRepository;

import by.bsac.dao.PageRepository;

import by.bsac.dao.TopicRepository;

import by.bsac.entity.User;

import by.bsac.entity.content.Course;

import by.bsac.entity.content.Page;

import by.bsac.entity.content.Topic;

import by.bsac.service.Tools;

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

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

import org.springframework.security.core.annotation.AuthenticationPrincipal;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.util.StringUtils;

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

import org.springframework.web.multipart.MultipartFile;

import java.io.File;

import java.io.IOException;

import java.util.\*;

@Controller

@RequestMapping("/constructor")

@PreAuthorize("hasAuthority('TEACHER')")

public class ConstructorController {

private final CourseRepository courseRepository;

private final TopicRepository topicRepository;

private final PageRepository pageRepository;

private final Tools tools;

@Value("${upload.path}")

private String pathImgCourse;

public ConstructorController(CourseRepository courseRepository, TopicRepository topicRepository, PageRepository pageRepository, Tools tools) {

this.courseRepository = courseRepository;

this.topicRepository = topicRepository;

this.pageRepository = pageRepository;

this.tools = tools;

}

@GetMapping("/course")

public String courseConstructor() {

return "constructorCourse";

}

@PostMapping("/course")

public String createCourse(

@AuthenticationPrincipal User user,

Course course,

@RequestParam MultipartFile imgFile

) throws IOException {

Set<User> editors = new HashSet<>();

editors.add(user);

course.setEditors(editors);

if (!imgFile.isEmpty()) {

course.setImgName(tools.saveImg(imgFile));

}

Course save = courseRepository.save(course);

return "redirect:/constructor/course/edit/" + save.getId();

}

@PostMapping("/course/{course}")

public String saveCourse(

@PathVariable Course course,

@RequestParam String name,

@RequestParam String description,

@RequestParam(required = false) Boolean active,

@RequestParam MultipartFile imgFile

) throws IOException {

course.setName(name);

course.setDescription(description);

course.setActive(active);

if (!imgFile.isEmpty()) {

course.setImgName(tools.saveImg(imgFile));

}

courseRepository.save(course);

return "redirect:/constructor/course/edit/" + course.getId();

}

@GetMapping("/course/edit/{course}")

public String editCourse(@PathVariable Course course, Model model) {

model.addAttribute("course", course);

List<Topic> topics = course.getTopics();

Integer newPriorityTopic = 0;

if (!topics.isEmpty()) newPriorityTopic = topics.get(topics.size() - 1).getPriority();

model.addAttribute("newPriorityTopic", newPriorityTopic + 1);

return "editorCourse";

}

@GetMapping("/course/{course}/imgdelete")

public String deleteImgCourse(@PathVariable Course course) {

File imgCourse = new File(this.pathImgCourse, course.getImgName());

imgCourse.delete();

course.setImgName(null);

courseRepository.save(course);

return "redirect:/constructor/course/edit/" + course.getId();

}

@GetMapping("/course/delete/{course}")

public String deleteCourse(@PathVariable Course course) {

for (Topic topic : course.getTopics()) {

for (Page page : topic.getPages()) {

if (!StringUtils.isEmpty(page.getPath())) {

tools.deleteContentPage(page.getPath());

}

}

}

if (!StringUtils.isEmpty(course.getImgName())) {

File imgCourse = new File(this.pathImgCourse, course.getImgName());

imgCourse.delete();

}

courseRepository.delete(course);

return "redirect:/";

}

@GetMapping("/course/{course}/topic/create")

public String createTopic(

@PathVariable Course course,

Topic topic

) {

List<Topic> topics = course.getTopics();

if (topics.isEmpty()) {

topics = new ArrayList<>();

}

topics.add(topic);

course.setTopics(topics);

courseRepository.save(course);

return "redirect:/constructor/course/edit/" + course.getId();

}

@GetMapping("/course/{course}/topic/{topic}")

public String actionTopic(

@PathVariable Course course,

@PathVariable Topic topic,

@RequestParam(required = false) Integer priority,

@RequestParam(required = false) String name,

@RequestParam String action,

Model model

) {

if (action.equals("edit")) {

model.addAttribute("course", course);

model.addAttribute("topicEditId", topic.getId());

return "editorCourse";

} else if (action.equals("delete")) {

topic.getPages().forEach(page -> tools.deleteContentPage(page.getPath()));

topicRepository.delete(topic);

return "redirect:/constructor/course/edit/" + course.getId();

} else {

if (priority != null) {

topic.setPriority(priority);

}

if (!StringUtils.isEmpty(name)) {

topic.setName(name);

}

topicRepository.save(topic);

return "redirect:/constructor/course/edit/" + course.getId();

}

}

@GetMapping("/course/{course}/topic/{topic}/page/create")

public String createPage(

@PathVariable Course course,

@PathVariable Topic topic,

Page page

) {

page.setTopic(topic);

Page save = pageRepository.save(page);

return "redirect:/constructor/course/" + course.getId() + "/topic/" + topic.getId() + "/page/" + save.getId() + "?action=edit";

}

@GetMapping("/course/{course}/topic/{topic}/page/{page}")

public String actionPage(

@PathVariable Course course,

@PathVariable Topic topic,

@PathVariable Page page,

@RequestParam String action,

Model model

) {

if (action.equals("delete")) {

tools.deleteContentPage(page.getPath());

pageRepository.delete(page);

return "redirect:/constructor/course/edit/" + course.getId();

} else {

model.addAttribute(course);

model.addAttribute(topic);

model.addAttribute(page);

if (!StringUtils.isEmpty(page.getPath())) {

model.addAttribute("content", tools.readContentPage(page.getPath()));

}

return "editPage";

}

}

@PostMapping("/course/{course}/topic/{topic}/page/{page}/preview")

public String previewPage(

@PathVariable Course course,

@PathVariable Topic topic,

@PathVariable Page page,

@RequestParam Integer priority,

@RequestParam String name,

@RequestParam String content,

@RequestParam MultipartFile contentFile,

Model model

) {

if (!contentFile.isEmpty()) {

content = tools.readContentPage(contentFile);

}

model.addAttribute("priority", priority);

model.addAttribute("name", name);

model.addAttribute("content", content);

model.addAttribute("contentBit", tools.convertStringToStringByte(content, " "));

return "previewPage";

}

@PostMapping("/course/{course}/topic/{topic}/page/{page}/save")

public String savePage(

@PathVariable Course course,

@PathVariable Topic topic,

@PathVariable Page page,

@RequestParam Integer priority,

@RequestParam String name,

@RequestParam String contentBit

) {

page.setPriority(priority);

page.setName(name);

