Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления» Кафедра «Системы обработки информации и управления»

Студент Андреев К.А.

Отчет

по выполнению лабораторной работы По курсу

“Разработка интернет-приложений”

Лабораторная работа № 6

Задание и порядок выполнения

Основная цель данной лабораторной работы – научиться обрабатывать веб-формы на стороне приложения, освоить инструменты, которые предоставляет Django, по работе с формами. Также в этой лабораторной работе вы освоите инструменты Django по работе с авторизацией и реализуете простейшую авторизацию. Напоследок, вы познакомитесь с инструментом администрирования Django – как в несколько строчек кода сделать панель администратора сайта.

1. Создайте view, которая возвращает форму для регистрации.

Поля формы:

• Логин

• Пароль

• Повторный ввод пароля

• Email

• Фамилия

• Имя

2. Создайте view, которая возвращает форму для авторизации.

Поля формы:

• Логин

• Пароль

3. При отправке формы регистрации во view проверять каждый параметр по правилам валидации, если валидация всех полей пройдена, то создавать пользователя и делать перенаправление на страницу логина, а ошибки, если они есть, выводить над формой.

Правила валидации:

• Логин не меньше 5 символов

• Пароль не меньше 8 символов

• Пароли должны совпадать

• Все поля должны быть заполнены

• Логин – уникален для каждого пользователя

4. При возникновении ошибок в момент отправки формы, введенные значения в полях ввода, кроме пароля, не должны исчезать.

5. Переписать view регистрации с использованием Django Form, правила валидации удалить из view, использовать встроенный механизм валидации полей.

6. Во view авторизации реализовать логин при POST запросе. При успешной авторизации должен происходить переход на страницу успешной авторизации.

7. Страница успешной авторизации должна проверять, что пользователь авторизован. Иначе делать перенаправление на страницу авторизации.

8. Реализовать view для выхода из аккаунта.

9. Заменить проверку на авторизацию на декоратор login_required

10. Добавить superuser'а через комманду manage.py

11. Подключить django.contrib.admin и войти в панель администрирования.

12. Зарегистрировать все свои модели в django.contrib.admin

13. Для выбранной модели настроить страницу администрирования:

• Настроить вывод необходимых полей в списке

• Добавить фильтры

• Добавить поиск

• Добавить дополнительное поле в список

Исходники:

Views.py:

```
from django.shortcuts import render
from django.http import HttpResponseRedirect, HttpResponse
from django.views.generic import ListView
from django import forms
from django.contrib.auth import authenticate,login, logout
from django.contrib.auth.decorators import login_required
from django.contrib.auth.mixins import LoginRequiredMixin
from .models import  *
from django.core.paginator import Paginator, EmptyPage, PageNotAnInteger

class UserList(LoginRequiredMixin, ListView):
    login_url = '/labs/'
    redirect_field_name = 'redirect_to'
    model = Userus
    template_name = 'user_list.html'

class FilmList(LoginRequiredMixin, ListView):
    login_url = '/labs/'
    redirect_field_name = 'redirect_to'
    model = Film
```

```
    template_name = 'film_list.html'

    paginate_by = 3


class ReviewList(LoginRequiredMixin, ListView):
    login_url = '/labs/'
    redirect_field_name = 'redirect_to'
    model = Review
    template_name = 'review_list.html'

def registration_dumb(request):
    errors = {}
    request.encoding = 'utf-8'
    if request.method == 'POST':
        username = request.POST.get('username')
        if not username:
            errors['uname']='Введите логин'
        elif len(username) < 5:
            errors['uname']='Длина логина должна быть не меньше 5 символов'

        if User.objects.filter(username=username).exists():
            errors['uname']='Такой логин уже занят'

        password = request.POST.get('password')
        if not password:
            errors['psw']='Введите пароль'
        elif len(password) < 8:
            errors['psw']='Длина пароля должна быть не меньше 8 символов'

        password_repeat = request.POST.get('password2')
        if password != password_repeat:
            errors['psw2']='Пароли должны совпадать'

        email = request.POST.get('email')
        if not email:
            errors['email']='Введите email'

        last_name = request.POST.get('last_name')
        if not last_name:
            errors['lname']='Введите фамилию'

        first_name = request.POST.get('first_name')
        if not first_name:
            errors['fname']='Введите имя'

        if not errors:
            user = User.objects.create_user(username, email, password)
            usr = Userus()
            usr.user = user
            usr.first_name = first_name
            usr.last_name = last_name
            usr.save()
            return HttpResponseRedirect('/labs/users')
        else:
            context = {'errors':errors, 'username':username, 'email': email,
'last_name': last_name, 'first_name': first_name}
            return render(request, 'registration_dumb.html',context)

    return render(request, 'registration_dumb.html',{'errors':errors})

class RegistrationForm(forms.Form):
```

```
    username = forms.CharField(min_length=5,label='Логин')
    password = forms.CharField(min_length=8,widget=forms.PasswordInput,
label='Пароль')
    password2 = forms.CharField(min_length=8, widget=forms.PasswordInput,
label='Повторите ввод')
    email = forms.EmailField(label='Email')
    last_name = forms.CharField(label='Фамилия')
    first_name = forms.CharField(label='Имя')

def registration_user(request):
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        is_val = form.is_valid()
        data = form.cleaned_data
        if data['password']!=data['password2']:
            is_val = False
            form.add_error('password2',['Пароли должны совпадать'])
        if User.objects.filter(username=data['username']).exists():
            form.add_error('username',['Такой логин уже занят'])
            is_val = False

        if is_val:
            data = form.cleaned_data
            user = User.objects.create_user(data['username'], data['email'],
data['password'])
            usr = Userus()
            usr.user = user
            usr.first_name = data['first_name']
            usr.last_name = data['last_name']
            usr.nickname = data['username']
            usr.save()
            return HttpResponseRedirect('/labs/authorization')
    else:
        form = RegistrationForm()

    return render(request, 'registration_user.html',{'form':form})

@login_required(login_url='/labs/authorization')
def success_authorization(request):
    return HttpResponseRedirect('/labs')


def success_authorization_dumb(request):
    if request.user.is_authenticated:
        return HttpResponseRedirect('/labs/')
    else:
        return HttpResponseRedirect('/labs/authorization')


def authorization(request):
    errors = {}
    if request.method == 'POST':
        username = request.POST.get('username')
        if not username:
            errors['uname']='Введите логин'
        elif len(username) < 5:
            errors['uname']='Длина логина должна быть не меньше 5 символов'

        password = request.POST.get('password')
        if not password:
            errors['psw']='Введите пароль'
        elif len(password) < 8:
            errors['psw']='Длина пароля должна быть не меньше 8 символов'
```

```
        user = authenticate(request, username=username, password=password)
        if user is None and 'uname' not in errors.keys() and 'psw' not in
errors.keys():
            errors['login'] = 'Логин или пароль введены неправильно'

        if not errors:
            login(request,user)
            #return HttpResponseRedirect('/labs/success authorization dumb')
            return HttpResponseRedirect('/labs/success_authorization')
        else:
            context = {'errors':errors}
            return render(request, 'authorization.html',context)

    return render(request, 'authorization.html',{'errors':errors})

def logout_view(request):
    logout(request)
    return HttpResponseRedirect('/labs/')

class AutorizationForm(forms.Form):
    pass

def index(request):
    return render(request, 'index.html')

def index2(request):
    return HttpResponseRedirect('/labs/')

class FilmAddForm(forms.Form):
    name = forms.CharField(min_length=5,label='Название')
    genre = forms.CharField(label='Жанр')
    description = forms.CharField(label='Описание')


def add_film(request):
    if request.method == 'POST':
        form = FilmAddForm(request.POST)
        is_val = form.is_valid()
        data = form.cleaned_data
        if Film.objects.filter(name=data['name']).exists():
            form.add_error('name',['Этот фильм уже есть в базе'])
            is_val = False

        if is_val:
            data = form.cleaned_data
            flm = Film()
            flm.name = data['name']
            flm.genre = data['genre']
            flm.description = data['description']
            flm.save()
            return HttpResponseRedirect('/labs/films')
    else:
        form = FilmAddForm()

    return render(request, 'add_film.html',{'form':form})
```

