**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 Python с использованием фреймворка (в данном случае – Django), 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).

**Django**

Django – это высокоуровневая веб-инфраструктура Python, которая позволяет быстро создавать безопасные и поддерживаемые веб-сайты. Построенный опытными разработчиками, Django заботится о многих проблемах веб-разработки, поэтому вы можете сосредоточиться на написании своего приложения без необходимости изобретать колесо. Это бесплатный и открытый источник, имеет процветающее и активное сообщество, отличную документацию и множество опций для бесплатной и платной поддержки.

Django реализует уровневую архитектуру "Model View Template (MVT)". Она имеет много общего с более известной архитектурой Model View Controller. Ниже на диаграмме представлена структурная схема приложения:



Рисунок 1 – Схема взаимодействия компонентов Django

Веб-приложения написанные на Django обычно группируют код, который обрабатывает каждый из этих шагов, в отдельные файлы:

* URLs: Хотя можно обрабатывать запросы с каждого URL-адреса с помощью одной функции, гораздо удобнее писать отдельную функцию для обработки каждого ресурса. URL-mapper используется для перенаправления HTTP-запросов в соответствующее представление на основе URL-адреса запроса. URL-mapper также может извлекать данные из URL-адреса в соответствии с заданным шаблоном и передавать их в соответствующую функцию в виде аргументов.
* View: Представление (view) – это функция обработчика запросов, которая получает HTTP-запросы и возвращает ответы. View имеет доступ к данным через модели (необходимым для удовлетворения запросов и делегирования ответа в шаблоны).
* Models: Модели представляют собой объекты Python, которые определяют структуру данных приложения и предоставляют механизмы для управления (добавления, изменения, удаления) и выполнения запросов в базу данных.
* Templates: Template (шаблон) – это текстовый файл, определяющий структуру или разметку страницы (например HTML страницы), с полями для подстановки используемыми для представления актуального содержимого. View может динамически создавать HTML страницы, используя HTML шаблоны и заполняя их данными из модели (model). Шаблон может быть использован для определения структуры файлов любых типов, не обязательно HTML.

# Листинг

**Model**

from django.db import models

class Post(models.Model):

  title = models.CharField(max\_length=255)

  upload = models.FileField(upload\_to='uploads/')

**View**

import os  
  
from django.conf import settings  
from django.http import HttpResponse, Http404  
from django.shortcuts import render  
from django.urls import reverse\_lazy  
from django.views.generic import ListView, CreateView, DeleteView  
  
from .models import Post  
  
class PostUserList(ListView):  
 model = Post  
 template\_name = 'uploader/post\_list.html'  
 context\_object\_name = 'posts'  
  
 def get\_queryset(self):  
 query = self.request.GET.get('q')  
 if query:  
 return Post.objects.filter(title\_\_icontains=query)  
 else:  
 return Post.objects.all()  
  
class PostUserListDelete(ListView):  
 model = Post  
 template\_name = 'uploader/post\_delete.html'  
 context\_object\_name = 'posts'  
 success\_url = reverse\_lazy('post\_delete')  
  
class NewPost(CreateView):  
 model = Post  
 fields = ['title', 'upload']  
 template\_name\_suffix = '\_new'  
 success\_url = reverse\_lazy('post\_list')  
  
class DeletePost(DeleteView):  
 model = Post  
 success\_url = reverse\_lazy('post\_list')  
  
def about(request):  
 return render(request, 'uploader/about.html')  
  
  
def contact(request):  
 return render(request, 'uploader/contact.html')  
  
def download(request, pk):  
 post = Post.objects.get(id=pk)  
 file\_path = os.path.join(settings.MEDIA\_ROOT, post.upload.path)  
 if os.path.exists(file\_path):  
 with open(file\_path, 'rb') as fh:  
 response = HttpResponse(fh.read(), content\_type="application/vnd.ms-excel")  
 response['Content-Disposition'] = 'inline; filename=' + os.path.basename(file\_path)  
 return response  
 raise Http404

**Template (Base)**

{% load staticfiles %}

<html>

<head>

  <title>Загрузка файлов</title>

  <link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css">

  <link rel="stylesheet" href="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap-theme.min.css">

  <link href='//fonts.googleapis.com/css?family=Lobster&subset=latin,latin-ext' rel='stylesheet' type='text/css'>