String fileName = tools.writeContentPage(tools.convertStringByteToString(contentBit, " "));

tools.deleteContentPage(page.getPath());

page.setPath(fileName);

pageRepository.save(page);

return "redirect:/constructor/course/edit/" + course.getId();

}

}

Текст программы Г.23 – Исходный код ContentController.java

package by.bsac.controller;

import by.bsac.entity.User;

import by.bsac.entity.content.Course;

import by.bsac.service.ContentService;

import by.bsac.service.Tools;

import org.springframework.security.core.annotation.AuthenticationPrincipal;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

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

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

@Controller

public class ContentController {

private final ContentService contentService;

private final Tools tools;

public ContentController(ContentService contentService, Tools tools) {

this.contentService = contentService;

this.tools = tools;

}

@GetMapping

public String greeting(@AuthenticationPrincipal User user, Model model) {

Iterable<Course> courses = contentService.getCourses(true);

model.addAttribute("courses", courses);

if (user != null) {

model.addAttribute("coursesEdit", contentService.getCourses(user));

}

return "index";

}

@GetMapping("/course/{course}")

public String openCourse(@PathVariable Course course, Model model) {

model.addAttribute(course);

model.addAttribute(tools);

return "openCourse";

}

@GetMapping("/course/{course}/students")

public String studentsCourse(@PathVariable Course course, Model model) {

model.addAttribute("userList", course.getStudents());

return "userList";

}

@GetMapping("/course/{course}/editors")

public String editorsCourse(@PathVariable Course course, Model model) {

model.addAttribute("userList", course.getEditors());

return "userList";

}

@GetMapping("/course/search")

public String searchCourse(

@AuthenticationPrincipal User user,

@RequestParam String valueSearch,

Model model) {

valueSearch = '%' + valueSearch + '%';

model.addAttribute("courses", contentService.searchCoursesByName(true, valueSearch));

if (user != null) {

model.addAttribute("coursesEdit", contentService.searchCoursesByName(user, valueSearch));

}

return "index";

}

}

Текст программы Г.24 – Исходный код RegistrationController.java

package by.bsac.controller;

import by.bsac.entity.User;

import by.bsac.entity.dto.CaptchaResponseDto;

import by.bsac.service.Tools;

import by.bsac.service.UserService;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.util.StringUtils;

import org.springframework.validation.BindingResult;

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

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

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

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

import org.springframework.web.client.RestTemplate;

import javax.validation.Valid;

import java.util.Collections;

import java.util.Map;

@Controller

public class RegistrationController {

private final static String CAPTCHA\_URL = "https://www.google.com/recaptcha/api/siteverify?secret=%s&response=%s";

@Value("${recaptcha.secret}")

private String captchaSecret;

private final UserService userService;

private final RestTemplate restTemplate;

public RegistrationController(UserService userService, RestTemplate restTemplate) {

this.userService = userService;

this.restTemplate = restTemplate;

}

@GetMapping("/registration")

public String registration() {

return "registration";

}

@PostMapping("/registration")

public String addNewUser(

@RequestParam("g-recaptcha-response") String captchaResponse,

@RequestParam("password2") String passwordConfirmation,

@Valid User user,

BindingResult bindingResult,

Model model

) {

String url = String.format(CAPTCHA\_URL, captchaSecret, captchaResponse);

CaptchaResponseDto response = restTemplate.postForObject(url, Collections.EMPTY\_LIST, CaptchaResponseDto.class);

boolean isSuccessCaptcha = !response.isSuccess();

if (isSuccessCaptcha) {

model.addAttribute("captchaError", "Fill captcha");

}

boolean isConfirmEmpty = StringUtils.isEmpty(passwordConfirmation);

if (isConfirmEmpty) {

model.addAttribute("password2Error", "Password confirmation can not empty");

}

if (user.getPassword() != null && !user.getPassword().equals(passwordConfirmation)) {

model.addAttribute("passwordError", "Password are different");

}

if (bindingResult.hasErrors() || isConfirmEmpty || isSuccessCaptcha) {

Map<String, String> errorMap = Tools.getErrorMap(bindingResult);

model.mergeAttributes(errorMap);

return "registration";

}

if (!userService.addUser(user)) {

model.addAttribute("message", "User exists!");

return "registration";

}

return "redirect:/login";

}

@GetMapping("/activate/{code}")

public String activate(Model model, @PathVariable String code) {

boolean isActivated = userService.activateUser(code);

if (isActivated) {

model.addAttribute("messageType", "success");

model.addAttribute("message", "User successfully activated!");

} else {

model.addAttribute("messageType", "danger");

model.addAttribute("message", "Activation code is not found!");

}

return "login";

}

@GetMapping("/login")

public String login() {

return "login";

}

}

Текст программы Г.25 – Исходный код UserController.java

package by.bsac.controller;

import by.bsac.entity.User;

import by.bsac.entity.content.Course;

import by.bsac.service.UserService;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

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

@Controller

@RequestMapping("/user")

@PreAuthorize("hasAuthority('USER')")

public class UserController {

private final UserService userService;

public UserController(UserService userService) {

this.userService = userService;

}

@GetMapping("/{user}")

public String profile(

@PathVariable User user,

Model model

) {

model.addAttribute("uuser", user);

model.addAttribute("courses", user.getCourses());

model.addAttribute("editableCourses", user.getEditableCourses());

model.addAttribute("roles", user.getRoles());

if (!user.getGrades().isEmpty()) {

model.addAttribute("grades", user.getGrades());

}

return "profile";

}

@GetMapping("/{user}/edit")

public String editProfile(

@PathVariable User user,

Model model

) {

model.addAttribute("user", user);

return "editProfile";

}

@PostMapping("/{user}/edit")

public String saveProfile(

@PathVariable User user,

@RequestParam String username,

@RequestParam String password,

@RequestParam String email

) {

userService.updateUser(user, username, password, email);

return "redirect:/user/" + user.getId();

}

@GetMapping("/{user}/unsubscribe/{course}")

public String unsubscribe(

@PathVariable User user,

@PathVariable Course course

) {

user.getCourses().remove(course);

userService.updateUser(user);

return "redirect:/";

}

@GetMapping("/{user}/subscribe/{course}")

public String subscribe(

@PathVariable User user,

@PathVariable Course course

) {

user.getCourses().add(course);

userService.updateUser(user);

return "redirect:/";

}

}

Текст программы Г.26 – Исходный код ContentService.java

package by.bsac.service;

import by.bsac.dao.CourseRepository;

import by.bsac.entity.User;

import by.bsac.entity.content.Course;

import org.springframework.stereotype.Service;

@Service

public class ContentService {

private final CourseRepository courseRepository;

public ContentService(CourseRepository courseRepository) {

this.courseRepository = courseRepository;

