Московский государственный технический университет им. Н.Э.Баумана

Демонстрация ЛР:
Гапанюк. Ю.Е.

Отчет по лабораторной работе № 7 по курсу РИП

"Работа с формами, авторизация, django admin"

ИСПОЛНИТЕЛЬ:

студент группы **ИУ5-54**

Наседкин И.А.

Москва, МГТУ - 201**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

Текст программы

views.py

```
from django.http import HttpResponseRedirect
from django.shortcuts import render, redirect
from django.contrib.auth.decorators import login_required
from django.views import View

from polls.forms import LoginForm
from polls.models import *
from django import forms
from django.contrib.auth.hashers import make_password
from django.contrib.auth import authenticate,logout
from django.contrib import auth

# Create your views here.


class RegistrationForm(forms.Form):
    username = forms.CharField(
        widget=forms.TextInput(attrs={'class': 'form-control', 'id': 'username',
'placeholder': 'Введите логин', }), \
        min_length=5, label='Login:')
    name = forms.CharField(
        widget=forms.TextInput(attrs={'class': 'form-control', 'id': 'name',
'placeholder': 'Введите имя', }), \
        max_length=30, label='Name:')
    surname = forms.CharField(
        widget=forms.TextInput(attrs={'class': 'form-control', 'id': 'surname',
'placeholder': 'Введите фамилию', }), \
        max_length=30, label='Surname:')
    email = forms.EmailField(
        widget=forms.EmailInput(attrs={'class': 'form-control', 'id': 'email',
'placeholder': 'Введите email', })
    )
    password = forms.CharField(min_length=8, label='Password:',
widget=forms.PasswordInput(
        attrs={'class': 'form-control', 'id': 'password', 'placeholder': 'Введите
пароль', }))
    password2 = forms.CharField(min_length=8, label='Confirm password:',
widget=forms.PasswordInput(
        attrs={'class': 'form-control', 'id': 'password2', 'placeholder':
'Повторите пароль', }))

    def save(self):
        u = User()
        u.username = self.cleaned_data.get('username')
        u.password = make_password(self.cleaned_data.get('password'))
        u.first_name = self.cleaned_data.get('name')
        u.last_name = self.cleaned_data.get('surname')
        u.email = self.cleaned_data.get('email')
        u.is_staff = False
```

```
        u.is_active = True
        u.is_superuser = False
        u.save()

    def clean_password2(self):
        if self.cleaned_data.get('password') !=
self.cleaned_data.get('password2'):
            raise forms.ValidationError('Passwords does not match')

    def clean_username(self):
        username = self.cleaned_data.get('username')
        try:
            u = User.objects.get(username=username)
            raise forms.ValidationError('This login already uses')
        except User.DoesNotExist:
            return username
#########################################################################

def registration(request):
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('/')
        return render(request, 'signup.html', {'form': form})
    else:
        form = RegistrationForm()
    return render(request, 'signup.html', {'form': form})


def authorization(request):
    redirect_url = '/'
    if request.method == 'POST':
        redirect_url = '/'
        form = LoginForm(request.POST)
        if form.is_valid():
            user = auth.authenticate(username=form.cleaned_data['login'],
                                     password=form.cleaned_data['password'])
            if user is not None:
                auth.login(request, user)
                return HttpResponseRedirect(redirect_url)
            else:
                form.add_error(None, 'invalid login/password')
    else:
        form = LoginForm()
    return render(request, 'login.html', {'form':form, 'continue': redirect_url})


@login_required
def exit(request):
    logout(request)
    return render(request, 'logout.html')

class InitView(View):
    def get(self,request):
        return render(request,'init.html')
```

urls.py

```
__author__ = 'Work'


from django.conf.urls import url
from django.contrib import admin
from django.contrib.auth import views
```

```
from . import views



urlpatterns=[
    url(r'^admin/', admin.site.urls),
    url(r'^signup/', views.registration, name='signup'),
    url(r'^logout/', views.exit, name='logout'),
    url(r'^login/', views.authorization, name='login'),
    ##url(r'', views.logout_view, name='index'),
    url(r'',views.InitView.as_view(), name= 'init')
]
```

