Приложение А

(обязательное)

Функциональная модель процессов предметной области

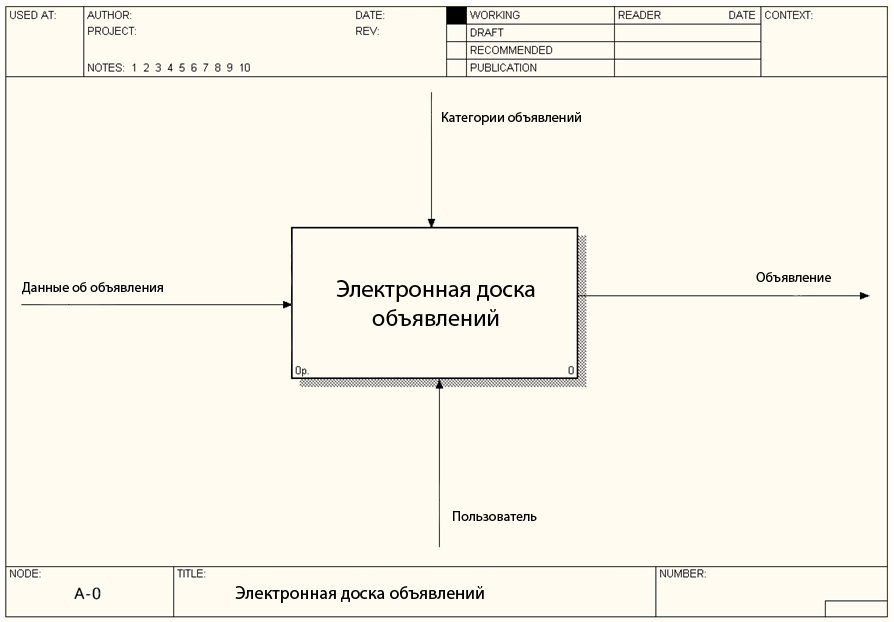


Рисунок А.1 – Основной процесс

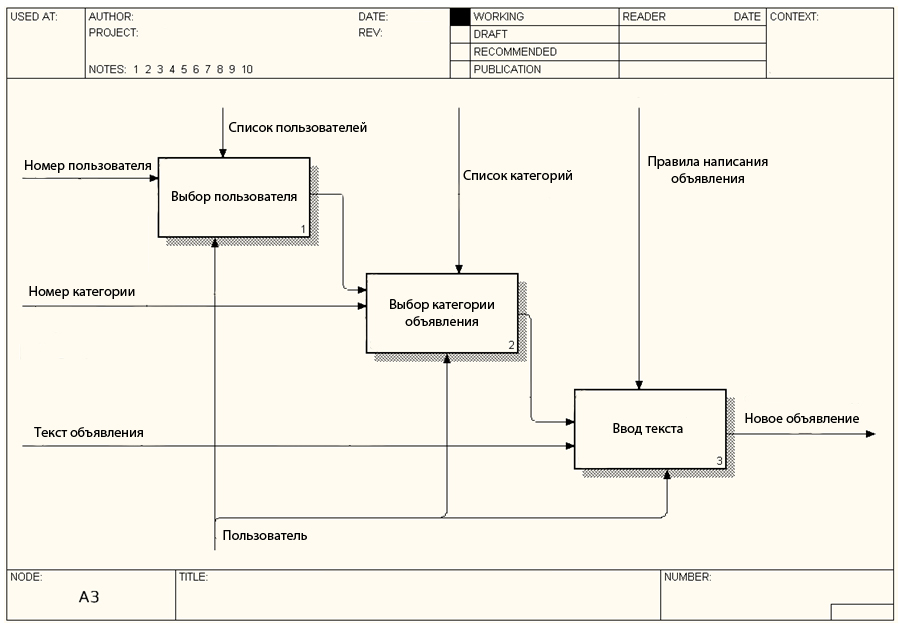


Рисунок А.2 – Декомпозиция основного процесса

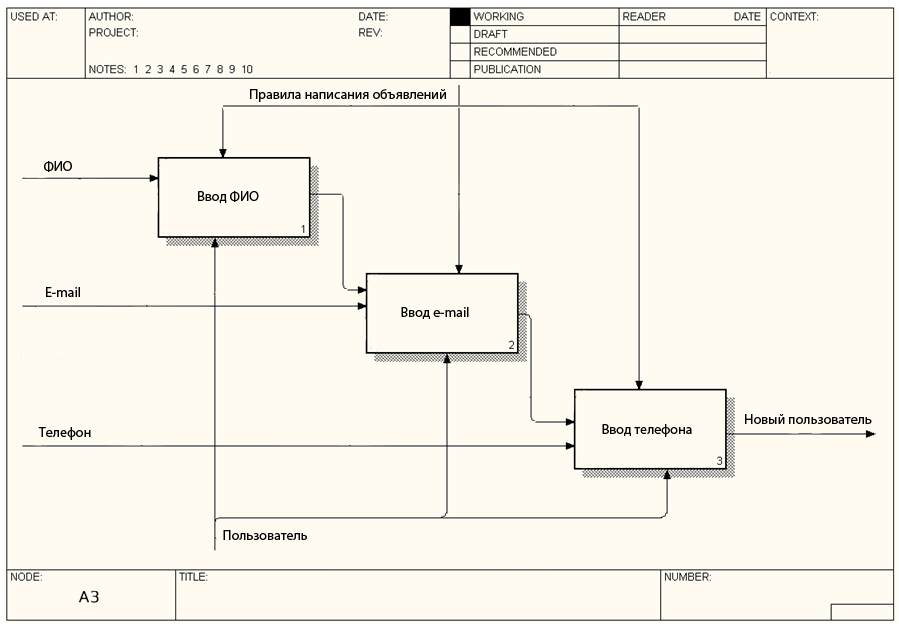
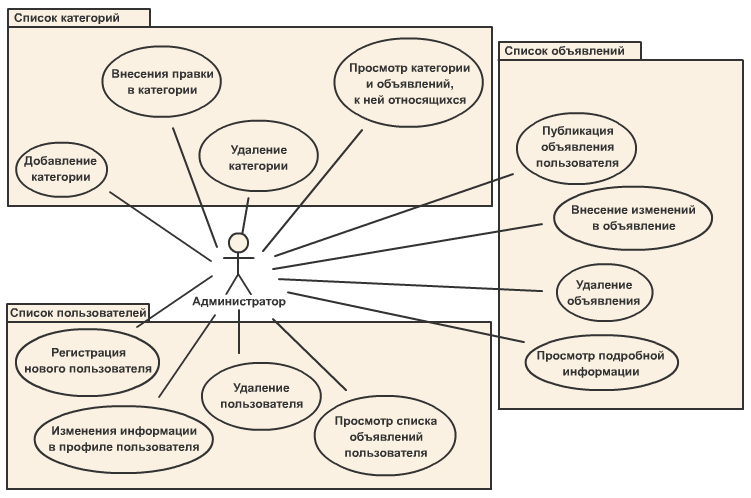


Рисунок А.3 – Декомпозиция второго уровня

Приложение Б

(обязательное)