}

public Iterable<Course> getCourses() {

return courseRepository.findAll();

}

public Iterable<Course> getCourses(Boolean active) {

return courseRepository.findCoursesByActive(active);

}

public Iterable<Course> getCourses(User editor) {

return courseRepository.findCoursesByEditorsContains(editor);

}

public Iterable<Course> searchCoursesByName(Boolean active, String regex) {

return courseRepository.findCoursesByActiveAndNameLike(active, regex);

}

public Iterable<Course> searchCoursesByName(User editor, String regex) {

return courseRepository.findCoursesByEditorsContainsAndNameLike(editor, regex);

}

}

Текст программы Г.27 – Исходный код MailSend.java

package by.bsac.service;

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

import org.springframework.mail.SimpleMailMessage;

import org.springframework.mail.javamail.JavaMailSender;

import org.springframework.stereotype.Service;

@Service

public class MailSend {

@Value("${spring.mail.username}")

private String username;

private final JavaMailSender mailSender;

public MailSend(JavaMailSender mailSender) {

this.mailSender = mailSender;

}

public void send(String emailTo, String subject, String message) {

SimpleMailMessage mailMessage = new SimpleMailMessage();

mailMessage.setFrom(username);

mailMessage.setTo(emailTo);

mailMessage.setSubject(subject);

mailMessage.setText(message);

mailSender.send(mailMessage);

}

}

Текст программы Г.28 – Исходный код Tools.java

package by.bsac.service;

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

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;

import org.springframework.util.StringUtils;

import org.springframework.validation.BindingResult;

import org.springframework.validation.FieldError;

import org.springframework.web.multipart.MultipartFile;

import java.io.\*;

import java.nio.charset.StandardCharsets;

import java.util.Map;

import java.util.StringJoiner;

import java.util.UUID;

import java.util.stream.Collector;

import java.util.stream.Collectors;

@Component

public class Tools {

@Value("${upload.path.page.content}")

private String pathContentPage;

@Value("${upload.path}")

private String pathImgCourse;

@Bean

public static PasswordEncoder getPasswordEncoder() {

return new BCryptPasswordEncoder(8);

}

public String readContentPage(String fileName) {

StringBuilder builder = new StringBuilder();

try (BufferedReader reader = new BufferedReader(

new FileReader(new File(pathContentPage, fileName)))) {

reader.lines().forEach(s -> {

builder.append(s);

builder.append(System.getProperty("line.separator"));

});

} catch (IOException e) {

e.printStackTrace();

}

return builder.toString();

}

public String readContentPage(MultipartFile contentFile) {

StringBuilder builder = new StringBuilder();

try (BufferedReader reader = new BufferedReader(new InputStreamReader(contentFile.getInputStream()))) {

reader.lines().forEach(s -> {

builder.append(s);

builder.append(System.getProperty("line.separator"));

});

} catch (IOException e) {

e.printStackTrace();

}

return builder.toString();

}

public String writeContentPage(String content) {

String uuidFilename = UUID.randomUUID().toString();

String filename = uuidFilename + ".txt";

File path = new File(pathContentPage, filename);

try (FileWriter writer = new FileWriter(path)) {

writer.write(content);

} catch (IOException e) {

e.printStackTrace();

}

return filename;

}

public String convertStringToStringByte(String content, String separator) {

StringJoiner joiner = new StringJoiner(separator);

for (byte b : content.getBytes()) joiner.add(Byte.toString(b));

return joiner.toString();

}

public String convertStringByteToString(String stringByte, String separator) {

String[] contentBites = stringByte.split(separator);

byte[] bytes = new byte[contentBites.length];

for (int i = 0; i < bytes.length; i++) {

bytes[i] = Byte.valueOf(contentBites[i]);

}

return new String(bytes, StandardCharsets.UTF\_8);

}

public void deleteContentPage(String path) {

if (!StringUtils.isEmpty(path)) {

File file = new File(pathContentPage, path);

file.delete();

}

}

public String saveImg(MultipartFile file) throws IOException {

String uuidFilename = UUID.randomUUID().toString();

String filename = uuidFilename + "-" + file.getOriginalFilename();

File pathImgCourse = new File(this.pathImgCourse, filename);

file.transferTo(pathImgCourse);

return filename;

}

public static Map<String, String> getErrorMap(BindingResult bindingResult) {

Collector<FieldError, ?, Map<String, String>> collector = Collectors.toMap(

fieldError -> (fieldError.getField()) + "Error",

FieldError::getDefaultMessage

);

return bindingResult.getFieldErrors().stream().collect(collector);

}

}

Текст программы Г.29 – Исходный код UserService.java

package by.bsac.service;

import by.bsac.dao.UserRepository;

import by.bsac.entity.Role;

import by.bsac.entity.User;

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

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

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

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

import org.springframework.stereotype.Service;

import org.springframework.util.StringUtils;

import java.util.\*;

import java.util.stream.Collectors;

@Service

public class UserService implements UserDetailsService {

private final UserRepository userRepository;

private final MailSend mailSend;

private final PasswordEncoder passwordEncoder;

public UserService(UserRepository userRepository, MailSend mailSend, PasswordEncoder passwordEncoder) {

this.userRepository = userRepository;

this.mailSend = mailSend;

this.passwordEncoder = passwordEncoder;

}

@Override

public UserDetails loadUserByUsername(String s) throws UsernameNotFoundException {

User byUsername = userRepository.findByUsername(s);

if (byUsername == null) {

throw new UsernameNotFoundException("User not found");

}

return byUsername;

}

public boolean addUser(User user) {

User byUsername = userRepository.findByUsername(user.getUsername());

if (byUsername != null) {

return false;

}

user.setActive(true);

user.setRoles(Collections.singleton(Role.USER));

user.setActivationCode(UUID.randomUUID().toString());

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

userRepository.save(user);

sendActivationCode(user);

return true;

}

private void sendActivationCode(User user) {

if (!StringUtils.isEmpty(user.getEmail())) {

String message = String.format(

"Hello %s!\n" +

"Welcome! Please, visit this link: http://localhost:8080/activate/%s",

user.getUsername(),

user.getActivationCode()

);

mailSend.send(user.getEmail(), "Activation code mail", message);

}

}

public boolean activateUser(String code) {

User byActivationCode = userRepository.findByActivationCode(code);

if (byActivationCode == null) {

return false;

}

byActivationCode.setActivationCode(null);

userRepository.save(byActivationCode);

return true;

}

public Iterable<User> findAll() {

return userRepository.findAll();

}

public void save(User user, Map<String, String> form) {

Set<String> roleSet = Arrays.stream(Role.values())