  <link rel="stylesheet" href="{% static 'css/style.css' %}">

</head>

<body>

<div class="wrap">

  <nav class="navbar navbar-expand-lg navbar-dark bg-gradient-success" role="navigation">

      <div class="container">

          <div class="navbar-header">

              <a class="navbar-brand" href="/">Файлообменник</a>

          </div>

          <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">

              <ul class="collapse navbar-collapse nav">

                  <li><a href="/">Поиск</a></li>

                  <li><a href="{% url 'post\_new' %}">Загрузка</a></li>

                  {% if request.user.is\_superuser %}

                      <li><a href="{% url 'post\_delete' %}">Удаление</a></li>

                  {% endif %}

                  <li><a href="{% url 'contact' %}">Контакты</a></li>

                  <li><a href="{% url 'about' %}">О сайте</a></li>

              </ul>

              <form class="navbar-form navbar-right" role="login" style="margin: 0; padding-right: 0">

                  <div class="login collapse navbar-collapse nav" style="padding-right: 0">

                      {% if user.is\_authenticated %}

                          <li>

                              <a href="{% url 'logout' %}">Выйти</a>

                          </li>

                      {% else %}

                          <li>

                              <a href="{% url 'login' %}">Войти</a>

                          </li>

                      {% endif %}

                  </div>

              </form>

          </div>

      </div>

  </nav>

  <div class="container">

      <div class="row">

          <div class="col-md-12">

              {% block content %}

              {% endblock %}

          </div>

      </div>

  </div>

</div>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

<script src="/static/js/bootstrap.min.js"></script>

</body>

</html>

**Template (Post\_list)**

{% extends 'uploader/base.html' %}

{% block content %}

  <div class="site-index card border-secondary">

      <div class="card-header"><h3>Поиск файлов</h3></div>

      <div class="body-content card-body">

          <form method="GET" action="">

              <div class="form-group row">

                  <label class="col-sm-4 col-form-label">Описание</label>

                  <div class="col-sm-8">

                      <div class="form-group field-post-desc required">

                          <input name="q" type="text" class="form-control" />

                      </div>

                  </div>

              </div>

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

                  <input type="submit" class="btn btn-info single-button mb60" value="Найти"/>

              </div>

              <table class="table">

                  <thead>

                  <tr>

                      <th class="col-md-8">Описание</th>

                      <th class="col-md-4">Действие</th>

                  </tr>

                  </thead>

                  <tbody>

                      {% for post in posts %}

                          <tr>

                              <td style="padding: 20.8px 8px">{{ post.title }}</td>

                              <td>

                                  <a class="btn btn-link" style="padding-left: 0" href="{% url 'download' post.id %}">Скачать</a>

                              </td>

                          </tr>

                      {% endfor %}

                  </tbody>

              </table>

          </form>

      </div>

  </div>

{% endblock %}

**Template (Post\_new)**

{% extends "uploader/base.html" %}

{% block content %}

  <div class="site-index card border-secondary">

      <div class="card-header"><h3>Загрузка файлов</h3></div>

      <div class="body-content card-body">

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

              {% csrf\_token %}

              <div class="form-group row">

                  <label class="col-sm-4 col-form-label">Файл</label>

                  <label class="btn btn-outline-info custom-button">

                      <div class="form-group field-post-upload">

                          <div class="hidden">

                              {{ form.upload }}

                          </div>

                          <label class="control-label">Выбрать файл</label>

                      </div>

                  </label>

              </div>

              <div class="form-group row">

                  <label class="col-sm-4 col-form-label">Описание</label>

                  <div class="col-sm-8">

                      <div class="form-group field-post-desc required">

                          <input class="form-control" {{ form.title }}

                      </div>

                  </div>

              </div>

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

                  <button type="submit" class="btn btn-info single-button">Загрузить</button>

              </div>

          </form>

      </div>

  </div>

{% endblock content %}

**Template (Post\_delete)**

{% extends 'uploader/base.html' %}

{% block content %}

  <div class="site-index card border-secondary">

      <div class="card-header"><h3>Удаление файлов</h3></div>

      <div class="body-content card-body">

          <table class="table">

              <thead>

              <tr>

                  <th class="col-md-8">Описание</th>

                  <th class="col-md-4">Действие</th>

              </tr>

              </thead>

              <tbody>

                  {% for post in posts %}

                      <tr>

                          <td style="padding: 20.8px 8px">{{ post.title }}</td>

                          <td>

                              <a class="btn btn-link" style="padding-left: 0" href="{% url 'post\_delete' post.id %}">Удалить</a>