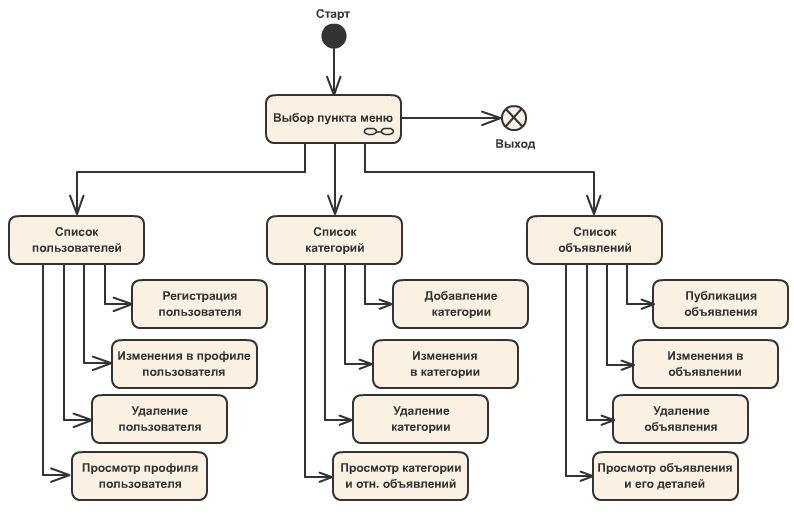
Диаграмма вариантов использования



Приложение В

(обязательное)

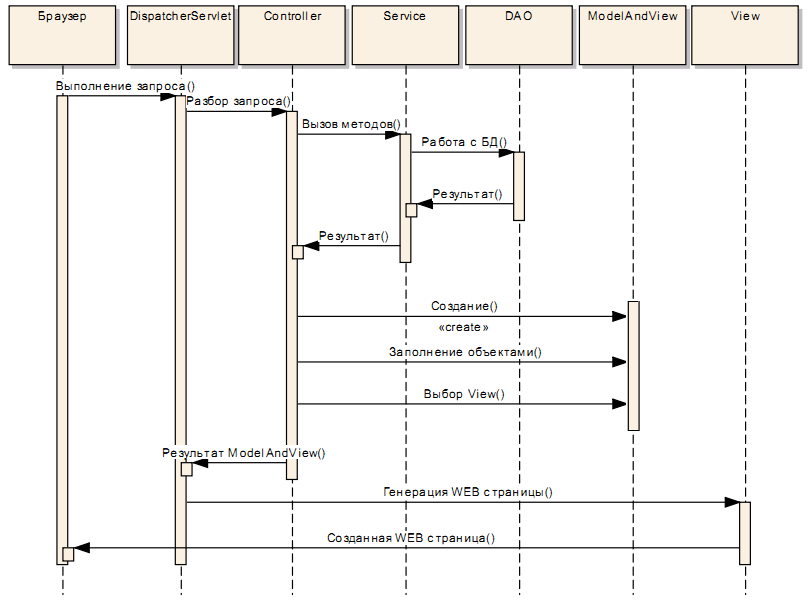
Диаграмма состояний клиентской части



Приложение Г

(обязательное)

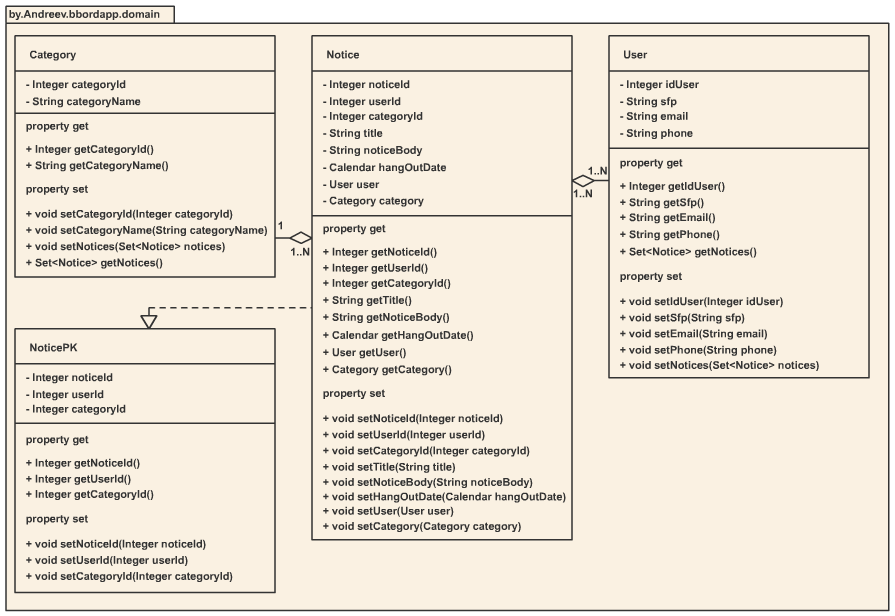
Диаграмма последовательностей

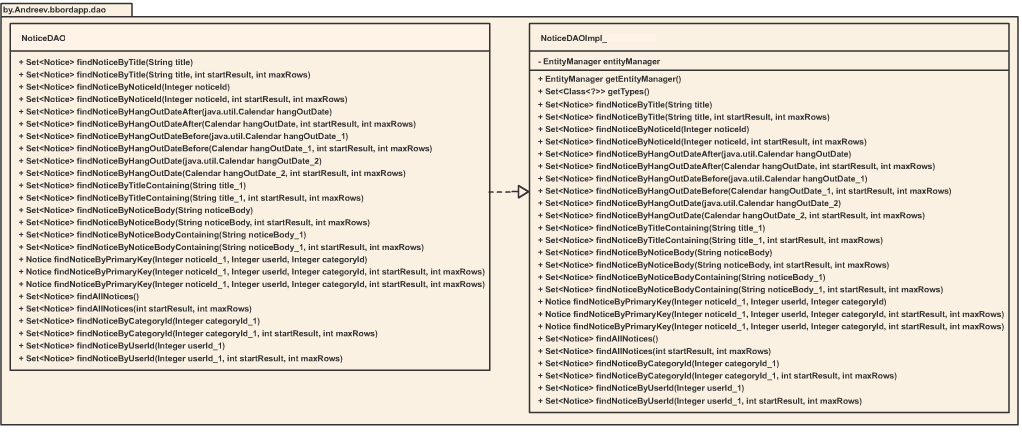


Приложение Д

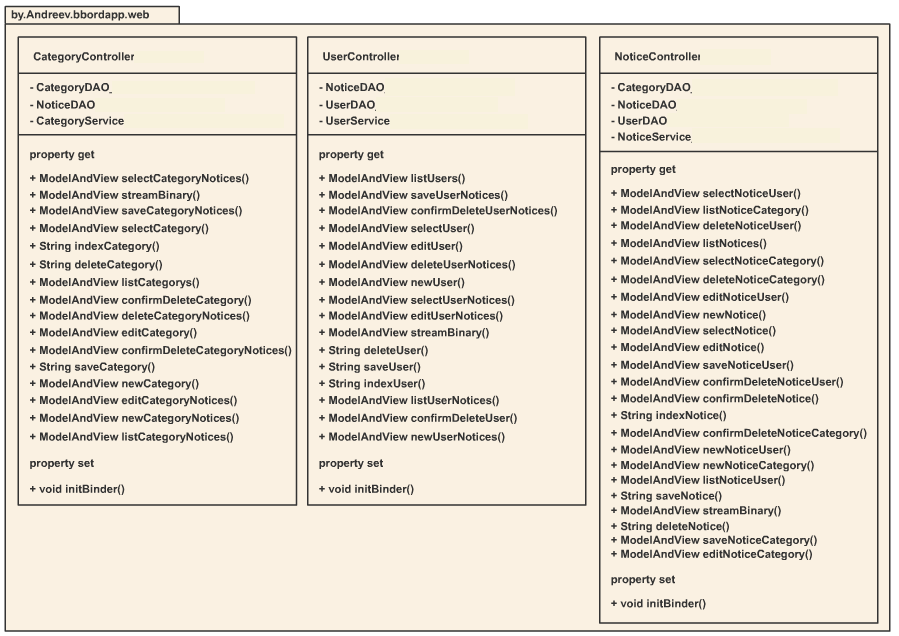
(обязательное)