.map(Role::name)

.collect(Collectors.toSet());

user.getRoles().clear();

form.keySet().stream()

.filter(roleSet::contains)

.forEach(s -> user.getRoles().add(Role.valueOf(s)));

userRepository.save(user);

}

public void updateUser(User user, String username, String password, String email) {

boolean isEmailChanged = (!StringUtils.isEmpty(email) && !email.equals(user.getEmail()));

if (isEmailChanged) {

user.setEmail(email);

user.setActivationCode(UUID.randomUUID().toString());

}

if (!StringUtils.isEmpty(password)) {

user.setPassword(password);

}

if (!StringUtils.isEmpty(username)) {

user.setUsername(username);

}

userRepository.save(user);

if (isEmailChanged) {

sendActivationCode(user);

}

}

public void deleteUser(User user) {

userRepository.delete(user);

}

public void updateUser(User user) {

userRepository.save(user);

}

}

Текст программы Г.30 – Исходный код CourseRepository.java

package by.bsac.dao;

import by.bsac.entity.User;

import by.bsac.entity.content.Course;

import org.springframework.data.repository.CrudRepository;

public interface CourseRepository extends CrudRepository<Course, Long> {

Iterable<Course> findCoursesByActive(Boolean active);

Iterable<Course> findCoursesByEditorsContains(User editor);

Iterable<Course> findCoursesByActiveAndNameLike(Boolean active, String regex);

Iterable<Course> findCoursesByEditorsContainsAndNameLike(User editor, String regex);

}

Текст программы Г.31 – Исходный код PageRepository.java

package by.bsac.dao;

import by.bsac.entity.content.Page;

import org.springframework.data.repository.CrudRepository;

public interface PageRepository extends CrudRepository<Page, Long> {

}

Текст программы Г.32 – Исходный код TopicRepository.java

package by.bsac.dao;

import by.bsac.entity.content.Topic;

import org.springframework.data.repository.CrudRepository;

public interface TopicRepository extends CrudRepository<Topic, Long> {

}

Текст программы Г.33 – Исходный код UserRepository.java

package by.bsac.dao;

import by.bsac.entity.User;

import org.springframework.data.repository.CrudRepository;

public interface UserRepository extends CrudRepository<User, Long> {

User findByUsername(String username);

User findByActivationCode(String code);

}

Текст программы Г.34 – Исходный код common.ftl

<#macro page>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Учебная платформа БГАС</title>

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

<script src="https://www.google.com/recaptcha/api.js" async defer></script>

</head>

<body>

<#include "navbar.ftl">

<div class="container mt-5">

<#nested>

</div>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>

</body>

</html>

</#macro>

Текст программы Г.35 – Исходный код commonCourse.ftl

<#macro page>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>${course.name}</title>

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"

integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"

crossorigin="anonymous">

<script src="https://www.google.com/recaptcha/api.js" async defer></script>

</head>

<body>

<#include "navbar.ftl">

<div class="container-fluid">

<div class="row">

<nav class="col d-md-block sidebar">

<div class="accordion mt-3" id="accordion">

<#list course.topics as topic>

<div class="card">

<div class="card-header" id="headingTopic${topic.id}">

<h2 class="mb-0">

<button class="btn btn-link collapsed" type="button" data-toggle="collapse"

data-target="#collapseTopic${topic.id}" aria-expanded="true"

aria-controls="collapseTopic${topic.id}">

${topic.name}

</button>

</h2>

</div>

<div id="collapseTopic${topic.id}" class="collapse"

aria-labelledby="headingTopic${topic.id}"

data-parent="#accordion">

<div class="card-body">

<div class="nav flex-column nav-pills" id="v-pills-topic${topic.id}-tab"

role="tablist"

aria-orientation="vertical">

<#list topic.pages as page>

<a class="nav-link" id="v-pills-topic${topic.id}-tab"

data-toggle="pill"

href="#v-pills-topic${topic.id}-page${page.id}"

role="tab" aria-controls="v-pills-topic${topic.id}-page${page.id}" aria-selected="true">${page.name}</a>

</#list>

</div>

</div>

</div>

</div>

</#list>

</div>

</nav>

<main role="main" class="col-md-9 ml-sm-auto col-lg-10 px-4">

<div class="container mt-5">

<#nested>

</div>

</main>

</div>

</div>

<!-- Optional JavaScript -->

<!-- jQuery first, then Popper.js, then Bootstrap JS -->

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"

integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"

crossorigin="anonymous"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"

integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"

crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"

integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"

crossorigin="anonymous"></script>

</body>

</html>

</#macro>

Текст программы Г.36 – Исходный код coursesList.ftl

<#include "security.ftl">

<#macro coursesList courses>

<div class="card-columns">

<#list courses as course>

<div class="card my-3">

<div class="card-header">

<h4><a href="/course/${course.id}">${course.name}</a></h4>

</div>

<#if course.imgName??>

<img class="card-img-top" src="/img/${course.imgName}">

</#if>

<div class="card-body">

<#if user??>

<#if course.students?seq\_contains(user)>

<form action="/user/${user.id}/unsubscribe/${course.id}">

<button type="submit" class="btn btn-primary mb-2">

Отписаться <span class="badge badge-light">${course.students?size}</span>

</button>

</form>

<#else>

<form action="/user/${user.id}/subscribe/${course.id}">

<button type="submit" class="btn btn-primary mb-2">

Подписаться <span class="badge badge-light">${course.students?size}</span>

</button>

</form>

</#if>

</#if>

<blockquote class="blockquote mb-2">

<p class="h6">${course.description}</p>

<footer class="blockquote-footer">

Модераторы

<#list course.editors as editor>

<a href="/user/${editor.id}">

<cite title="Source Title">${editor.username}</cite>

</a>

</#list>

</footer>

</blockquote>

<#if course.editors?seq\_contains(user)>

<form action="/constructor/course/delete/${course.id}" method="get">

<a class="btn btn-secondary"

href="/constructor/course/edit/${course.id}">Редактировать</a>

<button type="submit" class="btn btn-secondary">Удалить</button>

</form>

</#if>

</div>

</div>

<#else>

NO COURSES

</#list>

</div>

</#macro>

Текст программы Г.37 – Исходный код editorCourse.ftl

<#macro editorCourse isCreate>

<h1 class="mb-4"><#if isCreate>Создание курса<#else>Редактирование курса</#if></h1>

