**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное образовательное учреждение высшего образования

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Институт кибернетики

Кафедра информатики и проектирования систем

Направление «Информатика и вычислительная техника»

Отчет по лабораторной работе №6 по дисциплине

**«Современные платформы программирования»**

Разработка многостраничного веб-сайта по шаблону MVC

Выполнил:

Студент группы 8ВМ81 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.Т. Хусаинова

Проверил:

Доцент ОИТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Н. Попов

Томск 2018

# Задание

Разработать многостраничный web-сайт на языке Java с использованием фреймворка (в данном случае – Spring), шаблона проектирования и технологии MVC.

Необходимо:

* Реализовать веб-сайт со следующими страницами: административная страница для удаления файлов и их описаний с сайта; пользовательская страница для загрузки файлов и их описаний на сайт; пользовательская страница для поиска файла по ключевым словам описания и скачивания файла.
* Защитить все административные страницы сайта парольным доступом.
* Дополнительно создать на сайте страницу с описанием контакта на разработчика сайта («Об авторе» / «Контакты» и т.п.) и страницу с описанием принципа работы сайта («О сайте»).
* Реализовать на web-сайте возможность протоколирования действия пользователя (нажатия на ссылки, заполнения полей) в файле (текстовом, xml, json) и (при потребности) отображать их перечень на странице приложения в табличном виде (дата, время, действие).

**Spring**

Фреймворк **Spring** обеспечивает архитектуру паттерна Model – View – Controller (Модель – Отображение (далее – Вид) – Контроллер) при помощи слабо связанных готовых компонентов. Паттерн MVC разделяет аспекты приложения (логику ввода, бизнес-логику и логику UI), обеспечивая при этом свободную связь между ними.

* **Model** (Модель) инкапсулирует (объединяет) данные приложения, в целом они будут состоять из POJO («Старых добрых Java-объектов», или бинов).
* **View** (Отображение, Вид) отвечает за отображение данных Модели, – как правило, генерируя HTML, которые мы видим в своём браузере.
* **Controller** (Контроллер) обрабатывает запрос пользователя, создаёт соответствующую Модель и передаёт её для отображения в Вид.

Ниже на диаграмме представлена структурная схема приложения:

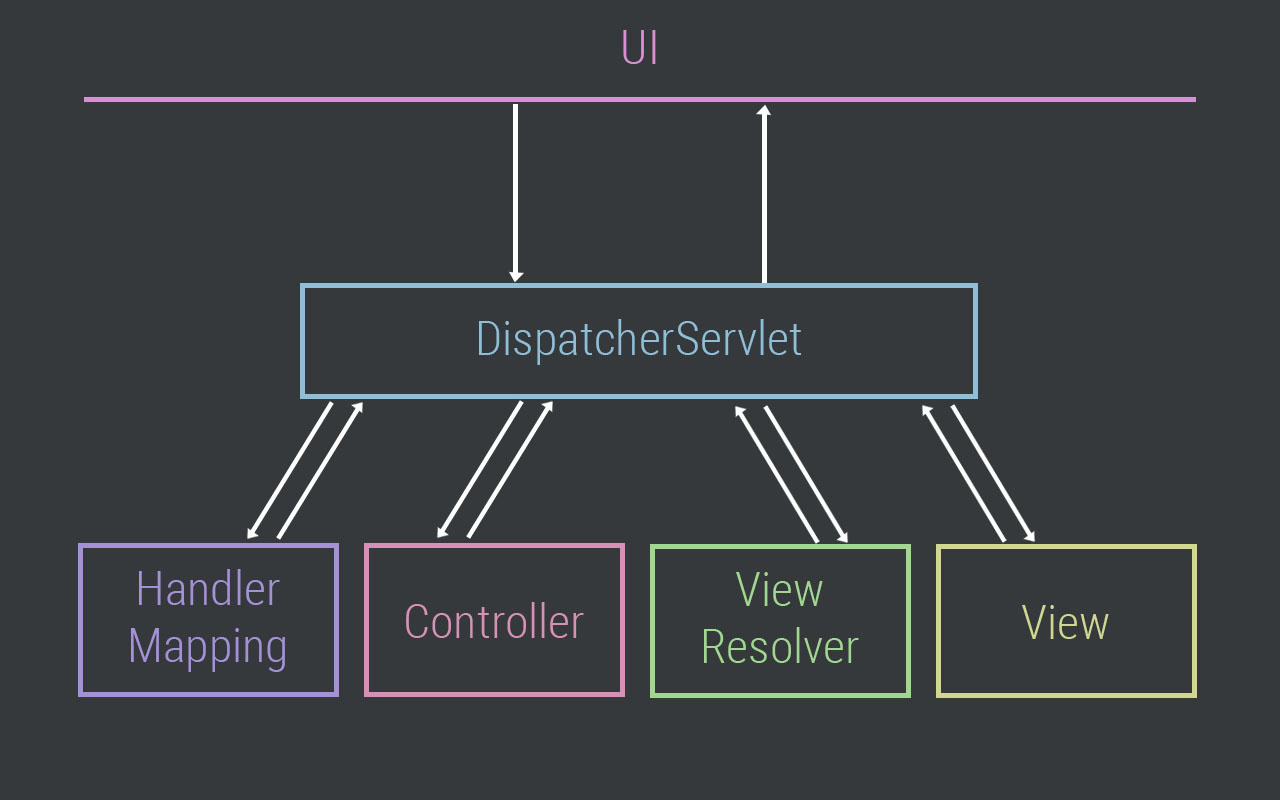


Рисунок 1 – Схема взаимодействия компонентов Spring

# Листинг

**Model**

package ru.tpu.file\_java.model;  
  
import org.hibernate.annotations.CreationTimestamp;  
import org.hibernate.annotations.GenericGenerator;  
  
import javax.persistence.\*;  
import java.util.Date;  
  
@Entity  
@Table(name = "files")  
public class DBFile {  
 @Id  
 @GeneratedValue(generator = "uuid")  
 @GenericGenerator(name = "uuid", strategy = "uuid2")  
 private String id;  
 private String fileName;  
 private String fileType;  
 private String description;  
private byte[] data;  
  
 @ManyToOne(fetch = FetchType.*EAGER*)  
 @JoinColumn(name = "user\_id")  
 private User creator;  
  
 @CreationTimestamp  
 @Temporal(TemporalType.*TIMESTAMP*)  
 @Column(name = "create\_date")  
 private Date createDate;  
  
 public DBFile() {  
 }  
  
 public DBFile(String fileName, String fileType, String description, byte[] data, User user) {  
 this.fileName = fileName;  
 this.fileType = fileType;  
 this.description = description;  
 this.data = data;  
 this.creator = user;  
 }  
  
 public String getId() {  
 return id;  
 }  
  
 public void setId(String id) {  
 this.id = id;  
 }  
  
 public String getFileName() {  
 return fileName;  
 }  
  
 public void setFileName(String fileName) {  
 this.fileName = fileName;  
 }  
  
 public String getFileType() {  
 return fileType;  
 }  
  
 public void setFileType(String fileType) {  
 this.fileType = fileType;  
 }  
  
 public String getDescription() {  
 return description;  
 }  
  
 public void setDescription(String description) {  
 this.description = description;  
 }  
  
 public byte[] getData() {  
 return data;  
 }  
  
 public void setData(byte[] data) {  
 this.data = data;  
 }  
  
 public User getCreator() {  
 return creator;  
 }  
  
 public void setCreator(User creator) {  
 this.creator = creator;  
 }  
  
 public Date getCreateDate() {  
 return createDate;  
 }  
  
 public void setCreateDate(Date createDate) {  
 this.createDate = createDate;  
 }  
}

**Controller**

package ru.tpu.file\_java.controller;  
  
import org.apache.log4j.Logger;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.core.io.ByteArrayResource;  
import org.springframework.core.io.Resource;  
import org.springframework.http.HttpHeaders;  
import org.springframework.http.MediaType;  
import org.springframework.http.ResponseEntity;  
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.PostMapping;  
import org.springframework.web.bind.annotation.RequestParam;  
import org.springframework.web.multipart.MultipartFile;  
import ru.tpu.file\_java.model.DBFile;  
import ru.tpu.file\_java.model.User;  
import ru.tpu.file\_java.repository.DBFileRepository;  
import ru.tpu.file\_java.service.DBFileStorageService;  
  
@Controller  
public class FileController {  
 private static final Logger *logger* = Logger.*getLogger*(FileController.class);  
  
 @Autowired  
 private DBFileRepository dbFileRepository;  
  
 @Autowired  
 private DBFileStorageService fileStorageService;  
  
 @GetMapping("/")  
 public String main(Model model) {  
 model.addAttribute("dbFiles", fileStorageService.loadAll());  
 return "main";  
 }  
  
 @PostMapping("/")  
 public String addFile(  
 @AuthenticationPrincipal User user,  
 @RequestParam String description,  
 @RequestParam("file") MultipartFile file  
 ) {  
 *logger*.info(String.*format*("Add new file %s", description));  
 fileStorageService.store(file, description, user);  
 return "redirect:/";  
 }  
  