                          </td>

                      </tr>

                  {% endfor %}

              </tbody>

          </table>

      </div>

  </div>

{% endblock %}

# Результаты работы

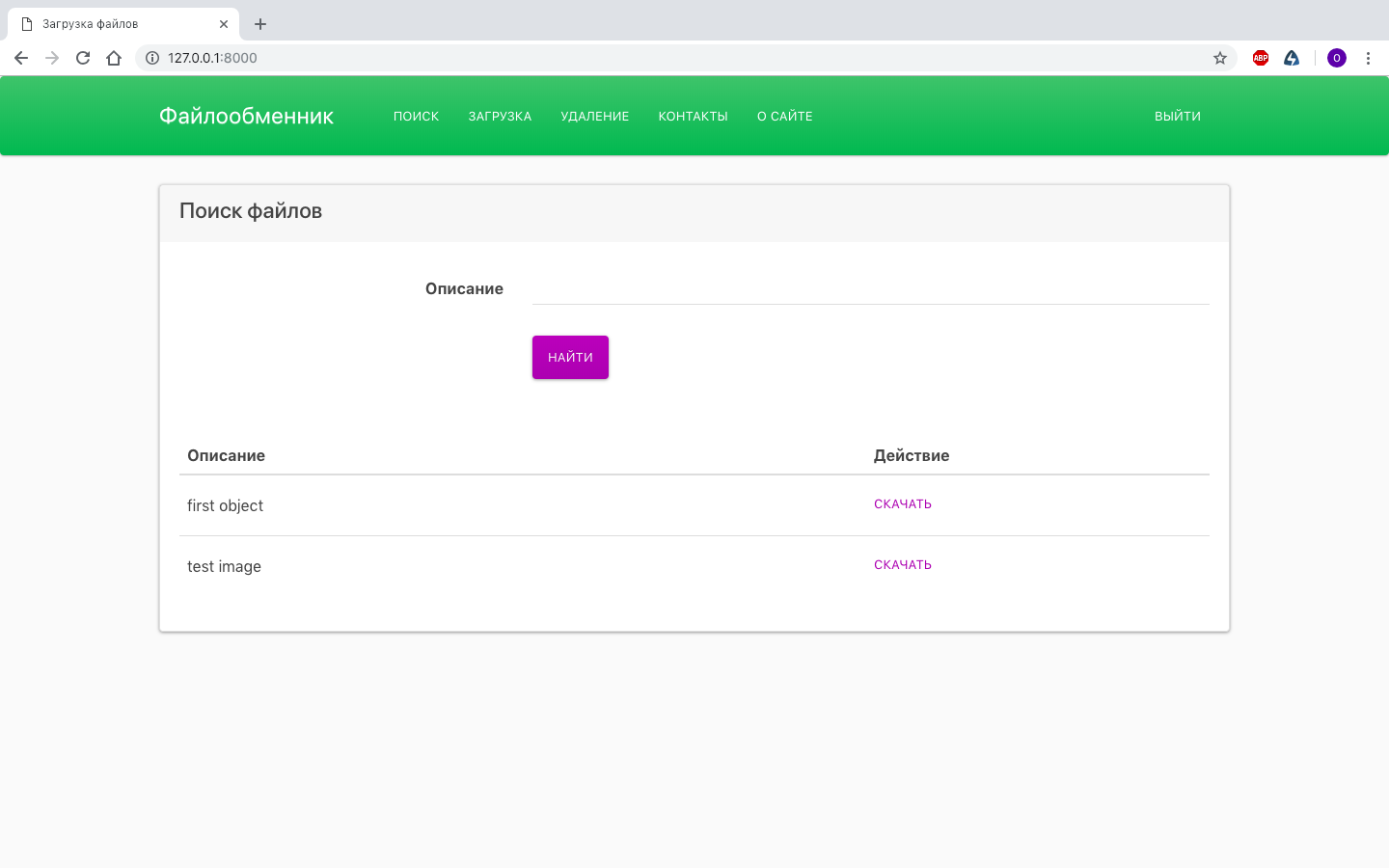


Рисунок 1 – Поиск и скачивание файлов