<#if !isCreate>

<a class="btn btn-primary" data-toggle="collapse" href="#editCourse" role="button" aria-expanded="false"

aria-controls="editCourse">

Другое

</a>

</#if>

<div class="collapse <#if isCreate>show</#if>" id="editCourse">

<div class="form-group mt-3">

<form action="<#if isCreate>/constructor/course<#else>/constructor/course/${course.id}</#if>" method="post"

enctype="multipart/form-data">

<input type="hidden" name="\_csrf" value="${\_csrf.token}">

<div class="form-group row">

<div class="col">

<div class="input-group mb-2">

<div class="input-group-prepend">

<div class="input-group-text">Название курса</div>

</div>

<input type="text" class="form-control" name="name"

value="<#if course??>${course.name}</#if>">

</div>

</div>

<div class="file-field">

<div class="btn btn-primary btn-sm">

<span>Картинка</span>

<input type="file" name="imgFile" accept="image/\*">

</div>

</div>

</div>

<div class="form-group row">

<div class="col">

<div class="input-group mb-2">

<div class="input-group-prepend">

<div class="input-group-text">Описание</div>

</div>

<textarea rows="4" class="form-control"

name="description"><#if course??>${course.description}</#if></textarea>

</div>

</div>

</div>

<#if !isCreate && course.imgName??>

<div class="form-group row">

<div class="text-center">

<img class="rounded" src="/img/${course.imgName}" alt="">

</div>

</div>

</#if>

<div class="form-group row">

<div class="col-2">

<button type="submit" class="btn btn-primary">

<#if isCreate>Создать курс<#else>Сохранить изменения</#if>

</button>

</div>

<div class="col">

<div class="form-check">

<input class="form-check-input" type="checkbox" id="activeCourse" name="active"

<#if !isCreate && course.active??>checked</#if>>

<label class="form-check-label" for="activeCourse">

Активировать курс

</label>

</div>

</div>

</div>

</form>

<#if !isCreate && course.imgName??>

<form action="/constructor/course/${course.id}/imgdelete" method="get">

<button type="submit" class="btn btn-primary">Удалить изображение</button>

</form>

</#if>

</div>

</div>

</#macro>

Текст программы Г.38 – Исходный код login.ftl

<#include "security.ftl">

<#macro login path isRegitrationForm>

<form action="${path}" method="post">

<input type="hidden" name="\_csrf" value="${\_csrf.token}">

<div class="form-group row">

<label class="col-sm-2 col-form-label">Имя:</label>

<div class="col-sm-4">

<input class="form-control ${(usernameError??)?string('is-invalid', '')}" type="text"

name="username" placeholder="Имя пользователя" value="<#if user??>${user.username}</#if>"/>

<#if usernameError??>

<div class="invalid-feedback">

${usernameError}

</div>

</#if>

</div>

</div>

<#if isRegitrationForm>

<div class="form-group row">

<label class="col-sm-2 col-form-label">Email:</label>

<div class="col-sm-4">

<input class="form-control ${(emailError??)?string('is-invalid', '')}" type="email"

name="email" placeholder="some@some.com" value="<#if user??>${user.email}</#if>"/>

<#if emailError??>

<div class="invalid-feedback">

${emailError}

</div>

</#if>

</div>

</div>

</#if>

<div class="form-group row">

<label class="col-sm-2 col-form-label">Пароль:</label>

<div class="col-sm-4">

<input class="form-control ${(passwordError??)?string('is-invalid', '')}" type="password"

name="password" placeholder="Пароль"/>

<#if passwordError??>

<div class="invalid-feedback">

${passwordError}

</div>

</#if>

</div>

</div>

<#if isRegitrationForm>

<div class="form-group row">

<label class="col-sm-2 col-form-label">Повторите пароль:</label>

<div class="col-sm-4">

<input class="form-control ${(password2Error??)?string('is-invalid', '')}" type="password"

name="password2" placeholder="Повторите пароль"/>

<#if password2Error??>

<div class="invalid-feedback">

${password2Error}

</div>

</#if>

</div>

</div>

<div class="col-sm-4 mb-3">

<div class="g-recaptcha" data-sitekey="6LdRp6MUAAAAAB\_WIUApcC71Pq2ymjCGJdvF4g8T"></div>

</div>

<#if captchaError??>

<div class="alert alert-danger" role="alert">

${captchaError}

</div>

</#if>

</#if>

<button class="btn btn-primary mb-2 mr-4" type="submit">

<#if isRegitrationForm>

Зарегестрироваться

<#else>

Войти

</#if>

</button>

<#if !isRegitrationForm>

<a href="/registration">Регестрация</a>

</#if>

</form>

</#macro>

<#macro logout>

<form action="/logout" method="post">

<input type="hidden" name="\_csrf" value="${\_csrf.token}">

<button class="btn btn-primary mb-2" type="submit"><#if user??>Выйти<#else>Войти</#if></button>

</form>

</#macro>

Текст программы Г.39 – Исходный код navbar.ftl

<#include "security.ftl">

<#import "login.ftl" as l>

<nav class="navbar navbar-expand-lg navbar-dark bg-dark">

<a class="navbar-brand" href="/">Learning platform</a>

<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent"

aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">

<ul class="navbar-nav mr-auto">

<li class="nav-item">

<a class="nav-link" href="/">Главная</a>

</li>

<#if isTeacher>

<li class="nav-item dropdown">

<a class="nav-link dropdown-toggle" href="#" id="navbarDropdownMenuLink" role="button"

data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">

Конструкторы

</a>

<div class="dropdown-menu" aria-labelledby="navbarDropdownMenuLink">

<a class="dropdown-item" href="/constructor/course">Курсов</a>

<a class="dropdown-item" href="#">Тестов</a>

<a class="dropdown-item" href="#">Практики</a>

</div>

</li>

</#if>

<#if isAdmin>

<li class="nav-item">

<a class="nav-link" href="/admin/userList">Пользователи</a>

</li>

</#if>

</ul>

<form action="/course/search" method="get">

<input class="form-control form-control-dark w-100" type="text" placeholder="Поиск курса"

aria-label="Поиск курса" name="valueSearch">

</form>

<ul class="navbar-nav px-1">

<li class="nav-item text-nowrap">

<#if user??>

<a class="nav-link" href="/user/${(user.id)!}">${username}</a>

<#else>

<a class="nav-link" href="/registration">Регистрация</a>

</#if>

</li>

<li class="nav-item text-nowrap">

<@l.logout/>

</li>

</ul>

</div>

</nav>

Текст программы Г.40 – Исходный код security.ftl

<#assign

known = Session.SPRING\_SECURITY\_CONTEXT??