Urls.py

```
"""lab6 URL Configuration
```

```
The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/1.10/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  url(r'^$', views.home, name='home')
Class-based views
    1. Add an import:  from other app.views import Home
    2. Add a URL to urlpatterns:  url(r'^$', Home.as view(), name='home')
Including another URLconf
    1. Import the include() function: from django.conf.urls import url, include
    2. Add a URL to urlpatterns:  url(r'^blog/', include('blog.urls'))
"""
from django.conf.urls import url, include
from django.contrib import admin
from labapp import views



urlpatterns = [
    url(r'^$', views.index2, name='index2'),
    url(r'^labs/', include('labapp.urls')),
    url(r'^admin/', admin.site.urls),
]
```

Labbapp/urls.py

```
from django.conf.urls import url

from . import views

urlpatterns = [
    url(r'^logout$', views.logout_view, name='logout'),
    url(r'^success_authorization_dumb$', views.success_authorization_dumb,
name='success_authorization_dumb'),
    url(r'^success_authorization$', views.success_authorization,
name='success_authorization'),
    url(r'^authorization/$', views.authorization, name='authorization'),
    url(r'^registration_dumb/$', views.registration_dumb,
name='registration_dumb'),
    url(r'^registration_user/$',views.registration_user,
name='registration_user'),
    url(r'^add_film/$', views.add_film, name='add_film'),
    url(r'^users/$',views.UserList.as_view(),name='user_list'),
    url(r'^films/$',views.FilmList.as_view(),name='film_list'),
    url(r'^reviews/$',views.ReviewList.as_view(),name='review_list'),
    url(r'^$', views.index, name='index'),
]
```

admin.py

```
from django.contrib import admin
from .models import *

# Register your models here.
@admin.register(Userus)
class UserAdmin(admin.ModelAdmin):
    #fields = ('first name', 'last name')
    list_display = ('username','full_name','age','has_reviews',)
    list_filter = ('age',)
    search_fields = ['last_name', 'first_name']

    def full_name(self, obj):
        return "{} {}".format(obj.last_name, obj.first_name)
```