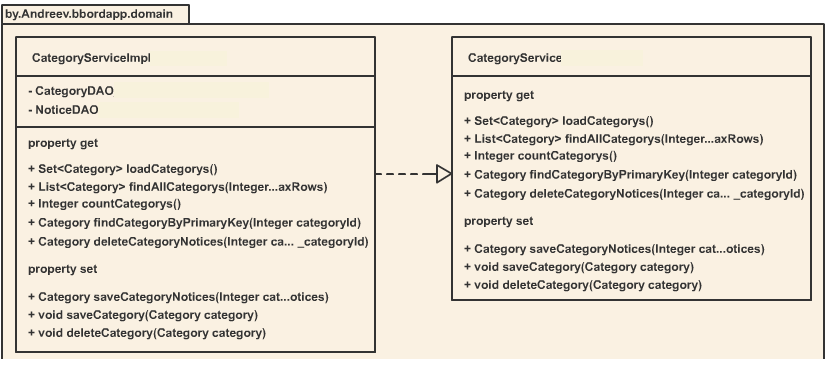
Диаграмма классов





продолжение диаграммы Д

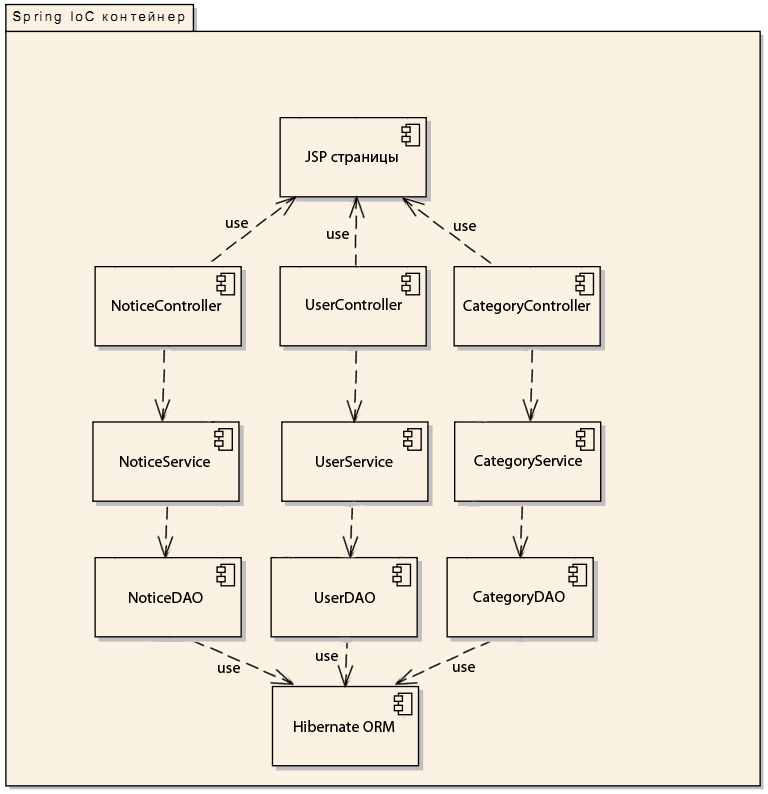




Приложение Е

(обязательное)

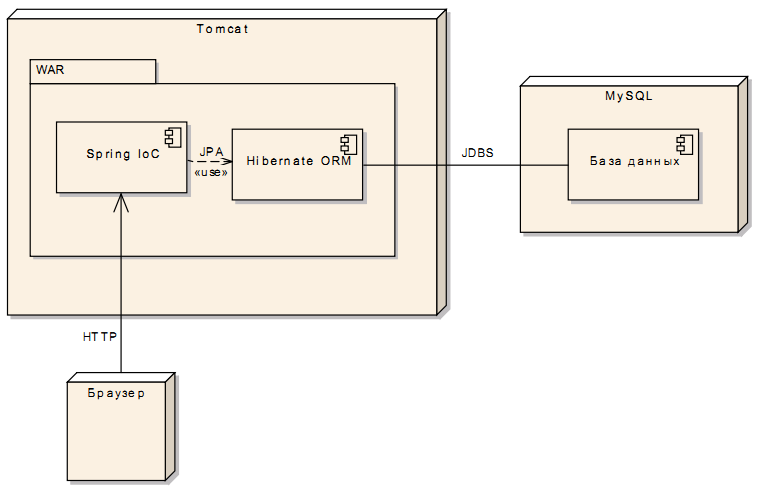
Диаграмма компонентов



Приложение Ж

(обязательное)

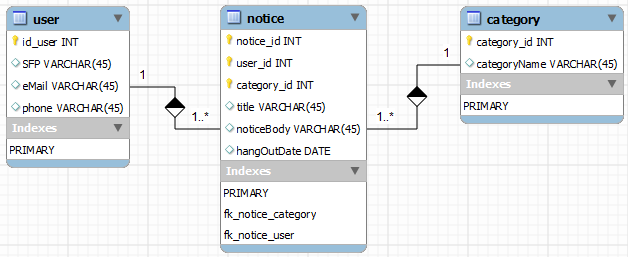
Диаграмма развертывания



Приложение З

(обязательное)

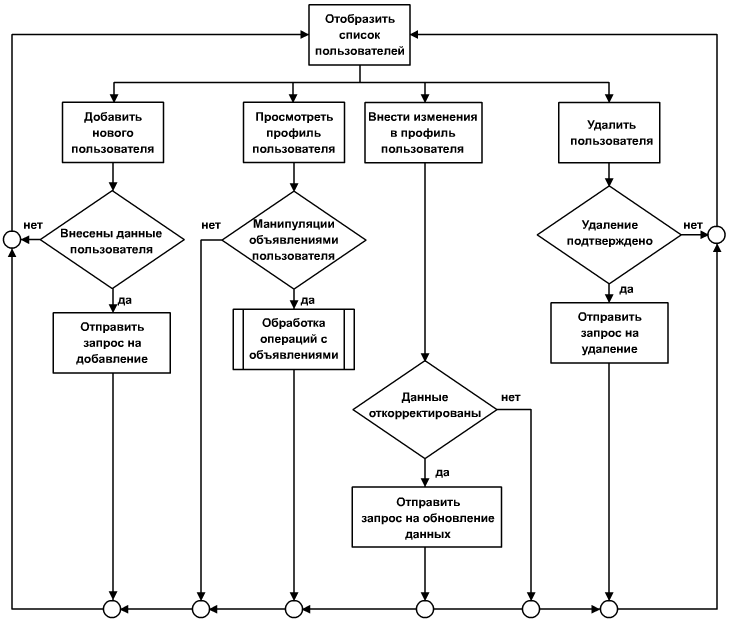
Информационная модель



Приложение И

(обязательное)

Блок-схемы алгоритмов, реализующих бизнес-логику



Приложение К

(обязательное)

Листинг алгоритмов реализующих бизнес-логику

**CategoryServiceImpl\_.java**

**package** by.bbordapp.service;

**import** by.bbordapp.dao.CategoryDAO\_;

**import** by.bbordapp.dao.NoticeDAO\_;

**import** by.bbordapp.domain.Category;

**import** by.bbordapp.domain.Notice;

**import** java.util.List;

**import** java.util.Set;

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

**import** org.springframework.stereotype.Service;

**import** org.springframework.transaction.annotation.Transactional;

/\*\*

\* Spring service that handles CRUD requests for Category entities

\* **@author** LaBestia

\*/

@Service("CategoryService")

@Transactional

**public** **class** CategoryServiceImpl\_ **implements** CategoryService\_ {

@Autowired

**private** CategoryDAO\_ categoryDAO;

@Autowired

**private** NoticeDAO\_ noticeDAO;

**public** CategoryServiceImpl\_() {

}

@Transactional

**public** Category saveCategoryNotices(Integer categoryId, Notice related\_notices) {

Category category = categoryDAO.findCategoryByPrimaryKey(categoryId, -1, -1);