 @GetMapping("/search")  
 public String search(@RequestParam(required = false, defaultValue = "") String search, Model model) {  
 Iterable<DBFile> dbFiles;  
 if (search != null && !search.isEmpty()) {  
 dbFiles = dbFileRepository.findByDescriptionContaining(search);  
 } else {  
 dbFiles = dbFileRepository.findAll();  
 }  
 *logger*.info(String.*format*("Find file by param %s", search));  
 model.addAttribute("dbFiles", dbFiles);  
 return "search";  
 }  
  
 @GetMapping("/downloadFile/{id}")  
 public ResponseEntity<Resource> downloadFile(@PathVariable String id) {  
 DBFile dbFile = fileStorageService.load(id);  
 *logger*.info(String.*format*("Download new file %s", id));  
  
 return ResponseEntity.*ok*()  
 .contentType(MediaType.*parseMediaType*(dbFile.getFileType()))  
 .header(HttpHeaders.*CONTENT\_DISPOSITION*, "attachment; filename=\"" + dbFile.getFileName() + "\"")  
 .body(new ByteArrayResource(dbFile.getData()));  
 }  
  
 @GetMapping("/deleteFile/{id}")  
 public String deleteFile(@PathVariable String id) {  
 *logger*.info(String.*format*("Deleted file %s", id));  
 fileStorageService.delete(id);  
 return "redirect:/";  
 }  
}

**Repository**

package ru.tpu.file\_java.repository;  
  
import org.springframework.data.jpa.repository.JpaRepository;  
import org.springframework.stereotype.Repository;  
import ru.tpu.file\_java.model.DBFile;  
  
import java.util.List;  
  
@Repository  
public interface DBFileRepository extends JpaRepository<DBFile, String> {  
 List<DBFile> findByDescriptionContaining(String description);  
}

**Service**

package ru.tpu.file\_java.service;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.stereotype.Service;  
import org.springframework.util.StringUtils;  
import org.springframework.web.multipart.MultipartFile;  
import ru.tpu.file\_java.model.DBFile;  
import ru.tpu.file\_java.model.User;  
import ru.tpu.file\_java.repository.DBFileRepository;  
  
import java.io.IOException;  
  
  
@Service  
public class DBFileStorageService {  
  
 @Autowired  
 private DBFileRepository dbFileRepository;  
  
 public Iterable<DBFile> loadAll() {  
 return dbFileRepository.findAll();  
 }  
  
 public void store(MultipartFile file, String description, User user) {  
 try {  
 String fileName = StringUtils.*cleanPath*(file.getOriginalFilename());  
 DBFile dbFile = new DBFile(fileName, file.getContentType(), description, file.getBytes(), user);  
 dbFileRepository.save(dbFile);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public DBFile load(String id) {  
 DBFile dbFile = dbFileRepository.findById(id).orElse(null);  
 if (dbFile != null) {  
 return dbFile;  
 } else {  
 return null;  
 }  
 }  
  
 public void delete(String id) {  
 dbFileRepository.findById(id).ifPresent(dbFile -> dbFileRepository.delete(dbFile));  
 }  
}

**Template (Common)**

<#macro page>

<!DOCTYPE html>

  <html lang="en">

  <head>

      <meta charset="UTF-8">

      <title>Загрузка файлов</title>

      <meta name="viewport"

            content="width=device-width, initial-scale=1, shrink-to-fit=no">

      <link rel="stylesheet"

            href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css"

            integrity="sha384-WskhaSGFgHYWDcbwN70/dfYBj47jz9qbsMId/iRN3ewGhXQFZCSftd1LZCfmhktB"

            crossorigin="anonymous">

  </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/1.14.3/umd/popper.min.js"

          integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"

          crossorigin="anonymous"></script>

  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"

          integrity="sha384-smHYKdLADwkXOn1EmN1qk/HfnUcbVRZyYmZ4qpPea6sjB/pTJ0euyQp0Mk8ck+5T"

          crossorigin="anonymous"></script>

  </body>

  </html>

</#macro>

**Template (Navbar)**

<#include "security.ftl">

<#import "login.ftl" as l>

<div class="wrap">

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

      <div class="container">

          <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>

                  <li class="nav-item">

                      <a class="nav-link" href="/search">Поиск</a>

                  </li>

                  <li class="nav-item">

                      <a class="nav-link" href="/contacts">Контакты</a>

                  </li>

                  <li class="nav-item">

                      <a class="nav-link" href="/about">О сайте</a>

                  </li>

              </ul>

              <div class="navbar-text mr-3">${name}</div>

              <@l.logout />

          </div>

      </div>

  </nav>

</div>

**Template (Main)**

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

<#include "parts/security.ftl">

<@c.page>

  <div class="page-header">

      <h3 class="mb-5">Главная</h3>

  </div>

  <form method="post" enctype="multipart/form-data">

  <div class="form-group row">

      <label class="col-sm-4 col-form-label" style="text-align: right; font-weight: bold;">Описание</label>