models.py

```
from django.db import models
from django.contrib.auth.models import AbstractUser


# Create your models here.

class User(AbstractUser):
    id = models.IntegerField(primary_key=True)
    name = models.CharField(max_length=30)
    phone = models.CharField(max_length=20)

    class Meta(AbstractUser.Meta):
        swappable = 'AUTH_USER_MODEL'


class BankModel(models.Model):
    idbank = models.IntegerField(primary_key=True)
    director = models.CharField(max_length=30)
    address = models.CharField(max_length=255)

    def __str__(self):
        return str(self.address)


class TransactionModel(models.Model):
    idtran = models.IntegerField(primary_key=True)
    sum = models.IntegerField()
    type = models.CharField(max_length=50)
    user = models.ForeignKey('User')
    bank = models.ForeignKey('BankModel')
    date = models.DateTimeField()

    def __str__(self):
        return str(self.sum)
```

admin.py

```
from django.contrib import admin
from .models import *
# Register your models here.

class BankAdmin(admin.ModelAdmin):
    list_display = ('idbank', 'address')
    search_fields = ('idbank', 'address')

admin.site.register(BankModel, BankAdmin)

class TranAdmin(admin.ModelAdmin):
    list_display = ('idtran', 'sum', 'type','user','bank','date')
    list_filter = ['sum']
    search_fields = ('id', 'user')
```

```
admin.site.register(TransactionModel, TranAdmin)

class UserAdmin(admin.ModelAdmin):
    list_display = ('username', 'first_name', 'last_name')

admin.site.register(User, UserAdmin)
```

forms.py

```
__author__ = 'Work'
from django import forms
from django.core.exceptions import ValidationError
from django.core.validators import validate_email
from django.contrib.auth.models import User


class LoginForm(forms.Form):
    login = forms.CharField(label='Логин')
    password = forms.CharField(label='Пароль', widget=forms.PasswordInput)


class SignupForm(forms.Form):
    login = forms.CharField(label='Логин', min_length=5)
    password = forms.CharField(label='Пароль', min_length=8,
widget=forms.PasswordInput)
    repeat_password = forms.CharField(label='Повторите пароль',
widget=forms.PasswordInput)
    email = forms.CharField(label='Адрес электронной Почты')
    first_name = forms.CharField(label='Имя')
    last_name = forms.CharField(label='Фамилия')

    def clean_login(self):
        login = self.cleaned_data['login']
        if User.objects.filter(username=login):
            raise ValidationError('Этот login уже занят')
        return login

    def clean_email(self):
        email = self.cleaned_data['email']
        validate_email( self.cleaned_data['email'])
        if User.objects.filter(email=email):
            raise ValidationError('Этот email уже зарегистрирован')
        return self.cleaned_data['email']

    def clean(self):
        cleaned_data = super(SignupForm, self).clean()
        if self.cleaned_data.get('password') and
self.cleaned_data.get('repeat_password'):
            if self.cleaned_data['password'] !=
self.cleaned_data['repeat_password']:
                raise ValidationError('Пароли не совпадают')
        return cleaned_data

    def save(self):
        user = User.objects.create_user(username=self.cleaned_data['login'],
                                        email=self.cleaned_data['email'],
                                        password=self.cleaned_data['password'],

first_name=self.cleaned_data['first_name'],
                                        last_name=self.cleaned_data['last_name'],
                                        )
        return user
```

init.html

```
<!DOCTYPE html>
<html>
<head lang="en">
    <meta charset="UTF-8">
    <title></title>
</head>
<body>
    {% if user.is_authenticated %}
        <p>Здраствуйте,{{user.get_username}}</p>
        <a href="{% url 'logout' %}">Выйти</a>
    {% else %}
        <p>Здраствуйте, Гость</p>
        <a href="{% url 'signup' %}">Зарегистрироваться</a>
        <a href="{% url 'login' %}">Войти</a>
    {% endif %}


</body>
</html>
```

login.html

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

<body>

{{ form.non_field_errors }}
<form action="/login/" method="post" class="form-horizontal">
                    {% csrf_token %}
                    {%  for i in form %}
                    <div class="form-group">
                        <label class="col-sm-2">{{ i.label }}</label>
                        <div class="col-sm-10">
                            <div>
                                {{ i }}
                            </div>

                        </div>
                    </div>
                    {% endfor %}

                    <div class="form-group">
                        <div class="col-sm-offset-2 col-sm-10">
                            <input type="submit" value="Войти" class="btn btn-
default"/>
                        </div>
                    </div>
</form>
<a href="{% url 'init' %}">На главную</a>