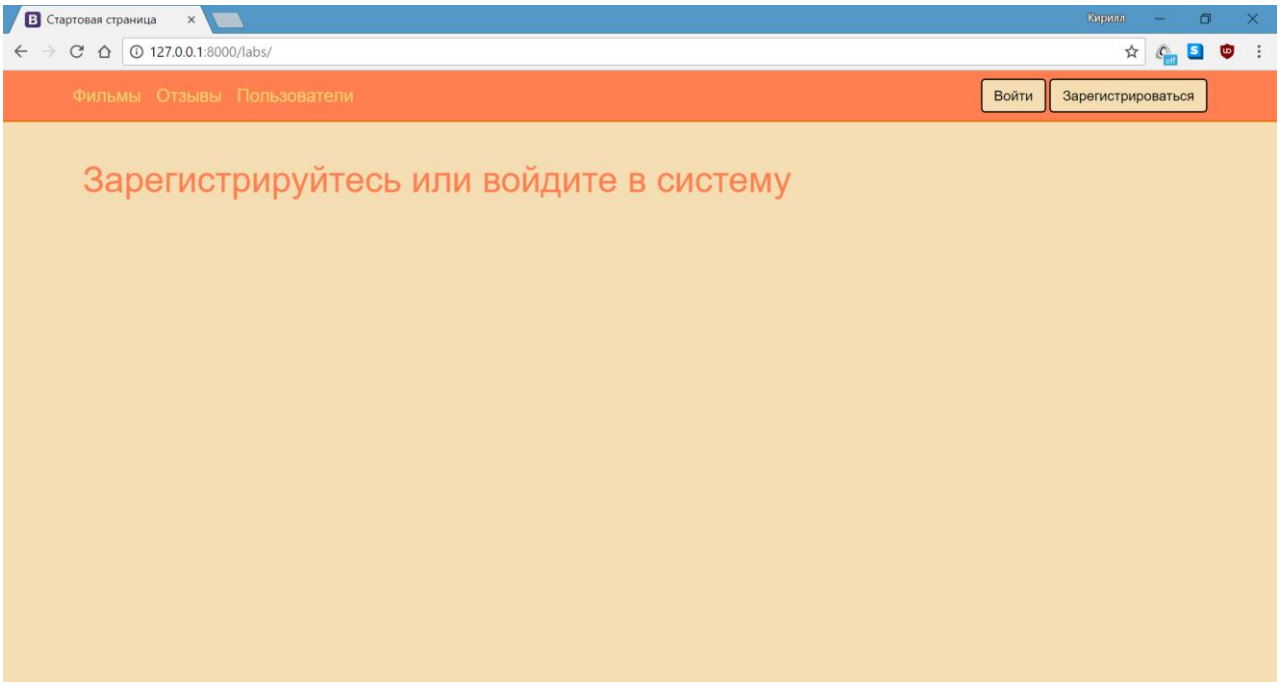
```
    def username(self, obj):
        return "{}".format(obj.user.username)

    def has_reviews(self, obj):
        hs = Review.objects.filter(user=obj)
        return len(hs)>0


@admin.register(Film)
class FilmAdmin(admin.ModelAdmin):
    empty_value_display = '-empty-'

@admin.register(Review)
class ReviewAdmin(admin.ModelAdmin):
    empty_value_display = '-empty-'
```

models.py

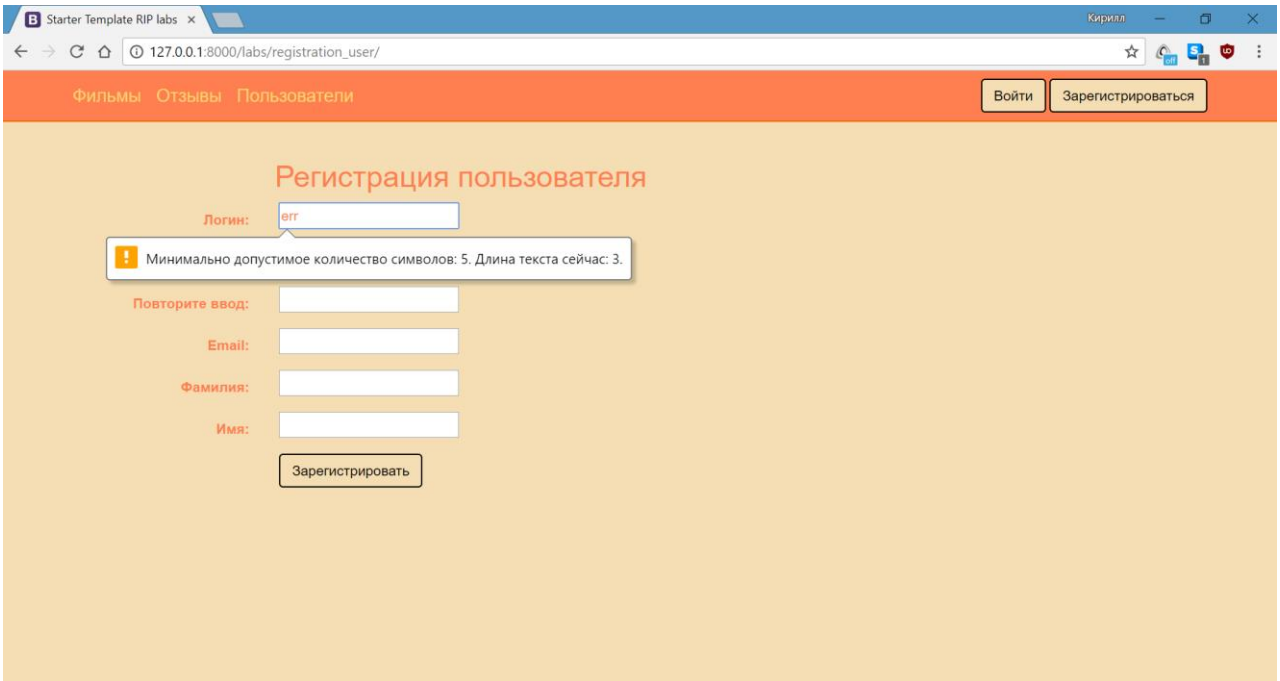
```
from django.db import models
from django.contrib.auth.models import User
from django.contrib import admin

# Create your models here.
class Userus(models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    first_name = models.CharField(max_length=30)
    last_name = models.CharField(max_length=30)
    age = models.IntegerField(null=True)
    nickname = models.CharField(max_length=30, null=True)


class Film(models.Model):
    # added_by = models.ForeignKey(Userus, on_delete=models.CASCADE)
    name = models.CharField(max_length=30)
    genre = models.CharField(max_length=30)
    description = models.CharField(max_length=255,null=True)

    objects = models.Manager()

class Review(models.Model):
    user = models.ForeignKey(Userus, on_delete=models.CASCADE)
    film = models.ForeignKey(Film, on_delete=models.CASCADE)
    rating = models.IntegerField()
    review_date = models.DateField()
    review_content = models.CharField(max_length=255, null=True)
```