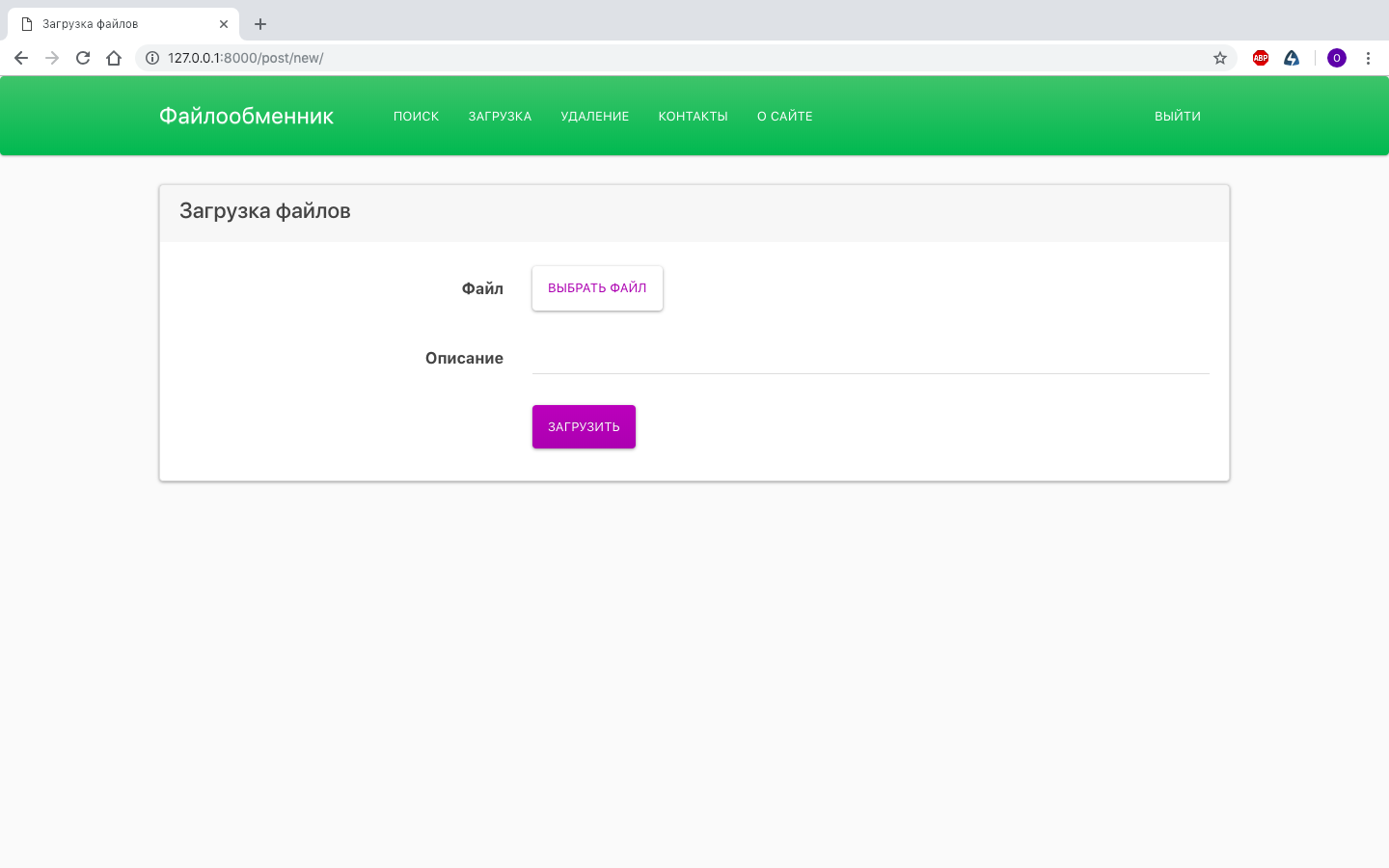


Рисунок 2 – Загрузка файлов

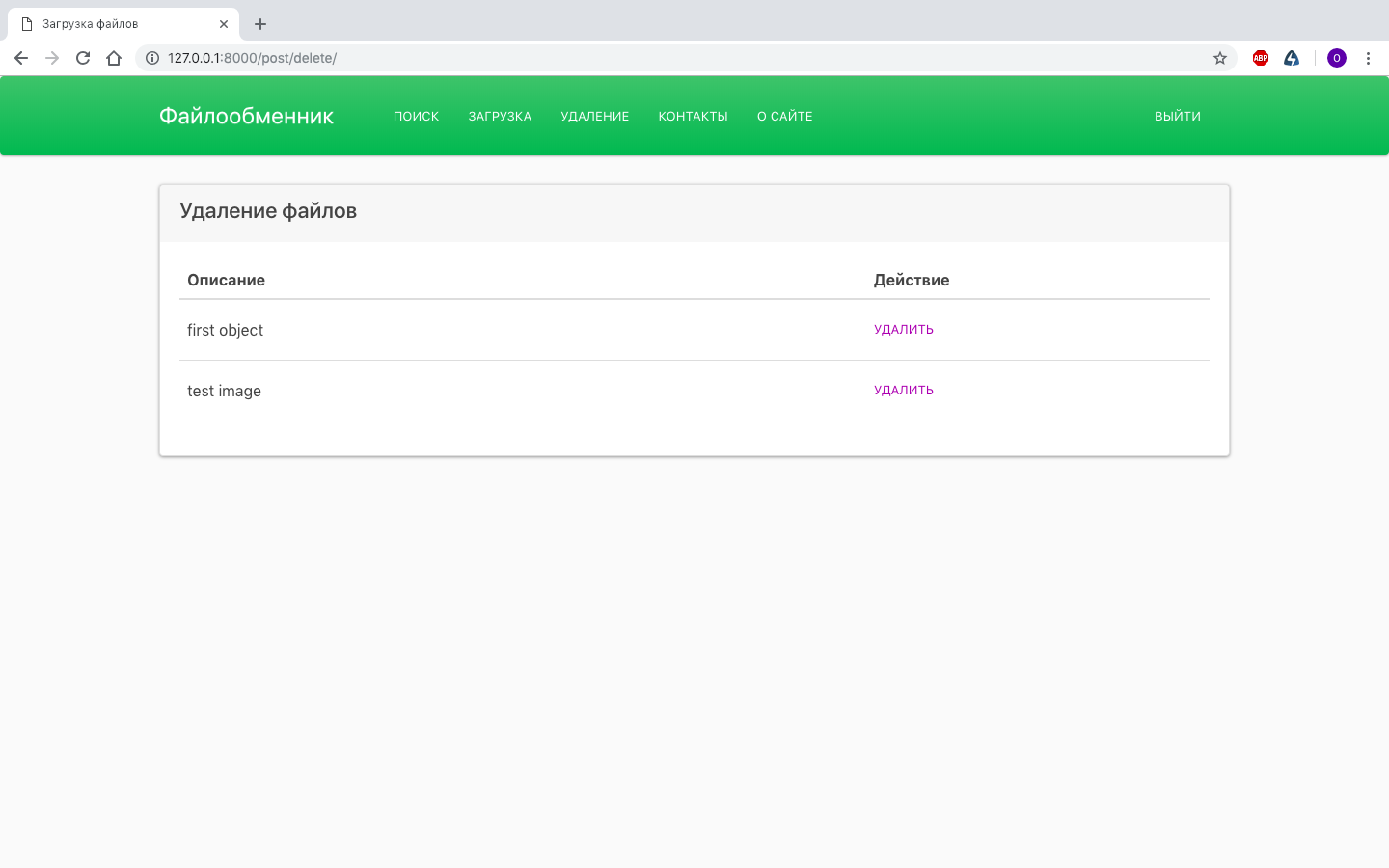


Рисунок 3 – Удаление файлов