Notice existingnotices = noticeDAO.findNoticeByPrimaryKey(related\_notices.getNoticeId(), related\_notices.getUserId(), related\_notices.getCategoryId());

// copy into the existing record to preserve existing relationships

**if** (existingnotices != **null**) {

existingnotices.setNoticeId(related\_notices.getNoticeId());

existingnotices.setUserId(related\_notices.getUserId());

existingnotices.setCategoryId(related\_notices.getCategoryId());

existingnotices.setTitle(related\_notices.getTitle());

existingnotices.setNoticeBody(related\_notices.getNoticeBody());

existingnotices.setHangOutDate(related\_notices.getHangOutDate());

related\_notices = existingnotices;

}

related\_notices.setCategory(category);

category.getNotices().add(related\_notices);

related\_notices = noticeDAO.store(related\_notices);

noticeDAO.flush();

category = categoryDAO.store(category);

categoryDAO.flush();

**return** category;

}

@Transactional

**public** Set<Category> loadCategorys() {

**return** categoryDAO.findAllCategorys();

}

@Transactional

**public** **void** saveCategory(Category category) {

Category existingCategory = categoryDAO.findCategoryByPrimaryKey(category.getCategoryId());

**if** (existingCategory != **null**) {

**if** (existingCategory != category) {

existingCategory.setCategoryId(category.getCategoryId());

existingCategory.setCategoryName(category.getCategoryName());

}

category = categoryDAO.store(existingCategory);

} **else** {

category = categoryDAO.store(category);

}

categoryDAO.flush();

}

@Transactional

**public** List<Category> findAllCategorys(Integer startResult, Integer maxRows) {

**return** **new** java.util.ArrayList<Category>(categoryDAO.findAllCategorys(startResult, maxRows));

}

@Transactional

**public** **void** deleteCategory(Category category) {

categoryDAO.remove(category);

categoryDAO.flush();

}

@Transactional

**public** Integer countCategorys() {

**return** ((Long) categoryDAO.createQuerySingleResult("select count(o) from Category o").getSingleResult()).intValue();

}

@Transactional

**public** Category findCategoryByPrimaryKey(Integer categoryId) {

**return** categoryDAO.findCategoryByPrimaryKey(categoryId);

}

@Transactional

**public** Category deleteCategoryNotices(Integer category\_categoryId, Integer related\_notices\_noticeId, Integer related\_notices\_userId, Integer related\_notices\_categoryId) {

Notice related\_notices = noticeDAO.findNoticeByPrimaryKey(related\_notices\_noticeId, related\_notices\_userId, related\_notices\_categoryId, -1, -1);

Category category = categoryDAO.findCategoryByPrimaryKey(category\_categoryId, -1, -1);

related\_notices.setCategory(**null**);

category.getNotices().remove(related\_notices);

noticeDAO.remove(related\_notices);

noticeDAO.flush();

**return** category;

}

}

**NoticeServiceImpl\_.java**

**package** by.bbordapp.service;

**import** by.bbordapp.dao.CategoryDAO\_;

**import** by.bbordapp.dao.NoticeDAO\_;

**import** by.bbordapp.dao.UserDAO\_;

**import** by.bbordapp.domain.Category;

**import** by.bbordapp.domain.Notice;

**import** by.bbordapp.domain.User;

**import** java.util.List;

**import** java.util.Set;

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

**import** org.springframework.stereotype.Service;

**import** org.springframework.transaction.annotation.Transactional;

/\*\*

\* Spring service that handles CRUD requests for Notice entities

\* **@author** LaBestia

\*/

@Service("NoticeService")

@Transactional

**public** **class** NoticeServiceImpl\_ **implements** NoticeService\_ {

@Autowired

**private** CategoryDAO\_ categoryDAO;

@Autowired

**private** NoticeDAO\_ noticeDAO;

@Autowired

**private** UserDAO\_ userDAO;

**public** NoticeServiceImpl\_() {

}

@Transactional

**public** Notice deleteNoticeCategory(Integer notice\_noticeId, Integer notice\_userId, Integer notice\_categoryId, Integer related\_category\_categoryId) {

Notice notice = noticeDAO.findNoticeByPrimaryKey(notice\_noticeId, notice\_userId, notice\_categoryId, -1, -1);

Category related\_category = categoryDAO.findCategoryByPrimaryKey(related\_category\_categoryId, -1, -1);

notice.setCategory(**null**);

related\_category.getNotices().remove(notice);

notice = noticeDAO.store(notice);

noticeDAO.flush();

related\_category = categoryDAO.store(related\_category);

categoryDAO.flush();

categoryDAO.remove(related\_category);

categoryDAO.flush();

**return** notice;

}

@Transactional

**public** Set<Notice> loadNotices() {

**return** noticeDAO.findAllNotices();

}

@Transactional

**public** **void** saveNotice(Notice notice) {

Notice existingNotice = noticeDAO.findNoticeByPrimaryKey(notice.getNoticeId(), notice.getUserId(), notice.getCategoryId());

**if** (existingNotice != **null**) {

**if** (existingNotice != notice) {

existingNotice.setNoticeId(notice.getNoticeId());

existingNotice.setUserId(notice.getUserId());

existingNotice.setCategoryId(notice.getCategoryId());

existingNotice.setTitle(notice.getTitle());

existingNotice.setNoticeBody(notice.getNoticeBody());

existingNotice.setHangOutDate(notice.getHangOutDate());

}

notice = noticeDAO.store(existingNotice);

} **else** {

notice = noticeDAO.store(notice);

}

noticeDAO.flush();

}

@Transactional

**public** **void** deleteNotice(Notice notice) {

noticeDAO.remove(notice);

noticeDAO.flush();

}

@Transactional

**public** Integer countNotices() {

**return** ((Long) noticeDAO.createQuerySingleResult("select count(\*) from Notice o").getSingleResult()).intValue();

}

@Transactional

**public** List<Notice> findAllNotices(Integer startResult, Integer maxRows) {

**return** **new** java.util.ArrayList<Notice>(noticeDAO.findAllNotices(startResult, maxRows));

}

@Transactional

**public** Notice deleteNoticeUser(Integer notice\_noticeId, Integer notice\_userId, Integer notice\_categoryId, Integer related\_user\_idUser) {

Notice notice = noticeDAO.findNoticeByPrimaryKey(notice\_noticeId, notice\_userId, notice\_categoryId, -1, -1);

User related\_user = userDAO.findUserByPrimaryKey(related\_user\_idUser, -1, -1);

notice.setUser(**null**);

related\_user.getNotices().remove(notice);

notice = noticeDAO.store(notice);

noticeDAO.flush();

related\_user = userDAO.store(related\_user);

userDAO.flush();

userDAO.remove(related\_user);

userDAO.flush();

**return** notice;

}

@Transactional

**public** Notice saveNoticeUser(Integer noticeId, Integer userId, Integer categoryId, User related\_user) {

Notice notice = noticeDAO.findNoticeByPrimaryKey(noticeId, userId, categoryId, -1, -1);

User existinguser = userDAO.findUserByPrimaryKey(related\_user.getIdUser());

// copy into the existing record to preserve existing relationships

**if** (existinguser != **null**) {

existinguser.setIdUser(related\_user.getIdUser());

existinguser.setSfp(related\_user.getSfp());

existinguser.setEmail(related\_user.getEmail());

existinguser.setPhone(related\_user.getPhone());

related\_user = existinguser;

}

notice.setUser(related\_user);

related\_user.getNotices().add(notice);

notice = noticeDAO.store(notice);

noticeDAO.flush();

related\_user = userDAO.store(related\_user);

userDAO.flush();