>

<#if known>

<#assign

user = Session.SPRING\_SECURITY\_CONTEXT.authentication.principal

username = user.getUsername()

isAdmin = user.isAdmin()

isTeacher = user.isTeacher()

isUser = user.isUser()

currentUserId = user.getId()

>

<#else>

<#assign

username = "none"

isAdmin = false

isTeacher = false

isUser = false

currentUserId = -1

>

</#if>

Текст программы Г.40 – Исходный код constructorCourse.ftl

<#import "parts/common.ftl" as c>

<#import "parts/editorCourse.ftl" as ec>

<@c.page>

<@ec.editorCourse true/>

</@c.page>

Текст программы Г.41 – Исходный код editorCourse.ftl

<#import "parts/common.ftl" as c>

<#import "parts/editorCourse.ftl" as ec>

<@c.page>

<@ec.editorCourse false/>

<div class="table-responsive">

<table class="table table-sm">

<thead>

<tr>

<th scope="col">#</th>

<th scope="col">Название темы</th>

<th scope="col">Количество страниц</th>

<th scope="col">Действия</th>

</tr>

</thead>

<tbody>

<#list course.topics as topic>

<tr>

<th scope="row">

<#if topicEditId?? && topicEditId == topic.id>

<input type="number" class="form-control" placeholder="Номер темы" form="editTopic${topic.id}"

name="priority" value="${topic.priority}">

<#else>

${topic.priority}

</#if>

</th>

<td>

<#if topicEditId?? && topicEditId == topic.id>

<input type="text" class="form-control" placeholder="Название темы" form="editTopic${topic.id}"

name="name" value="${topic.name}">

<#else>

${topic.name}

</#if>

</td>

<td>

<a class="btn btn-primary" class="btn btn-secondary" data-toggle="collapse"

href="#pagesList${topic.id}" role="button" aria-expanded="false"

aria-controls="pagesList">

${topic.pages?size}

</a>

</td>

<td>

<form action="/constructor/course/${course.id}/topic/${topic.id}" method="get"

id="editTopic${topic.id}">

<div class="btn-group">

<#if topicEditId?? && topicEditId == topic.id>

<button type="submit" class="btn btn-secondary" name="action" value="save">Сохранить

</button>

<#else>

<button type="submit" class="btn btn-secondary" name="action" value="edit">Редактировать

</button>

</#if>

<button type="submit" class="btn btn-secondary" name="action" value="delete">Удалить

</button>

</div>

</form>

</td>

</tr>

<tr>

<td></td>

<td colspan="3">

<div class="collapse" id="pagesList${topic.id}">

<div class="table-responsive">

<table class="table table-sm">

<thead>

<tr>

<th scope="col">#</th>

<th scope="col">Название страницы</th>

<th scope="col">Действия</th>

</tr>

</thead>

<tbody>

<#list topic.pages as page>

<tr <#if !page.path??>class="table-warning"</#if>>

<th scope="row">${page.priority}</th>

<td>${page.name}</td>

<td>

<form action="/constructor/course/${course.id}/topic/${topic.id}/page/${page.id}"

method="get">

<div class="btn-group" role="group" aria-label="Basic example">

<button type="submit" class="btn btn-secondary" name="action"

value="edit">

Редактировать

</button>

<button type="submit" class="btn btn-secondary" name="action"

value="delete">Удалить

</button>

</div>

</form>

</td>

</tr>

<#else>

<tr>

<td>NO PAGES</td>

</tr>

</#list>

<tr>

<td colspan="3">

<form action="/constructor/course/${course.id}/topic/${topic.id}/page/create"

method="get">

<div class="form-row">

<div class="col">

<input type="number" class="form-control"

placeholder="Номер страницы"

name="priority"

value="${topic.lastPriorityPage + 1}">

</div>

<div class="col">

<input type="text" class="form-control"

placeholder="Название страницы"

name="name">

</div>

<div class="col">

<button type="submit" class="btn btn-primary">Добавить страницу

</button>

</div>

</div>

</form>

</td>

</tr>

</tbody>

</table>

</div>

</div>

</td>

</tr>

<#else>

<tr>

<td>NO TOPICS</td>

</tr>

</#list>

<tr>

<td colspan="4">

<form action="/constructor/course/${course.id}/topic/create" method="get">

<div class="form-row">

<div class="col">

<input type="number" class="form-control" placeholder="Номер темы" name="priority"

value="${course.lastPriorityPage + 1}">

</div>

<div class="col">

<input type="text" class="form-control" placeholder="Название темы" name="name">

</div>

<div class="col">

<button type="submit" class="btn btn-primary">Добавить тему</button>

</div>

</div>

</form>

</td>

</tr>

</tbody>

</table>

<div class="table-responsive">

</@c.page>

Текст программы Г.42 – Исходный код editPage.ftl

<#import "parts/common.ftl" as c>

<@c.page>

<h3>Редактирование страницы</h3>

<form action="/constructor/course/${course.id}/topic/${topic.id}/page/${page.id}/preview" method="post"

enctype="multipart/form-data">

<input type="hidden" name="${\_csrf.parameterName}" value="${\_csrf.token}">

<div class="form-group">

<label for="priority">Номер страницы</label>

<input type="number" class="form-control" id="priority" name="priority" value="${page.priority}">

</div>

<div class="form-group">

<label for="title">Заголовок страницы</label>

<input type="text" class="form-control" id="title" name="name" value="${page.name}">

</div>

<div class="form-group">

<label for="content">Содержимое страницы

<button type="button" class="badge badge-pill badge-info" data-toggle="modal" data-target="#contentHelp">

Получить подсказку

</button>

</label>

<div class="modal fade" id="contentHelp" tabindex="-1" role="dialog" aria-labelledby="exampleModalScrollableTitle" aria-hidden="true">

<div class="modal-dialog modal-dialog-scrollable modal-lg" role="document">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="exampleModalScrollableTitle">Рекомендации по добавлению содержимого страницы</h5>

<button type="button" class="close" data-dismiss="modal" aria-label="Close">

<span aria-hidden="true">&times;</span>

</button>

</div>

<div class="modal-body">

<div class="alert alert-danger mt-2" role="alert">

<h4 class="alert-heading">Структура содержимого</h4>

<p>Запрещено использовать следующую структуру содержимого страницы:</p>

<hr>

<pre class="html4strict" style="font-family:monospace;"><span style="color: #009900;">&lt;<span style="color: #000000; font-weight: bold;">html</span>&gt;</span>

<span style="color: #009900;">&lt;header&gt;&lt;<span style="color: #66cc66;">/</span>header&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #000000; font-weight: bold;">body</span>&gt;</span>

Содержимое...