      <div class="col-sm-8">

          <div class="form-group">

              <input type="text" class="form-control" name="description" placeholder="Введите описание файла" />

          </div>

      </div>

  </div>

  <div class="form-group row">

      <label class="col-sm-4" style="text-align: right; font-weight: bold;">Файл</label>

      <div class="col-sm-8">

          <div class="form-group">

              <input type="file" name="file" id="customFile">

          </div>

      </div>

  </div>

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

  <div class="form-group row mb-5">

      <div class="col-sm-4"></div>

      <div class="col-sm-8">

          <div class="form-group">

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

          </div>

      </div>

  </div>

  </form>

  <div class="table">

      <table class="table table-hover">

          <thead>

              <tr>

                  <th width="40%">Наименование</th>

                  <th width="40%">Описание</th>

                  <th width="20%">Действие</th>

              </tr>

          </thead>

          <tbody>

          <#list dbFiles as dbFile>

              <tr>

                  <td>${dbFile.fileName}</td>

                  <td>${dbFile.description}</td>

                  <td>

                      <a class="btn btn-outline-primary" href="/downloadFile/${dbFile.id}" role="button">Скачать</a>

                  <#if isAdmin>

                      <a class="btn btn-outline-danger" href="/deleteFile/${dbFile.id}" role="button">Удалить</a>

                  </#if>

                  </td>

              </tr>

          </#list>

          </tbody>

      </table>

  </div>

</@c.page>

**Template (Search)**

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

<#include "parts/security.ftl">

<@c.page>

  <div class="page-header">

      <h3 class="mb-5">Поиск</h3>

  </div>

  <form method="get">

      <div class="form-group row">

          <label class="col-sm-4 col-form-label" style="text-align: right; font-weight: bold;">Описание</label>

          <div class="col-sm-8">

              <div class="form-group">

                  <input type="text" class="form-control" name="search" value="${search?ifExists}" placeholder="Введите описание файла" />

              </div>

          </div>

      </div>

      <div class="form-group row mb-5">

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

          <div class="col-sm-8">

              <div class="form-group">

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

              </div>

          </div>

      </div>

  </form>

  <div class="table">

      <table class="table table-hover">

          <thead>

              <tr>

                  <th width="40%">Наименование</th>

                  <th width="40%">Описание</th>

                  <th width="20%">Действие</th>

              </tr>

          </thead>

          <tbody>

          <#list dbFiles as dbFile>

              <tr>

                  <td>${dbFile.fileName}</td>

                  <td>${dbFile.description}</td>

                  <td>

                      <a class="btn btn-outline-primary" href="/downloadFile/${dbFile.id}" role="button">Скачать</a>

                  <#if isAdmin>

                      <a class="btn btn-outline-danger" href="/deleteFile/${dbFile.id}" role="button">Удалить</a>