**return** notice;

}

@Transactional

**public** Notice saveNoticeCategory(Integer noticeId, Integer userId, Integer categoryId, Category related\_category) {

Notice notice = noticeDAO.findNoticeByPrimaryKey(noticeId, userId, categoryId, -1, -1);

Category existingcategory = categoryDAO.findCategoryByPrimaryKey(related\_category.getCategoryId());

// copy into the existing record to preserve existing relationships

**if** (existingcategory != **null**) {

existingcategory.setCategoryId(related\_category.getCategoryId());

existingcategory.setCategoryName(related\_category.getCategoryName());

related\_category = existingcategory;

}

notice.setCategory(related\_category);

related\_category.getNotices().add(notice);

notice = noticeDAO.store(notice);

noticeDAO.flush();

related\_category = categoryDAO.store(related\_category);

categoryDAO.flush();

**return** notice;

}

@Transactional

**public** Notice findNoticeByPrimaryKey(Integer noticeId, Integer userId, Integer categoryId) {

**return** noticeDAO.findNoticeByPrimaryKey(noticeId, userId, categoryId);

}

}

**UserServiceImpl\_.java**

**package** by.bbordapp.service;

**import** by.bbordapp.dao.NoticeDAO\_;

**import** by.bbordapp.dao.UserDAO\_;

**import** by.bbordapp.domain.Notice;

**import** by.bbordapp.domain.User;

**import** java.util.List;

**import** java.util.Set;

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

**import** org.springframework.stereotype.Service;

**import** org.springframework.transaction.annotation.Transactional;

/\*\*

\* Spring service that handles CRUD requests for User entities

\* **@author** LaBestia

\*/

@Service("UserService")

@Transactional

**public** **class** UserServiceImpl\_ **implements** UserService\_ {

@Autowired

**private** NoticeDAO\_ noticeDAO;

@Autowired

**private** UserDAO\_ userDAO;

**public** UserServiceImpl\_() {

}

@Transactional

**public** User findUserByPrimaryKey(Integer idUser) {

**return** userDAO.findUserByPrimaryKey(idUser);

}

@Transactional

**public** **void** saveUser(User user) {

User existingUser = userDAO.findUserByPrimaryKey(user.getIdUser());

**if** (existingUser != **null**) {

**if** (existingUser != user) {

existingUser.setIdUser(user.getIdUser());

existingUser.setSfp(user.getSfp());

existingUser.setEmail(user.getEmail());

existingUser.setPhone(user.getPhone());

}

user = userDAO.store(existingUser);

} **else** {

user = userDAO.store(user);

}

userDAO.flush();

}

@Transactional

**public** **void** deleteUser(User user) {

userDAO.remove(user);

userDAO.flush();

}

@Transactional

**public** List<User> findAllUsers(Integer startResult, Integer maxRows) {

**return** **new** java.util.ArrayList<User>(userDAO.findAllUsers(startResult, maxRows));

}

@Transactional

**public** Integer countUsers() {

**return** ((Long) userDAO.createQuerySingleResult("select count(o) from User o").getSingleResult()).intValue();

}

@Transactional

**public** User saveUserNotices(Integer idUser, Notice related\_notices) {

User user = userDAO.findUserByPrimaryKey(idUser, -1, -1);

Notice existingnotices = noticeDAO.findNoticeByPrimaryKey(related\_notices.getNoticeId(), related\_notices.getUserId(), related\_notices.getCategoryId());

// copy into the existing record to preserve existing relationships

**if** (existingnotices != **null**) {

existingnotices.setNoticeId(related\_notices.getNoticeId());

existingnotices.setUserId(related\_notices.getUserId());

existingnotices.setCategoryId(related\_notices.getCategoryId());

existingnotices.setTitle(related\_notices.getTitle());

existingnotices.setNoticeBody(related\_notices.getNoticeBody());

existingnotices.setHangOutDate(related\_notices.getHangOutDate());

related\_notices = existingnotices;

}

related\_notices.setUser(user);

user.getNotices().add(related\_notices);

related\_notices = noticeDAO.store(related\_notices);

noticeDAO.flush();

user = userDAO.store(user);

userDAO.flush();

**return** user;

}

@Transactional

**public** User deleteUserNotices(Integer user\_idUser, Integer related\_notices\_noticeId, Integer related\_notices\_userId, Integer related\_notices\_categoryId) {

Notice related\_notices = noticeDAO.findNoticeByPrimaryKey(related\_notices\_noticeId, related\_notices\_userId, related\_notices\_categoryId, -1, -1);

User user = userDAO.findUserByPrimaryKey(user\_idUser, -1, -1);

related\_notices.setUser(**null**);

user.getNotices().remove(related\_notices);

noticeDAO.remove(related\_notices);

noticeDAO.flush();

**return** user;

}

@Transactional

**public** Set<User> loadUsers() {

**return** userDAO.findAllUsers();

}

}

Приложение Л

(обязательное)

Листинг основных элементов программы

**CategoryController\_.java**

**package** by.bbordapp.web;

**import** by.bbordapp.dao.CategoryDAO\_;

**import** by.bbordapp.dao.NoticeDAO\_;

**import** by.bbordapp.domain.Category;

**import** by.bbordapp.domain.Notice;

**import** by.bbordapp.service.CategoryService\_;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

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

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.WebDataBinder;

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

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

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

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

**import** org.springframework.web.servlet.ModelAndView;

/\*\*

\* Spring MVC controller that handles CRUD requests for Category entities

\* **@author** LaBestia

\*/

@Controller("CategoryController")

**public** **class** CategoryController\_ {

@Autowired

**private** CategoryDAO\_ categoryDAO;

@Autowired

**private** NoticeDAO\_ noticeDAO;

@Autowired

**private** CategoryService\_ categoryService;

@InitBinder

**public** **void** initBinder(WebDataBinder binder, HttpServletRequest request) { // Register static property editors.

binder.registerCustomEditor(java.util.Calendar.**class**, **new** org.skyway.spring.util.databinding.CustomCalendarEditor());

binder.registerCustomEditor(**byte**[].**class**, **new** org.springframework.web.multipart.support.ByteArrayMultipartFileEditor());

binder.registerCustomEditor(**boolean**.**class**, **new** org.skyway.spring.util.databinding.EnhancedBooleanEditor(**false**));

binder.registerCustomEditor(Boolean.**class**, **new** org.skyway.spring.util.databinding.EnhancedBooleanEditor(**true**));

binder.registerCustomEditor(java.math.BigDecimal.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(java.math.BigDecimal.**class**, **true**));

binder.registerCustomEditor(Integer.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(Integer.**class**, **true**));

binder.registerCustomEditor(java.util.Date.**class**, **new** org.skyway.spring.util.databinding.CustomDateEditor());

binder.registerCustomEditor(String.**class**, **new** org.skyway.spring.util.databinding.StringEditor());

binder.registerCustomEditor(Long.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(Long.**class**, **true**));

binder.registerCustomEditor(Double.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(Double.**class**, **true**));

}

@RequestMapping("/selectCategoryNotices")

**public** ModelAndView selectCategoryNotices(@RequestParam Integer category\_categoryId, @RequestParam Integer notices\_noticeId, @RequestParam Integer notices\_userId, @RequestParam Integer notices\_categoryId) {

Notice notice = noticeDAO.findNoticeByPrimaryKey(notices\_noticeId, notices\_userId, notices\_categoryId, -1, -1);

ModelAndView mav = **new** ModelAndView();

mav.addObject("category\_categoryId", category\_categoryId);

mav.addObject("notice", notice);

mav.setViewName("category/notices/viewNotices.jsp");