<span style="color: #009900;">&lt;<span style="color: #66cc66;">/</span><span style="color: #000000; font-weight: bold;">body</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #66cc66;">/</span><span style="color: #000000; font-weight: bold;">html</span>&gt;</span></pre>

<hr>

<p>Так как все содержимое страницы будет добавлено в эту структуру.</p>

</div>

<div class="alert alert-warning mt-2" role="alert">

<h4 class="alert-heading">Подсветка синтаксиса</h4>

<p>Подсветку ситакциса вставляемого кода можно получить на <a href="http://highlight.hohli.com/index.php" class="alert-link">этом сайте</a></p>

<p>Полученную подсветку ситакциса кода советуеться добавить в блок для центрирования и прокрутки:</p>

<hr>

<pre class="html4strict" style="font-family:monospace;"><span style="color: #009900;">&lt;<span style="color: #000000; font-weight: bold;">div</span> <span style="color: #000066;">class</span><span style="color: #66cc66;">=</span><span style="color: #ff0000;">&quot;container&quot;</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #000000; font-weight: bold;">div</span> <span style="color: #000066;">class</span><span style="color: #66cc66;">=</span><span style="color: #ff0000;">&quot;row&quot;</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #000000; font-weight: bold;">div</span> <span style="color: #000066;">class</span><span style="color: #66cc66;">=</span><span style="color: #ff0000;">&quot;col&quot;</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #66cc66;">/</span><span style="color: #000000; font-weight: bold;">div</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #000000; font-weight: bold;">div</span> <span style="color: #000066;">class</span><span style="color: #66cc66;">=</span><span style="color: #ff0000;">&quot;col&quot;</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #000000; font-weight: bold;">div</span> <span style="color: #000066;">style</span><span style="color: #66cc66;">=</span><span style="color: #ff0000;">&quot;height:500px; width:800px; overflow:scroll;&quot;</span>&gt;</span>

Содержимое кода...

<span style="color: #009900;">&lt;<span style="color: #66cc66;">/</span><span style="color: #000000; font-weight: bold;">div</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #66cc66;">/</span><span style="color: #000000; font-weight: bold;">div</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #000000; font-weight: bold;">div</span> <span style="color: #000066;">class</span><span style="color: #66cc66;">=</span><span style="color: #ff0000;">&quot;col&quot;</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #66cc66;">/</span><span style="color: #000000; font-weight: bold;">div</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #66cc66;">/</span><span style="color: #000000; font-weight: bold;">div</span>&gt;</span>

<span style="color: #009900;">&lt;<span style="color: #66cc66;">/</span><span style="color: #000000; font-weight: bold;">div</span>&gt;</span></pre>

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-dismiss="modal">Закрыть подсказку</button>

</div>

</div>

</div>

</div>

<textarea class="form-control" id="content" aria-describedby="content-help" rows="13"

name="content"><#if page.path??>${content}</#if></textarea>

<small id="content-help" class="form-text text-muted">

Разрешено использовать HTML теги и элементы <a class="badge badge-info"

href="https://getbootstrap.com/docs/4.3/content/typography/">Bootstrap</a>

<a href="http://highlight.hohli.com/index.php" class="badge badge-info">Подсветка синтаксиса</a> на

странице

</small>

<hr>

<div class="form-group">

<label for="content-file">Или можно загрузить содержимое из файла (\*.txt, \*.html)</label>

<input type="file" class="form-control-file" id="content-file" aria-describedby="content-file-help"

accept=".txt,.html" name="contentFile">

<small id="content-file-help" class="form-text text-muted">

Если загружен файл, то содержимое страницы будет браться только из файла

</small>

</div>

<button type="submit" class="btn btn-primary">Предпросмотр</button>

</form>

</div>

</@c.page>

Текст программы Г.43 – Исходный код editProfile.ftl

<#import "parts/common.ftl" as c>

<@c.page>

<h3>Редактирование профиля</h3>

<form action="/user/${user.id}/edit" method="post">

<input type="hidden" name="\_csrf" value="${\_csrf.token}">

<div class="form-group row">

<label class="col-sm-2 col-form-label">Наименование профиля:</label>

<div class="col-sm-4">

<input class="form-control" type="text" name="username" placeholder="Username" value="${user.username}"/>

</div>

</div>

<div class="form-group row">

<label class="col-sm-2 col-form-label">Пароль:</label>

<div class="col-sm-4">

<input class="form-control" type="password" name="password" placeholder="Password"/>

</div>

</div>

<div class="form-group row">

<label class="col-sm-2 col-form-label">Email:</label>

<div class="col-sm-4">

<input class="form-control" type="email" name="email" placeholder="some@some.com"

value="${user.email!''}"/>

</div>

</div>

<button class="btn btn-primary mb-2" type="submit">Сохранить</button>

</form>

</@c.page>

Текст программы Г.44 – Исходный код index.ftl

<#import "parts/common.ftl" as c>

<#import "parts/coursesList.ftl" as cl>

<#include "parts/security.ftl">

<@c.page>

<h1>Добро пожаловать на учебную платформу!</h1>

<#if isTeacher>

<h3>Редактируемые вами курсы</h3>

<@cl.coursesList coursesEdit/>

</#if>

<h3>Активные курсы</h3>

<@cl.coursesList courses/>

</@c.page>

Текст программы Г.45 – Исходный код login.ftl

<#import "parts/common.ftl" as c>

<#import "parts/login.ftl" as l>

<@c.page>

<#if Session?? && Session.SPRING\_SECURITY\_LAST\_EXCEPTION??>

<div class="alert alert-danger" role="alert">

${Session.SPRING\_SECURITY\_LAST\_EXCEPTION.message}

</div>

</#if>

<#if message??>

<div class="alert alert-${messageType}" role="alert">

${message}

</div>

</#if>

<@l.login "/login" false/>

</@c.page>

Текст программы Г.46 – Исходный код openCourse.ftl

<#import "parts/commonCourse.ftl" as c>

<@c.page>

<div class="col">

<div class="tab-content" id="v-pills-tabContent">

<#list course.topics as topic>

<#list topic.pages as page>

<div class="tab-pane fade" id="v-pills-topic${topic.id}-page${page.id}" role="tabpanel"

aria-labelledby="v-pills-topic${topic.id}-tab">

<h3 class="mb-3">${page.name}</h3>

<hr>

${page.getContent(tools)}

</div>

</#list>

</#list>

</div>

</div>

</@c.page>

Текст программы Г.47 – Исходный код previewPage.ftl

<#import "parts/common.ftl" as c>

<@c.page>

<h3 class="mb-3">${name}</h3>

<hr>

${content}

<form action="/constructor/course/${course.id}/topic/${topic.id}/page/${page.id}/save" method="post">