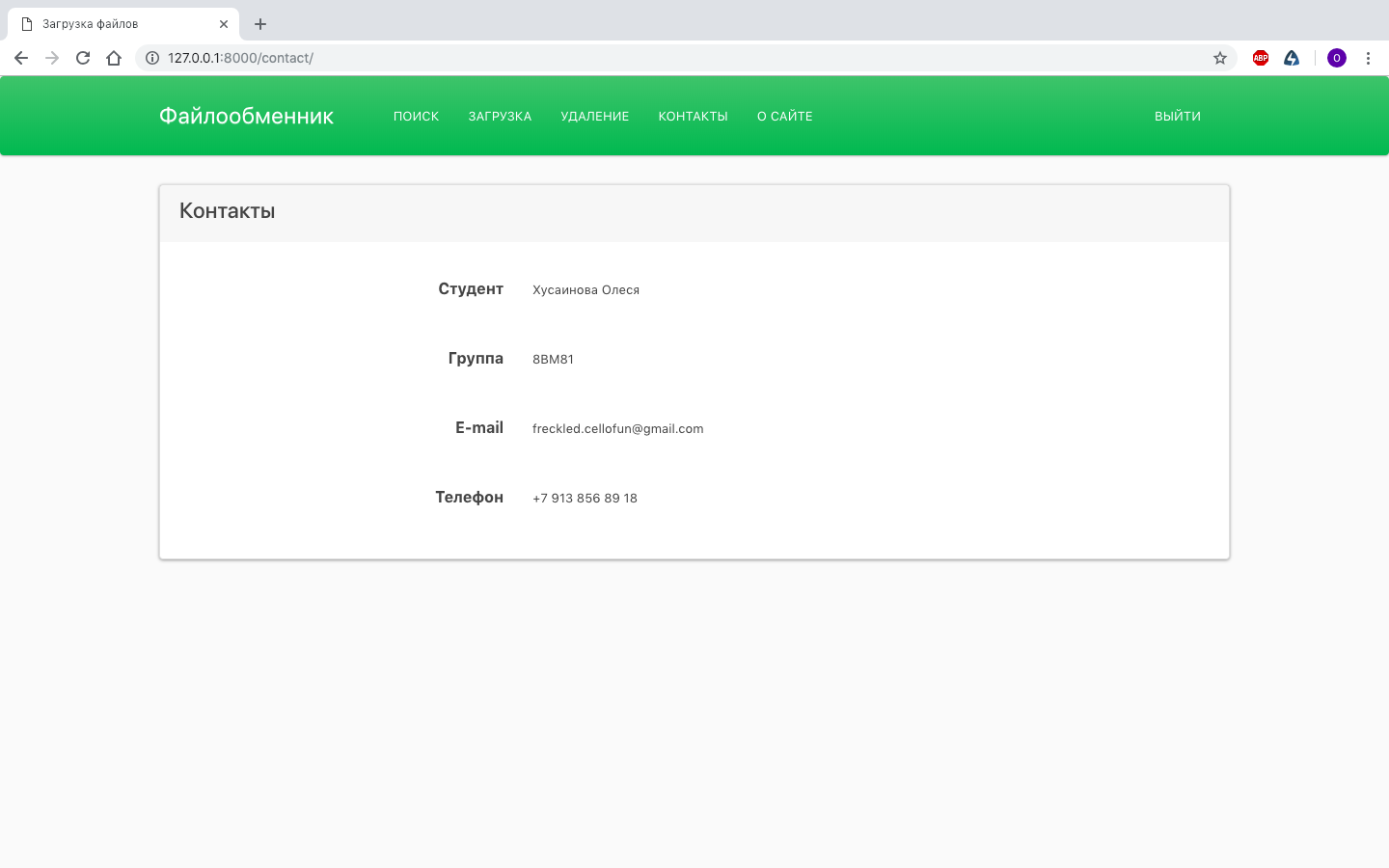


Рисунок 4 – Контакты

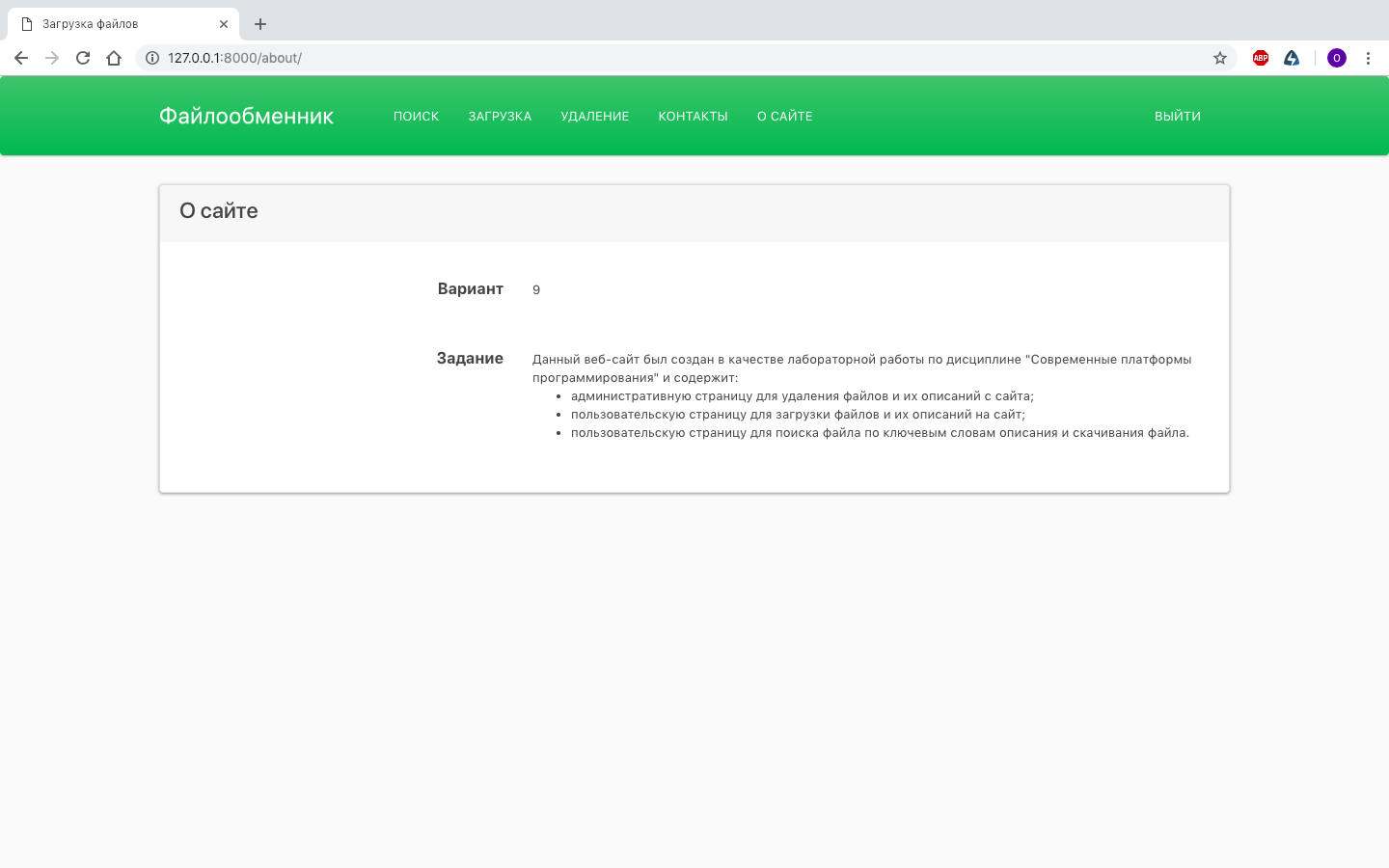


Рисунок 5 – О сайте