**return** mav;

}

@RequestMapping("/categoryController/binary.action")

**public** ModelAndView streamBinary(@ModelAttribute HttpServletRequest request, @ModelAttribute HttpServletResponse response) {

ModelAndView mav = **new** ModelAndView();

mav.setViewName("streamedBinaryContentView");

**return** mav;

}

@RequestMapping("/saveCategoryNotices")

**public** ModelAndView saveCategoryNotices(@RequestParam Integer category\_categoryId, @ModelAttribute Notice notices) {

Category parent\_category = categoryService.saveCategoryNotices(category\_categoryId, notices);

ModelAndView mav = **new** ModelAndView();

mav.addObject("category\_categoryId", category\_categoryId);

mav.addObject("category", parent\_category);

mav.setViewName("category/viewCategory.jsp");

**return** mav;

}

@RequestMapping("/selectCategory")

**public** ModelAndView selectCategory(@RequestParam Integer categoryIdKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("category", categoryDAO.findCategoryByPrimaryKey(categoryIdKey));

mav.setViewName("category/viewCategory.jsp");

**return** mav;

}

**public** String indexCategory() {

**return** "redirect:/indexCategory";

}

@RequestMapping("/deleteCategory")

**public** String deleteCategory(@RequestParam Integer categoryIdKey) {

Category category = categoryDAO.findCategoryByPrimaryKey(categoryIdKey);

categoryService.deleteCategory(category);

**return** "forward:/indexCategory";

}

@RequestMapping("/indexCategory")

**public** ModelAndView listCategorys() {

ModelAndView mav = **new** ModelAndView();

mav.addObject("categorys", categoryService.loadCategorys());

mav.setViewName("category/listCategorys.jsp");

**return** mav;

}

@RequestMapping("/confirmDeleteCategory")

**public** ModelAndView confirmDeleteCategory(@RequestParam Integer categoryIdKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("category", categoryDAO.findCategoryByPrimaryKey(categoryIdKey));

mav.setViewName("category/deleteCategory.jsp");

**return** mav;

}

@RequestMapping("/deleteCategoryNotices")

**public** ModelAndView deleteCategoryNotices(@RequestParam Integer category\_categoryId, @RequestParam Integer related\_notices\_noticeId, @RequestParam Integer related\_notices\_userId, @RequestParam Integer related\_notices\_categoryId) {

ModelAndView mav = **new** ModelAndView();

Category category = categoryService.deleteCategoryNotices(category\_categoryId, related\_notices\_noticeId, related\_notices\_userId, related\_notices\_categoryId);

mav.addObject("category\_categoryId", category\_categoryId);

mav.addObject("category", category);

mav.setViewName("category/viewCategory.jsp");

**return** mav;

}

@RequestMapping("/editCategory")

**public** ModelAndView editCategory(@RequestParam Integer categoryIdKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("category", categoryDAO.findCategoryByPrimaryKey(categoryIdKey));

mav.setViewName("category/editCategory.jsp");

**return** mav;

}

@RequestMapping("/confirmDeleteCategoryNotices")

**public** ModelAndView confirmDeleteCategoryNotices(@RequestParam Integer category\_categoryId, @RequestParam Integer related\_notices\_noticeId, @RequestParam Integer related\_notices\_userId, @RequestParam Integer related\_notices\_categoryId) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice", noticeDAO.findNoticeByPrimaryKey(related\_notices\_noticeId, related\_notices\_userId, related\_notices\_categoryId));

mav.addObject("category\_categoryId", category\_categoryId);

mav.setViewName("category/notices/deleteNotices.jsp");

**return** mav;

}

@RequestMapping("/saveCategory")

**public** String saveCategory(@ModelAttribute Category category) {

categoryService.saveCategory(category);

**return** "forward:/indexCategory";

}

@RequestMapping("/newCategory")

**public** ModelAndView newCategory() {

ModelAndView mav = **new** ModelAndView();

mav.addObject("category", **new** Category());

mav.addObject("newFlag", **true**);

mav.setViewName("category/editCategory.jsp");

**return** mav;

}

@RequestMapping("/editCategoryNotices")

**public** ModelAndView editCategoryNotices(@RequestParam Integer category\_categoryId, @RequestParam Integer notices\_noticeId, @RequestParam Integer notices\_userId, @RequestParam Integer notices\_categoryId) {

Notice notice = noticeDAO.findNoticeByPrimaryKey(notices\_noticeId, notices\_userId, notices\_categoryId, -1, -1);

ModelAndView mav = **new** ModelAndView();

mav.addObject("category\_categoryId", category\_categoryId);

mav.addObject("notice", notice);

mav.setViewName("category/notices/editNotices.jsp");

**return** mav;

}

@RequestMapping("/newCategoryNotices")

**public** ModelAndView newCategoryNotices(@RequestParam Integer category\_categoryId) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("category\_categoryId", category\_categoryId);

mav.addObject("notice", **new** Notice());

mav.addObject("newFlag", **true**);

mav.setViewName("category/notices/editNotices.jsp");

**return** mav;

}

@RequestMapping("/listCategoryNotices")

**public** ModelAndView listCategoryNotices(@RequestParam Integer categoryIdKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("category", categoryDAO.findCategoryByPrimaryKey(categoryIdKey));

mav.setViewName("category/notices/listNotices.jsp");

**return** mav;

}

}

**NoticeController\_.java**

**package** by.bbordapp.web;

**import** by.bbordapp.dao.CategoryDAO\_;

**import** by.bbordapp.dao.NoticeDAO\_;

**import** by.bbordapp.dao.UserDAO\_;

**import** by.bbordapp.domain.Category;

**import** by.bbordapp.domain.Notice;

**import** by.bbordapp.domain.User;

**import** by.bbordapp.service.NoticeService\_;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

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

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.WebDataBinder;

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

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

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

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

**import** org.springframework.web.servlet.ModelAndView;

/\*\*

\* Spring MVC controller that handles CRUD requests for Notice entities

\* **@author** LaBestia

\*/

@Controller("NoticeController")

**public** **class** NoticeController\_ {

@Autowired

**private** CategoryDAO\_ categoryDAO;

@Autowired

**private** NoticeDAO\_ noticeDAO;

@Autowired

**private** UserDAO\_ userDAO;

@Autowired

**private** NoticeService\_ noticeService;

@RequestMapping("/selectNoticeUser")

**public** ModelAndView selectNoticeUser(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @RequestParam Integer user\_idUser) {

User user = userDAO.findUserByPrimaryKey(user\_idUser, -1, -1);

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("user", user);

mav.setViewName("notice/user/viewUser.jsp");

**return** mav;

}

@RequestMapping("/listNoticeCategory")

**public** ModelAndView listNoticeCategory(@RequestParam Integer noticeIdKey, @RequestParam Integer userIdKey, @RequestParam Integer categoryIdKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice", noticeDAO.findNoticeByPrimaryKey(noticeIdKey, userIdKey, categoryIdKey));

mav.setViewName("notice/category/listCategory.jsp");

**return** mav;

}

@RequestMapping("/deleteNoticeUser")

**public** ModelAndView deleteNoticeUser(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @RequestParam Integer related\_user\_idUser) {

ModelAndView mav = **new** ModelAndView();

Notice notice = noticeService.deleteNoticeUser(notice\_noticeId, notice\_userId, notice\_categoryId, related\_user\_idUser);

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("notice", notice);

mav.setViewName("notice/viewNotice.jsp");

**return** mav;

}

@RequestMapping("/indexNotice")

**public** ModelAndView listNotices() {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notices", noticeService.loadNotices());

mav.setViewName("notice/listNotices.jsp");