<input type="hidden" name="${\_csrf.parameterName}" value="${\_csrf.token}">

<input type="hidden" name="name" value="${name}">

<input type="hidden" name="priority" value="${priority}">

<input type="hidden" name="contentBit" value="${contentBit}">

<button type="submit" class="btn btn-primary">Сохранить</button>

</form>

</@c.page>

Текст программы Г.48 – Исходный код profile.ftl

<#import "parts/common.ftl" as c>

<#include "parts/security.ftl">

<@c.page>

<h3>Профиль пользователя ${uuser.username}</h3>

<div class="card">

<div class="card-header">

${uuser.username}

</div>

<div class="card-body">

<ul class="list-group list-group-flush">

<li class="list-group-item">

<dl class="row">

<dt class="col-sm-3">Emil</dt>

<dd class="col-sm-9">${uuser.email}</dd>

</dl>

<#if uuser.id == currentUserId>

<div class="btn-group">

<a href="/user/${uuser.id}/edit" class="btn btn-secondary">Редактировать</a>

<a href="/user/${uuser.id}/delete" class="btn btn-secondary">Удалить</a>

</div>

</#if>

</li>

<#if uuser.isTeacher()>

<li class="list-group-item">

<h5 class="card-title">Модератор курсов</h5>

<#if editableCourses?size != 0>

<div class="table-responsive">

<table class="table table-sm">

<thead>

<tr>

<th scope="col">Название курса</th>

<th scope="col">Студенты</th>

<th scope="col">Модераторы</th>

<#if uuser.id == currentUserId>

<th scope="col">Действия</th>

</#if>

</tr>

</thead>

<tbody>

<#list editableCourses as course>

<tr>

<td>

<#if course.active??>

<a href="/course/${course.id}">${course.name}</a>

<#else>

${course.name}

</#if>

</td>

<td>

<a href="/course/${course.id}/students"

class="btn btn-primary">${course.students?size}</a>

</td>

<td><a href="/course/${course.id}/editors"

class="btn btn-primary">${course.editors?size}</a></td>

<#if uuser.id == currentUserId>

<td>

<#if course.editors?seq\_contains(uuser)>

<form action="/constructor/course/delete/${course.id}"

method="get">

<a class="btn btn-secondary"

href="/constructor/course/edit/${course.id}">Редактировать</a>

<button type="submit" class="btn btn-secondary">Удалить

</button>

</form>

</#if>

</td>

</#if>

</tr>

</#list>

</tbody>

</table>

</div>

<#else>

Нет модерируемых курс

</#if>

</li>

</#if>

<li class="list-group-item">

<h5 class="card-title">Подписан на курсы</h5>

<#if courses?size != 0>

<div class="table-responsive">

<table class="table table-sm">

<thead>

<tr>

<th scope="col">Название курса</th>

<th scope="col">Студенты</th>

<th scope="col">Модераторы</th>

<#if uuser.id == currentUserId>

<th scope="col">Действия</th>

</#if>

</tr>

</thead>

<tbody>

<#list courses as course>

<tr>

<td>

<#if course.active??>

<a href="/course/${course.id}">${course.name}</a>

<#else>

${course.name}

</#if>

</td>

<td>

<a href="/course/${course.id}/students"

class="btn btn-primary">${course.students?size}</a>

</td>

<td><a href="/course/${course.id}/editors"

class="btn btn-primary">${course.editors?size}</a></td>

<#if uuser.id == currentUserId>

<td>

<form action="/user/${uuser.id}/unsubscribe/${course.id}">

<button type="submit" class="btn btn-primary">Отписаться</button>

</form>

</td>

</#if>

</tr>

</#list>

</tbody>

</table>

</div>

<#else>

Не подписан ни на один курс

</#if>

</li>

<li class="list-group-item">

<h5 class="card-title">Успеваемость</h5>

<#if grades??>

<div class="table-responsive">

<table class="table table-sm">

<thead>

<tr>

<#list grades as grade>

<th scope="col">${grade.date}</th>

</#list>

</tr>

</thead>

<tbody>

<tr>

<#list grades as grade>

<th scope="col">${grade.grade}</th>

</#list>

</tr>

</tbody>

</table>

</div>

<#else>

Нет оценок

</#if>

</li>

</ul>

</div>

<div class="card-footer text-muted">

Статус аккаунта <#list roles as role>${role}<#sep>, </#list>

</div>

</div>

</@c.page>

Текст программы Г.49 – Исходный код registration.ftl

<#import "parts/common.ftl" as c>

<#import "parts/login.ftl" as l>

<@c.page>

<#if message??>

<div class="alert alert-danger" role="alert">

${message}

</div>

</#if>

<@l.login "/registration" true/>

</@c.page>

Текст программы Г.50 – Исходный код userList.ftl

<#import "parts/common.ftl" as c>

<#include "parts/security.ftl">

<@c.page>

<h3>Зарегестрированные пользователи</h3>

<div class="table-responsive">

<table class="table">

<thead class="thead-dark">

<tr>

<th scope="col">#</th>

<th scope="col">Наименование</th>

<th scope="col">Email</th>

<th scope="col">Роли</th>

<th scope="col">Подписки</th>

<th scope="col">Модерация</th>

<#if isAdmin>

<th scope="col">Действия</th>

</#if>

</tr>

</thead>

<tbody>

<#list userList as user>

<tr>

<th scope="row">${user.id}</th>

<td>

<a href="/user/${user.id}">${user.username}</a>

</td>

<td>${user.email}</td>

<td>

<#if userEditId?? && userEditId == user.id>

<#list roles as role>

<div class="form-check">

<label class="form-check-label">

<input class="form-check-input" type="checkbox" name="${role}"

form="userRole${user.id}" ${user.roles?seq\_contains(role)?string("checked", "")}>

${role}

</label>

</div>

</#list>

<#else>

<#list user.roles as role>${role}<#sep>, </#list>

</#if>

</td>

<td>

<a href="/user/${user.id}" class="btn btn-primary">${user.courses?size}</a>

</td>

<td>

<a href="/user/${user.id}" class="btn btn-primary">${user.editableCourses?size}</a>

</td>

<#if isAdmin>

<td>

<form action="/admin/user/${user.id}" method="post" id="userRole${user.id}">

<input type="hidden" name="\_csrf" value="${\_csrf.token}">

<div class="btn-group">

<#if userEditId?? && userEditId == user.id>

<button type="submit" class="btn btn-secondary" name="action" value="save">

Сохранить

</button>

<#else>

<button type="submit" class="btn btn-secondary" name="action" value="edit">

Редактировать

</button>

</#if>

<button type="submit" class="btn btn-secondary" name="action" value="delete">Удалить

</button>

</div>

</form>

</td>

</#if>

</tr>

</#list>

</tbody>

</table>

</div>

</@c.page>