                  </#if>

                  </td>

              </tr>

          </#list>

          </tbody>

      </table>

  </div>

</@c.page>

# Результаты работы

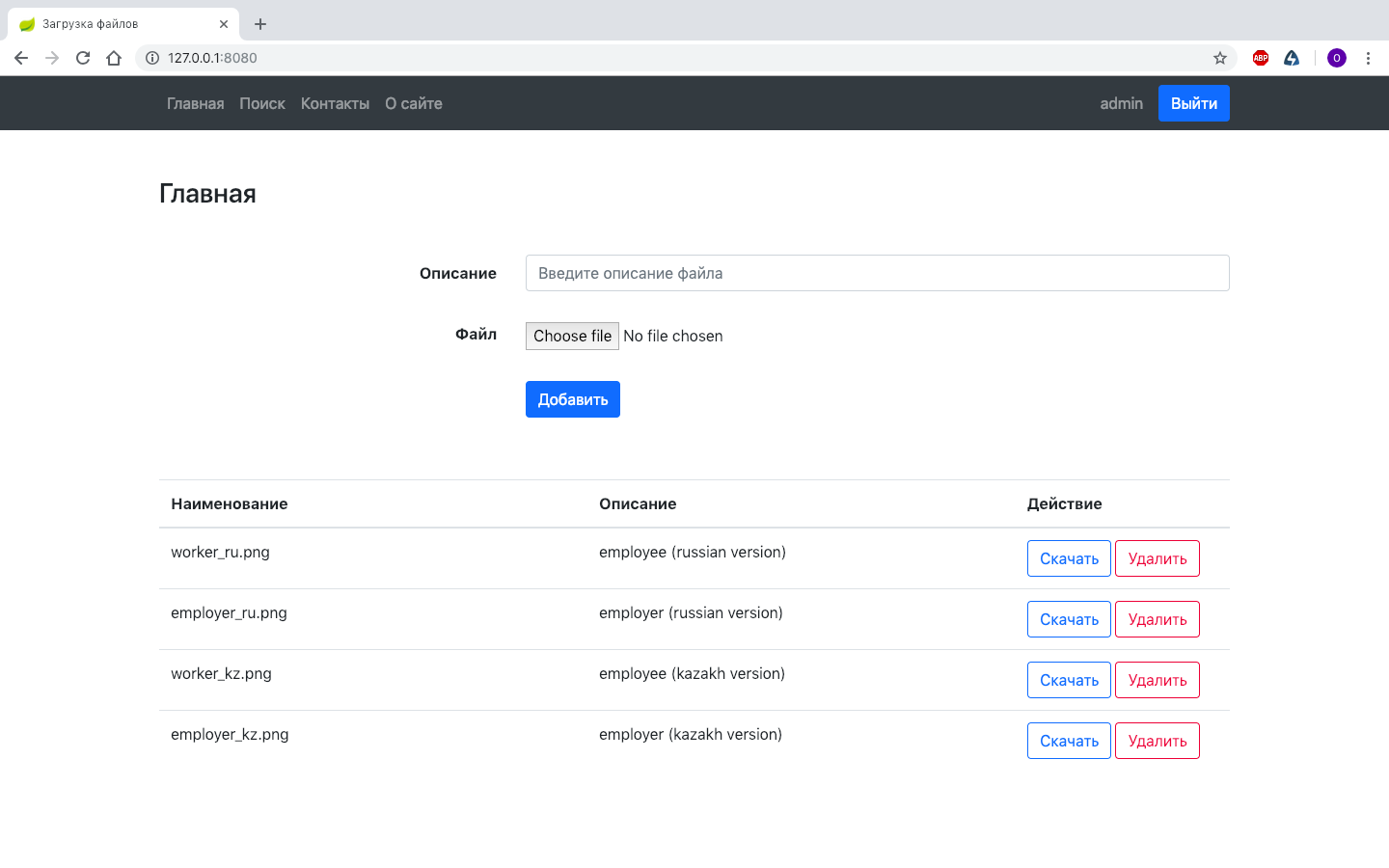


Рисунок 1 – Загрузка, скачивание и удаление (только для администратора) файлов