**return** mav;

}

@InitBinder

**public** **void** initBinder(WebDataBinder binder, HttpServletRequest request) { // Register static property editors.

binder.registerCustomEditor(java.util.Calendar.**class**, **new** org.skyway.spring.util.databinding.CustomCalendarEditor());

binder.registerCustomEditor(**byte**[].**class**, **new** org.springframework.web.multipart.support.ByteArrayMultipartFileEditor());

binder.registerCustomEditor(**boolean**.**class**, **new** org.skyway.spring.util.databinding.EnhancedBooleanEditor(**false**));

binder.registerCustomEditor(Boolean.**class**, **new** org.skyway.spring.util.databinding.EnhancedBooleanEditor(**true**));

binder.registerCustomEditor(java.math.BigDecimal.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(java.math.BigDecimal.**class**, **true**));

binder.registerCustomEditor(Integer.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(Integer.**class**, **true**));

binder.registerCustomEditor(java.util.Date.**class**, **new** org.skyway.spring.util.databinding.CustomDateEditor());

binder.registerCustomEditor(String.**class**, **new** org.skyway.spring.util.databinding.StringEditor());

binder.registerCustomEditor(Long.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(Long.**class**, **true**));

binder.registerCustomEditor(Double.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(Double.**class**, **true**));

}

@RequestMapping("/selectNoticeCategory")

**public** ModelAndView selectNoticeCategory(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @RequestParam Integer category\_categoryId) {

Category category = categoryDAO.findCategoryByPrimaryKey(category\_categoryId, -1, -1);

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("category", category);

mav.setViewName("notice/category/viewCategory.jsp");

**return** mav;

}

@RequestMapping("/deleteNoticeCategory")

**public** ModelAndView deleteNoticeCategory(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @RequestParam Integer related\_category\_categoryId) {

ModelAndView mav = **new** ModelAndView();

Notice notice = noticeService.deleteNoticeCategory(notice\_noticeId, notice\_userId, notice\_categoryId, related\_category\_categoryId);

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("notice", notice);

mav.setViewName("notice/viewNotice.jsp");

**return** mav;

}

@RequestMapping("/editNoticeUser")

**public** ModelAndView editNoticeUser(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @RequestParam Integer user\_idUser) {

User user = userDAO.findUserByPrimaryKey(user\_idUser, -1, -1);

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("user", user);

mav.setViewName("notice/user/editUser.jsp");

**return** mav;

}

@RequestMapping("/newNotice")

**public** ModelAndView newNotice() {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice", **new** Notice());

mav.addObject("newFlag", **true**);

mav.setViewName("notice/editNotice.jsp");

**return** mav;

}

@RequestMapping("/selectNotice")

**public** ModelAndView selectNotice(@RequestParam Integer noticeIdKey, @RequestParam Integer userIdKey, @RequestParam Integer categoryIdKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice", noticeDAO.findNoticeByPrimaryKey(noticeIdKey, userIdKey, categoryIdKey));

mav.setViewName("notice/viewNotice.jsp");

**return** mav;

}

@RequestMapping("/editNotice")

**public** ModelAndView editNotice(@RequestParam Integer noticeIdKey, @RequestParam Integer userIdKey, @RequestParam Integer categoryIdKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice", noticeDAO.findNoticeByPrimaryKey(noticeIdKey, userIdKey, categoryIdKey));

mav.setViewName("notice/editNotice.jsp");

**return** mav;

}

@RequestMapping("/saveNoticeUser")

**public** ModelAndView saveNoticeUser(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @ModelAttribute User user) {

Notice parent\_notice = noticeService.saveNoticeUser(notice\_noticeId, notice\_userId, notice\_categoryId, user);

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("notice", parent\_notice);

mav.setViewName("notice/viewNotice.jsp");

**return** mav;

}

@RequestMapping("/confirmDeleteNoticeUser")

**public** ModelAndView confirmDeleteNoticeUser(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @RequestParam Integer related\_user\_idUser) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("user", userDAO.findUserByPrimaryKey(related\_user\_idUser));

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.setViewName("notice/user/deleteUser.jsp");

**return** mav;

}

@RequestMapping("/confirmDeleteNotice")

**public** ModelAndView confirmDeleteNotice(@RequestParam Integer noticeIdKey, @RequestParam Integer userIdKey, @RequestParam Integer categoryIdKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice", noticeDAO.findNoticeByPrimaryKey(noticeIdKey, userIdKey, categoryIdKey));

mav.setViewName("notice/deleteNotice.jsp");

**return** mav;

}

**public** String indexNotice() {

**return** "redirect:/indexNotice";

}

@RequestMapping("/confirmDeleteNoticeCategory")

**public** ModelAndView confirmDeleteNoticeCategory(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @RequestParam Integer related\_category\_categoryId) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("category", categoryDAO.findCategoryByPrimaryKey(related\_category\_categoryId));

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.setViewName("notice/category/deleteCategory.jsp");

**return** mav;

}

@RequestMapping("/newNoticeUser")

**public** ModelAndView newNoticeUser(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("user", **new** User());

mav.addObject("newFlag", **true**);

mav.setViewName("notice/user/editUser.jsp");

**return** mav;

}

@RequestMapping("/newNoticeCategory")

**public** ModelAndView newNoticeCategory(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("category", **new** Category());

mav.addObject("newFlag", **true**);

mav.setViewName("notice/category/editCategory.jsp");

**return** mav;

}

@RequestMapping("/listNoticeUser")

**public** ModelAndView listNoticeUser(@RequestParam Integer noticeIdKey, @RequestParam Integer userIdKey, @RequestParam Integer categoryIdKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice", noticeDAO.findNoticeByPrimaryKey(noticeIdKey, userIdKey, categoryIdKey));

mav.setViewName("notice/user/listUser.jsp");

**return** mav;

}

@RequestMapping("/saveNotice")

**public** String saveNotice(@ModelAttribute Notice notice) {

noticeService.saveNotice(notice);

**return** "forward:/indexNotice";

}

@RequestMapping("/noticeController/binary.action")

**public** ModelAndView streamBinary(@ModelAttribute HttpServletRequest request, @ModelAttribute HttpServletResponse response) {

ModelAndView mav = **new** ModelAndView();

mav.setViewName("streamedBinaryContentView");

**return** mav;

}

@RequestMapping("/deleteNotice")

**public** String deleteNotice(@RequestParam Integer noticeIdKey, @RequestParam Integer userIdKey, @RequestParam Integer categoryIdKey) {

Notice notice = noticeDAO.findNoticeByPrimaryKey(noticeIdKey, userIdKey, categoryIdKey);

noticeService.deleteNotice(notice);

**return** "forward:/indexNotice";

}

@RequestMapping("/saveNoticeCategory")

**public** ModelAndView saveNoticeCategory(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @ModelAttribute Category category) {

Notice parent\_notice = noticeService.saveNoticeCategory(notice\_noticeId, notice\_userId, notice\_categoryId, category);

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("notice", parent\_notice);

mav.setViewName("notice/viewNotice.jsp");

**return** mav;

}

@RequestMapping("/editNoticeCategory")

**public** ModelAndView editNoticeCategory(@RequestParam Integer notice\_noticeId, @RequestParam Integer notice\_userId, @RequestParam Integer notice\_categoryId, @RequestParam Integer category\_categoryId) {

Category category = categoryDAO.findCategoryByPrimaryKey(category\_categoryId, -1, -1);

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice\_noticeId", notice\_noticeId);

mav.addObject("notice\_userId", notice\_userId);

mav.addObject("notice\_categoryId", notice\_categoryId);

mav.addObject("category", category);

mav.setViewName("notice/category/editCategory.jsp");