Результаты:

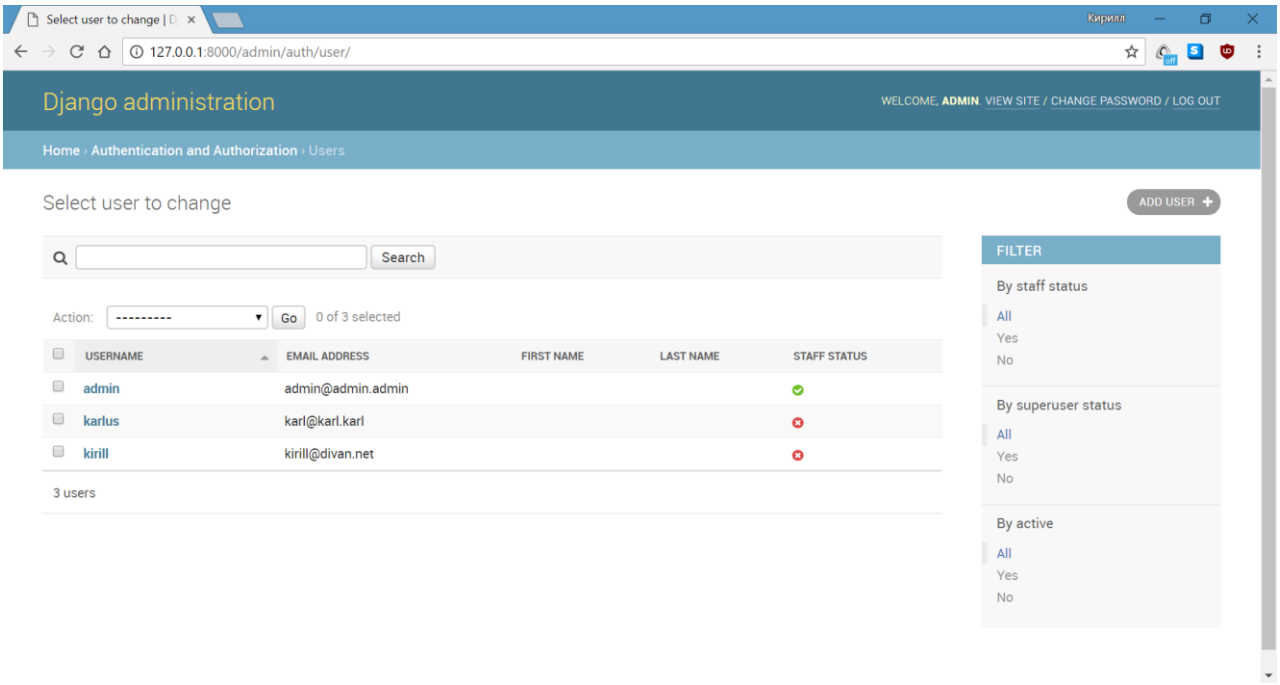
Стартовая страница для неавторизованных пользователей