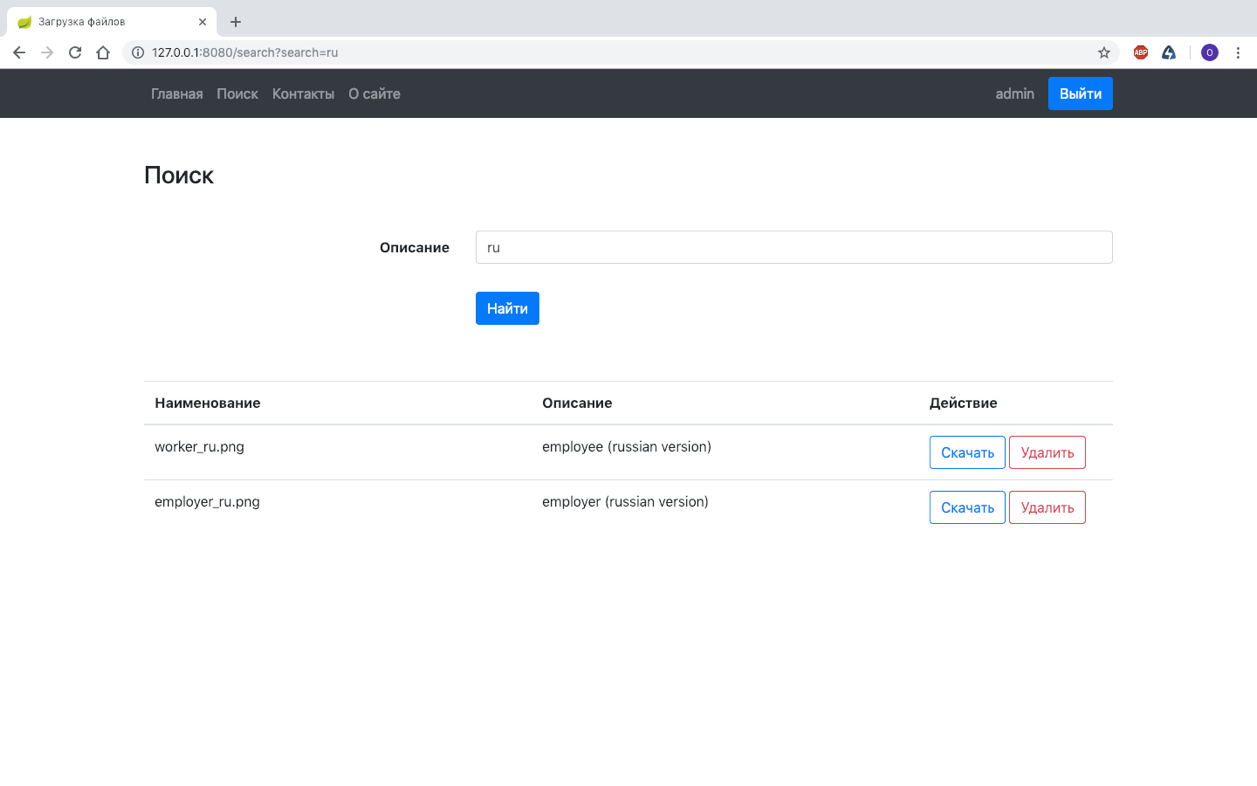


Рисунок 2 – Поиск файлов

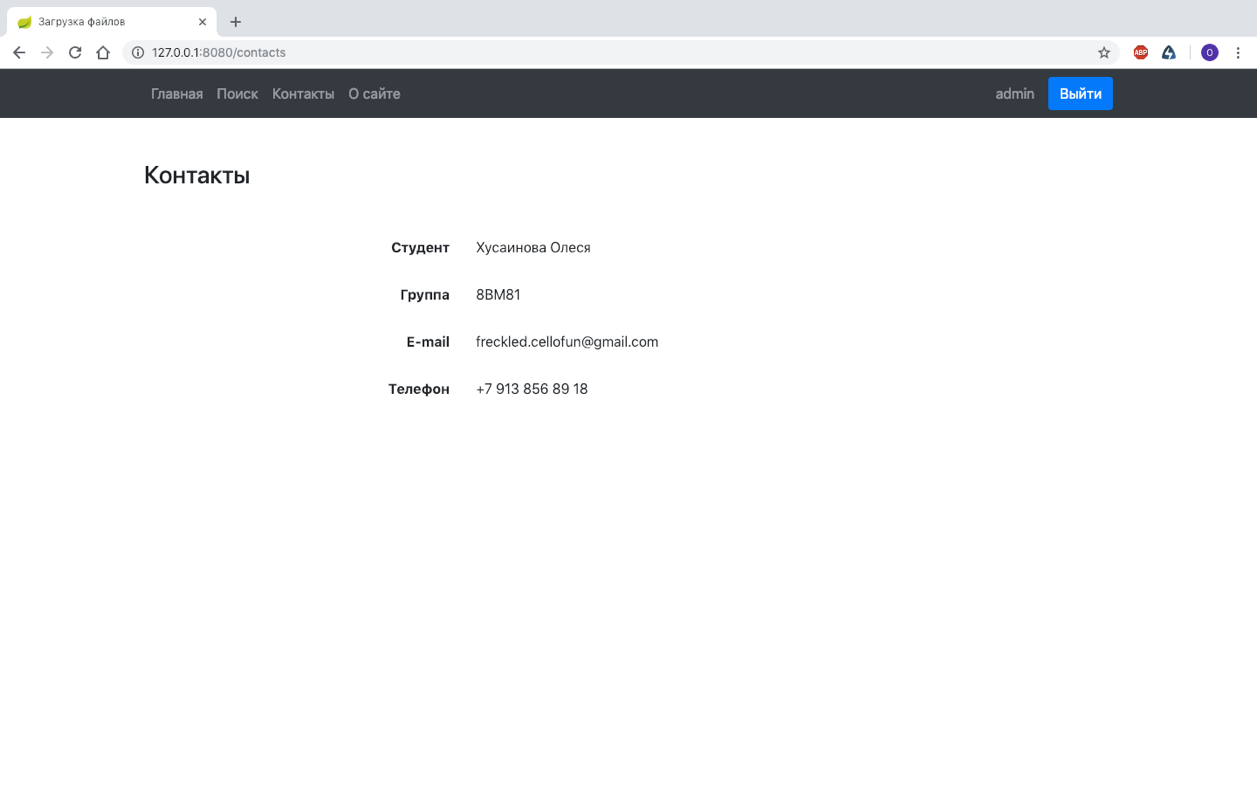


Рисунок 3 – Контакты

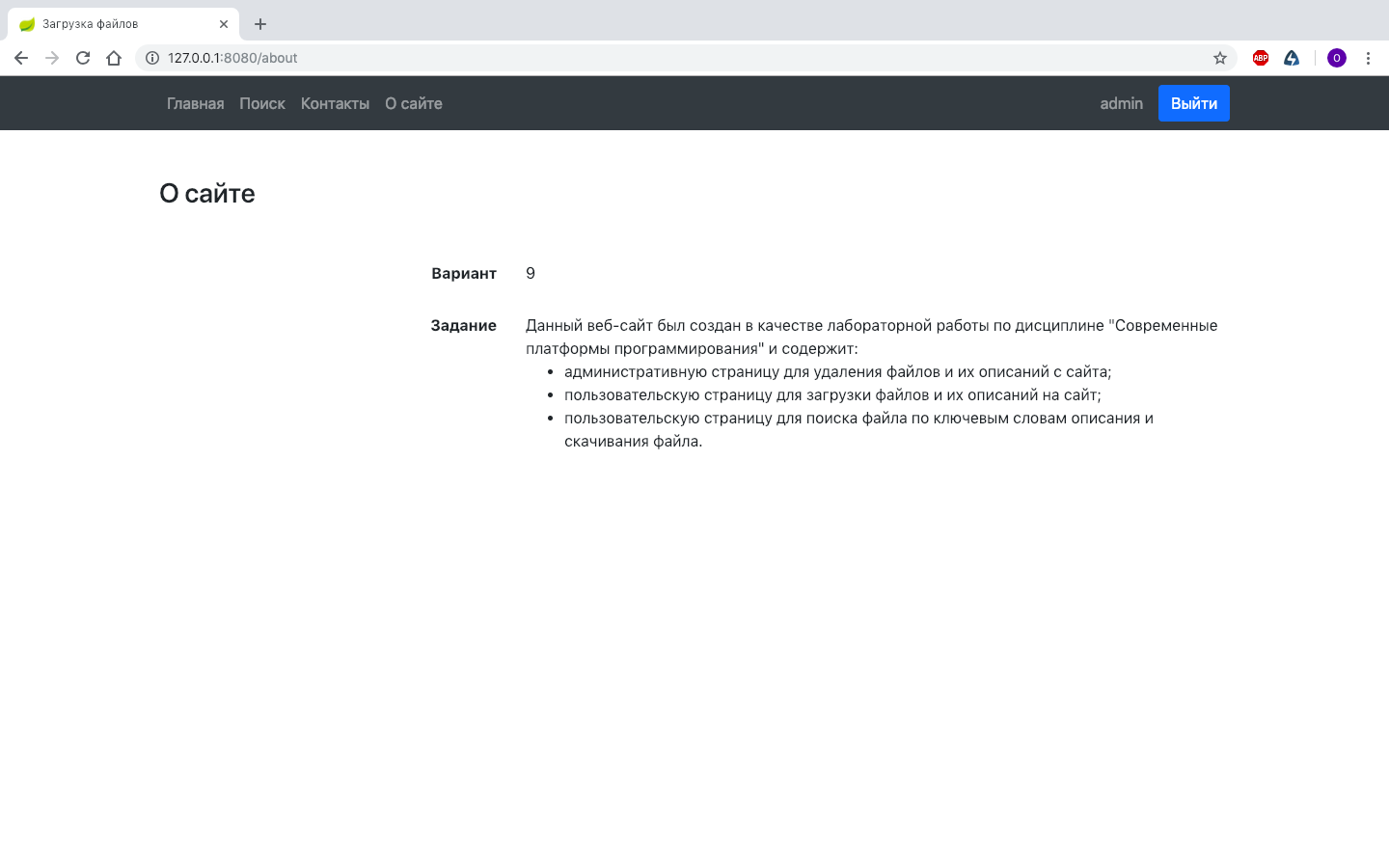


Рисунок 4 – О сайте

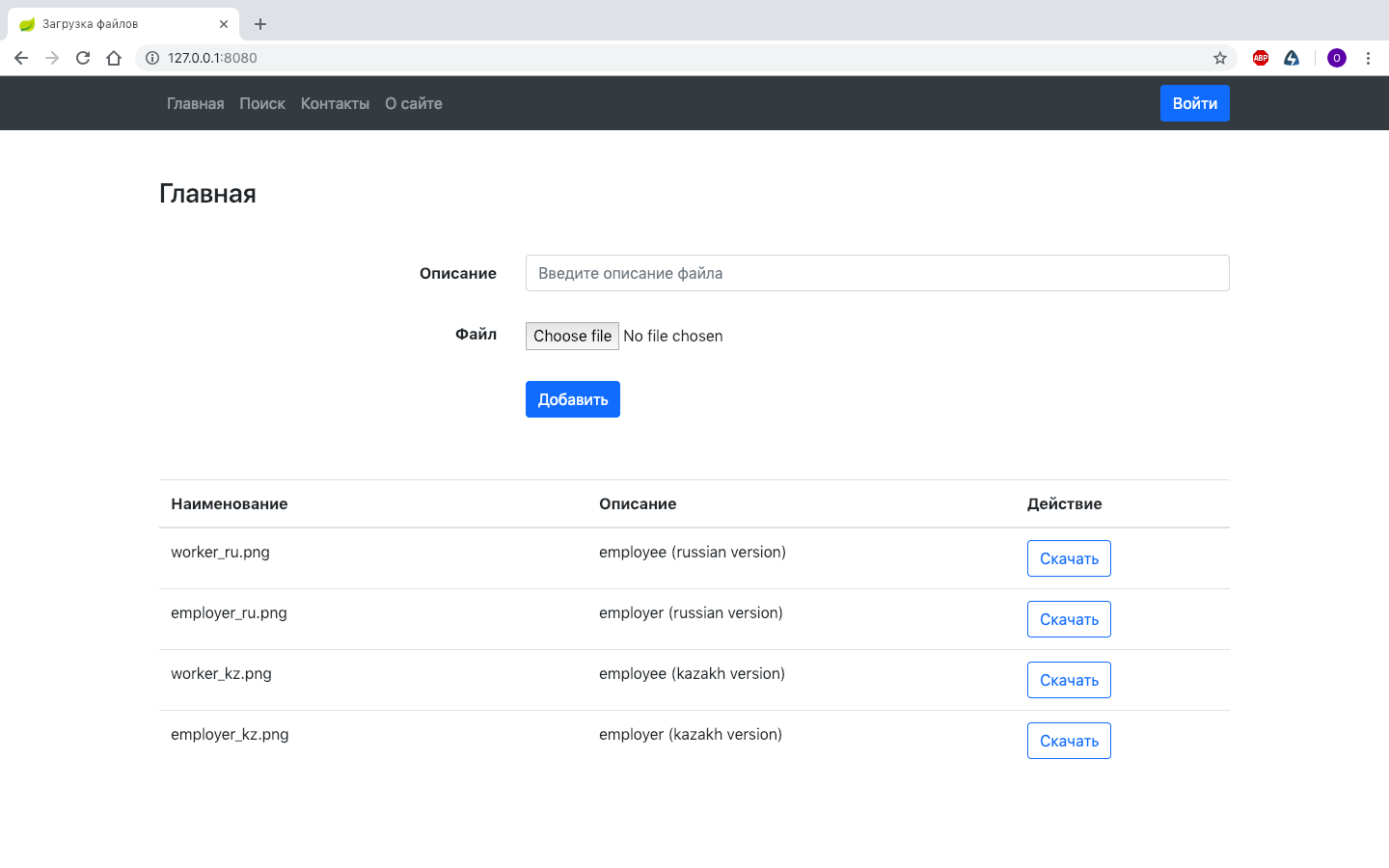