**return** mav;

}

}

**UserController\_.java**

**package** by.bbordapp.web;

**import** by.bbordapp.dao.NoticeDAO\_;

**import** by.bbordapp.dao.UserDAO\_;

**import** by.bbordapp.domain.Notice;

**import** by.bbordapp.domain.User;

**import** by.bbordapp.service.UserService\_;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

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

**import** org.springframework.stereotype.Controller;

**import** org.springframework.web.bind.WebDataBinder;

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

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

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

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

**import** org.springframework.web.servlet.ModelAndView;

/\*\*

\* Spring MVC controller that handles CRUD requests for User entities

\* **@author** LaBestia

\*/

@Controller("UserController")

**public** **class** UserController\_ {

@Autowired

**private** NoticeDAO\_ noticeDAO;

@Autowired

**private** UserDAO\_ userDAO;

@Autowired

**private** UserService\_ userService;

@RequestMapping("/indexUser")

**public** ModelAndView listUsers() {

ModelAndView mav = **new** ModelAndView();

mav.addObject("users", userService.loadUsers());

mav.setViewName("user/listUsers.jsp");

**return** mav;

}

@RequestMapping("/saveUserNotices")

**public** ModelAndView saveUserNotices(@RequestParam Integer user\_idUser, @ModelAttribute Notice notices) {

User parent\_user = userService.saveUserNotices(user\_idUser, notices);

ModelAndView mav = **new** ModelAndView();

mav.addObject("user\_idUser", user\_idUser);

mav.addObject("user", parent\_user);

mav.setViewName("user/viewUser.jsp");

**return** mav;

}

@RequestMapping("/confirmDeleteUserNotices")

**public** ModelAndView confirmDeleteUserNotices(@RequestParam Integer user\_idUser, @RequestParam Integer related\_notices\_noticeId, @RequestParam Integer related\_notices\_userId, @RequestParam Integer related\_notices\_categoryId) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("notice", noticeDAO.findNoticeByPrimaryKey(related\_notices\_noticeId, related\_notices\_userId, related\_notices\_categoryId));

mav.addObject("user\_idUser", user\_idUser);

mav.setViewName("user/notices/deleteNotices.jsp");

**return** mav;

}

@InitBinder

**public** **void** initBinder(WebDataBinder binder, HttpServletRequest request) { // Register static property editors.

binder.registerCustomEditor(java.util.Calendar.**class**, **new** org.skyway.spring.util.databinding.CustomCalendarEditor());

binder.registerCustomEditor(**byte**[].**class**, **new** org.springframework.web.multipart.support.ByteArrayMultipartFileEditor());

binder.registerCustomEditor(**boolean**.**class**, **new** org.skyway.spring.util.databinding.EnhancedBooleanEditor(**false**));

binder.registerCustomEditor(Boolean.**class**, **new** org.skyway.spring.util.databinding.EnhancedBooleanEditor(**true**));

binder.registerCustomEditor(java.math.BigDecimal.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(java.math.BigDecimal.**class**, **true**));

binder.registerCustomEditor(Integer.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(Integer.**class**, **true**));

binder.registerCustomEditor(java.util.Date.**class**, **new** org.skyway.spring.util.databinding.CustomDateEditor());

binder.registerCustomEditor(String.**class**, **new** org.skyway.spring.util.databinding.StringEditor());

binder.registerCustomEditor(Long.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(Long.**class**, **true**));

binder.registerCustomEditor(Double.**class**, **new** org.skyway.spring.util.databinding.NaNHandlingNumberEditor(Double.**class**, **true**));

}

@RequestMapping("/selectUser")

**public** ModelAndView selectUser(@RequestParam Integer idUserKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("user", userDAO.findUserByPrimaryKey(idUserKey));

mav.setViewName("user/viewUser.jsp");

**return** mav;

}

@RequestMapping("/editUser")

**public** ModelAndView editUser(@RequestParam Integer idUserKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("user", userDAO.findUserByPrimaryKey(idUserKey));

mav.setViewName("user/editUser.jsp");

**return** mav;

}

@RequestMapping("/deleteUserNotices")

**public** ModelAndView deleteUserNotices(@RequestParam Integer user\_idUser, @RequestParam Integer related\_notices\_noticeId, @RequestParam Integer related\_notices\_userId, @RequestParam Integer related\_notices\_categoryId) {

ModelAndView mav = **new** ModelAndView();

User user = userService.deleteUserNotices(user\_idUser, related\_notices\_noticeId, related\_notices\_userId, related\_notices\_categoryId);

mav.addObject("user\_idUser", user\_idUser);

mav.addObject("user", user);

mav.setViewName("user/viewUser.jsp");

**return** mav;

}

@RequestMapping("/newUser")

**public** ModelAndView newUser() {

ModelAndView mav = **new** ModelAndView();

mav.addObject("user", **new** User());

mav.addObject("newFlag", **true**);

mav.setViewName("user/editUser.jsp");

**return** mav;

}

@RequestMapping("/selectUserNotices")

**public** ModelAndView selectUserNotices(@RequestParam Integer user\_idUser, @RequestParam Integer notices\_noticeId, @RequestParam Integer notices\_userId, @RequestParam Integer notices\_categoryId) {

Notice notice = noticeDAO.findNoticeByPrimaryKey(notices\_noticeId, notices\_userId, notices\_categoryId, -1, -1);

ModelAndView mav = **new** ModelAndView();

mav.addObject("user\_idUser", user\_idUser);

mav.addObject("notice", notice);

mav.setViewName("user/notices/viewNotices.jsp");

**return** mav;

}

@RequestMapping("/editUserNotices")

**public** ModelAndView editUserNotices(@RequestParam Integer user\_idUser, @RequestParam Integer notices\_noticeId, @RequestParam Integer notices\_userId, @RequestParam Integer notices\_categoryId) {

Notice notice = noticeDAO.findNoticeByPrimaryKey(notices\_noticeId, notices\_userId, notices\_categoryId, -1, -1);

ModelAndView mav = **new** ModelAndView();

mav.addObject("user\_idUser", user\_idUser);

mav.addObject("notice", notice);

mav.setViewName("user/notices/editNotices.jsp");

**return** mav;

}

@RequestMapping("/userController/binary.action")

**public** ModelAndView streamBinary(@ModelAttribute HttpServletRequest request, @ModelAttribute HttpServletResponse response) {

ModelAndView mav = **new** ModelAndView();

mav.setViewName("streamedBinaryContentView");

**return** mav;

}

@RequestMapping("/deleteUser")

**public** String deleteUser(@RequestParam Integer idUserKey) {

User user = userDAO.findUserByPrimaryKey(idUserKey);

userService.deleteUser(user);

**return** "forward:/indexUser";

}

@RequestMapping("/saveUser")

**public** String saveUser(@ModelAttribute User user) {

userService.saveUser(user);

**return** "forward:/indexUser";

}

**public** String indexUser() {

**return** "redirect:/indexUser";

}

@RequestMapping("/listUserNotices")

**public** ModelAndView listUserNotices(@RequestParam Integer idUserKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("user", userDAO.findUserByPrimaryKey(idUserKey));

mav.setViewName("user/notices/listNotices.jsp");

**return** mav;

}

@RequestMapping("/confirmDeleteUser")

**public** ModelAndView confirmDeleteUser(@RequestParam Integer idUserKey) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("user", userDAO.findUserByPrimaryKey(idUserKey));

mav.setViewName("user/deleteUser.jsp");

**return** mav;

}

@RequestMapping("/newUserNotices")

**public** ModelAndView newUserNotices(@RequestParam Integer user\_idUser) {

ModelAndView mav = **new** ModelAndView();

mav.addObject("user\_idUser", user\_idUser);

mav.addObject("notice", **new** Notice());

mav.addObject("newFlag", **true**);

mav.setViewName("user/notices/editNotices.jsp");

**return** mav;

}

}