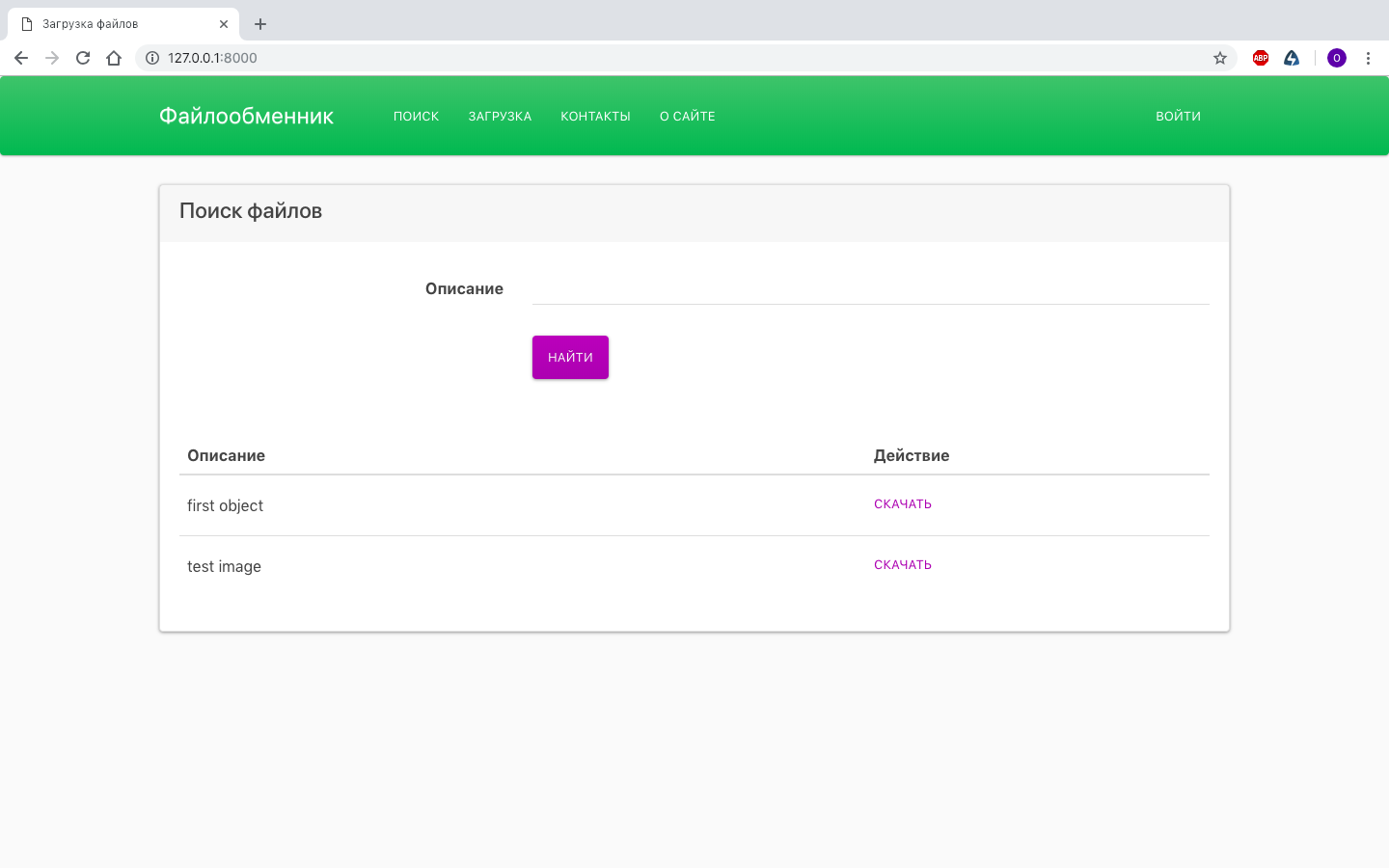


Рисунок 6 – Страницы, доступные для обычных пользователей

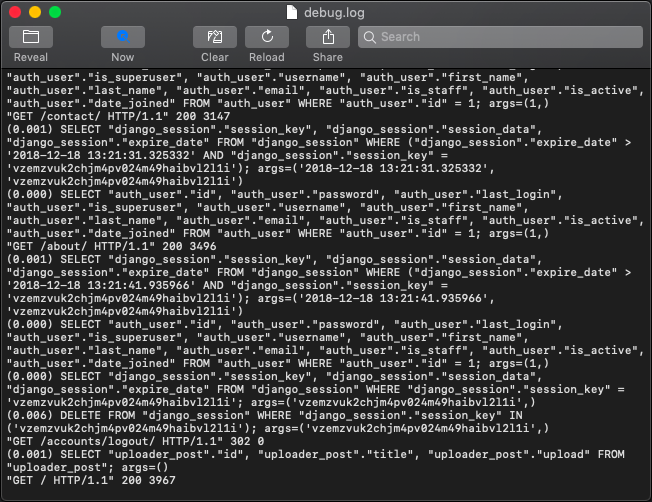


Рисунок 7 – Логирование действий

# Выводы

На основе фреймворка Django 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.