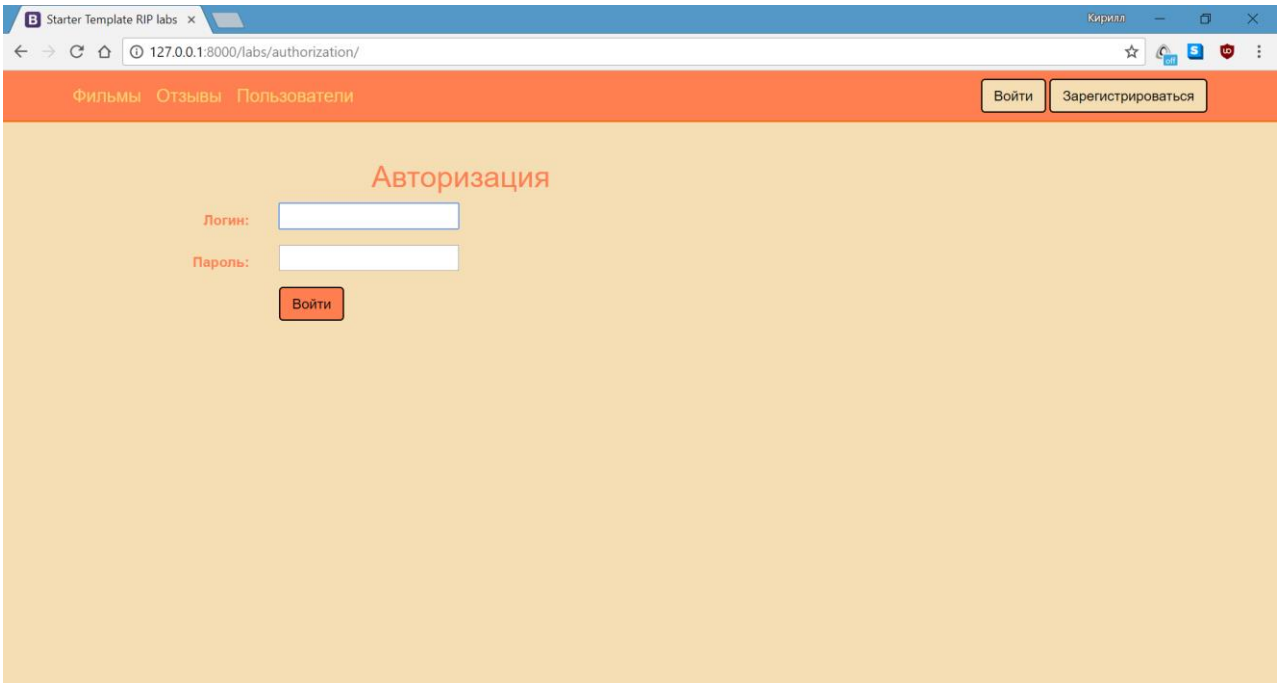
Страница регистрации



Меню администратора



Страница авторизации



Стартовая страница для авторизованных пользователей

Список фильмов