Рисунок 5 – Страницы, доступные для обычных пользователей

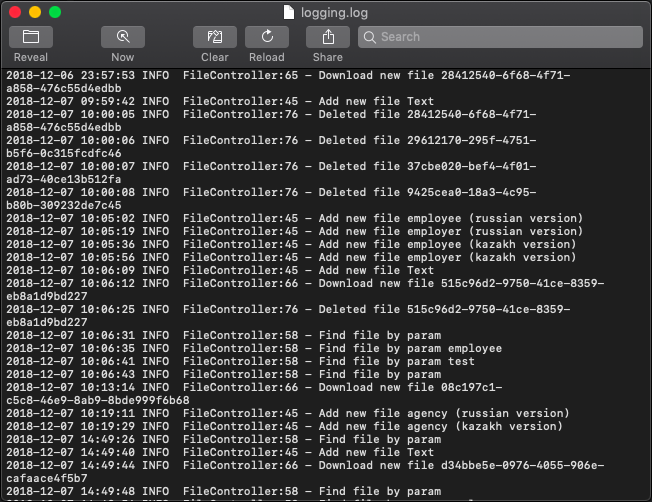


Рисунок 6 – Логирование действий

# Выводы

На основе фреймворка Spring был реализован веб-сайт со следующими страницами:

* административная страница для удаления файлов и их описаний с сайта;
* пользовательская страница для загрузки файлов и их описаний на сайт;
* пользовательская страница для поиска файла по ключевым словам описания и скачивания файла.

Все основные действия пользователя логируются в файл.