</body>
</html>
```

logout.html

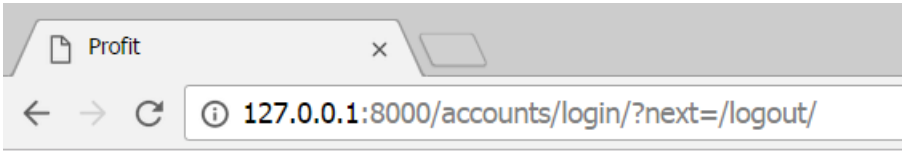
```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <title>Profit</title>
</head>
<body>
<a href="{% url 'logout' %}">Выход</a>
</body>
</html>
```

logout.html

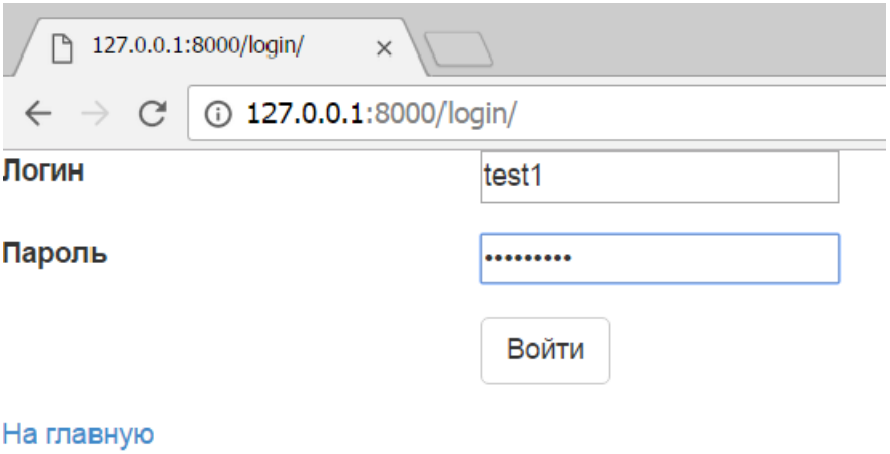
```
<html>
<head>
{% load static %}
<link rel="stylesheet"
href="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css">
</head>

<body>

{{ form.non_field_errors }}
<form action="/signup/" method="post" class="form-horizontal"
enctype="multipart/form-data">
                    {% csrf_token %}
                    {%  for i in form %}
                    <div class="form-group">
                        <label class="col-sm-4">{{ i.label }}</label>
                        <div class="col-sm-8">
                            <div>
                                {{ i }}
                            </div>
                            <div>
                                    {{ i.errors }}
                            </div>

                        </div>
                    </div>
                    {% endfor %}

                    <div class="form-group">
                        <div class="col-sm-offset-4 col-sm-8">
                            <input type="submit" value="Зарегистрировать меня"
class="btn btn-default"/>
                        </div>
                    </div>
</form>
</body>
</html>
```

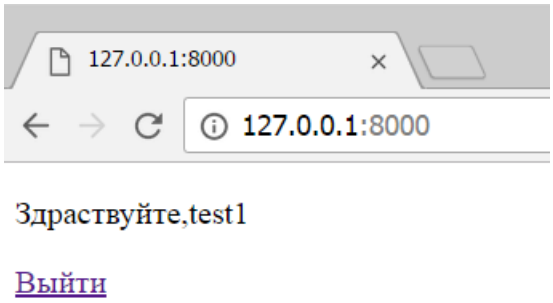
Результаты работы программы

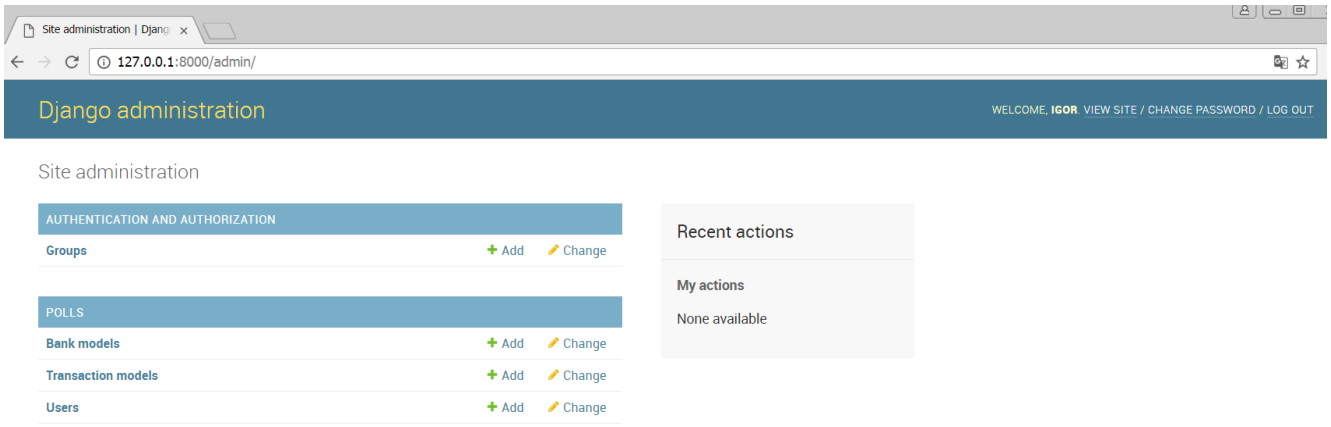


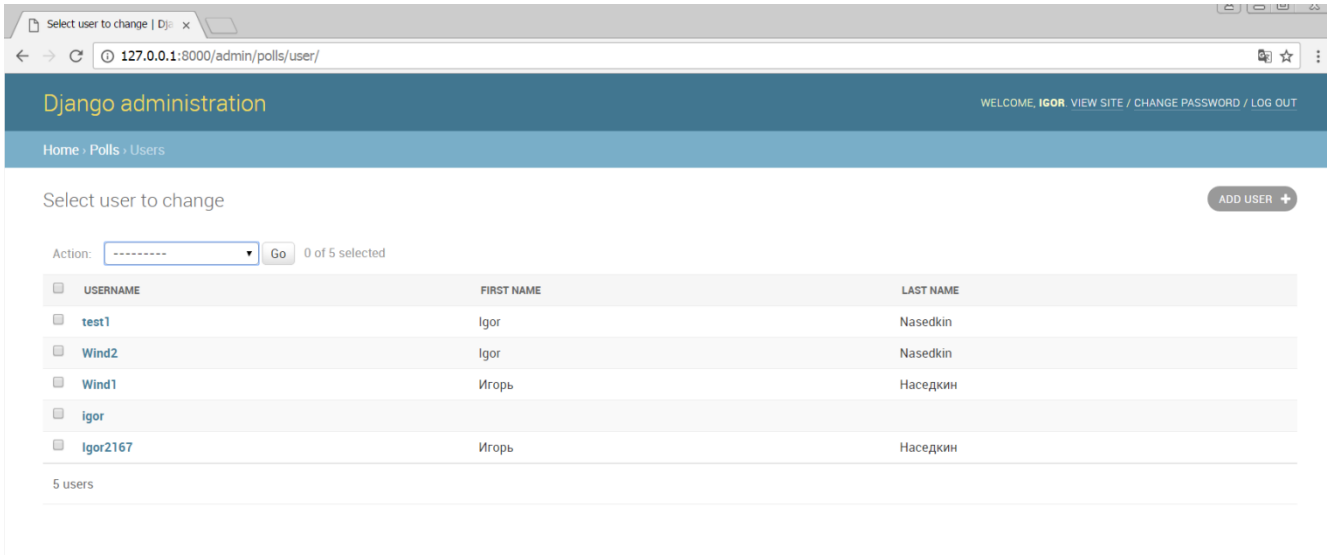
Здраствуйте, Гость

Зарегистрироваться Войти


127.0.0.1:8000/login/
127.0.0.1:8000/login/
Логин
test1
Пароль
Войти
На главную


127.0.0.1:8000
127.0.0.1:8000
Здраствуйте,test1
Выйти


Site administration | Djang
127.0.0.1:8000/admin/
Django administration
WELCOME, IGOR. VIEW SITE / CHANGE PASSWORD / LOG OUT
Site administration
AUTHENTICATION AND AUTHORIZATION
Groups
+ Add
Change
POLLS
Bank models
+ Add
Change
Transaction models
+ Add
Change
Users
+ Add
Change
Recent actions
My actions
None available


Select user to change | Dj
127.0.0.1:8000/admin/polls/user/
Django administration
WELCOME, IGOR. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Polls › Users
Select user to change
ADD USER
Action: ---------
Go
0 of 5 selected
USERNAME
FIRST NAME
LAST NAME
test1
Igor
Nasedkin
Wind2
Igor
Nasedkin
Wind1
Игорь
Наседкин
igor
Igor2167
Игорь
Наседкин
5 users