Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «РИП»

Отчет по лабораторной работе №7

«Работа с формами, авторизация, django admin»

Выполнил:
студент группы ИУ5-53
Гатауллин И. И.
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю.Е.
Подпись и дата:

Москва, 2017 г.

Задание и порядок выполнения ЛР №7

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

Код программы

forms.py

```
import django.contrib.auth.models as authmodel
from django.contrib.auth.forms import AuthenticationForm, UserCreationForm
from django.forms import *
from .models import *


class Login(forms.Form):
    username = forms.CharField(min_length=4, label='Логин',required=True)
    password = forms.CharField(min_length=6, widget=forms.PasswordInput,
label='Пароль')


class RegistrationForm(forms.Form):
    username = forms.CharField(min_length=5, label='Login')
    password = forms.CharField(min_length=8, widget=forms.PasswordInput,
label='Password')
    password2 = forms.CharField(min_length=8, widget=forms.PasswordInput,
label='Repeat Password')
    # email = forms.EmailField(label='Email')
    # first_name = forms.CharField(label='First name')
    # last_name = forms.CharField(label='Last name')
```

models.py

```
from django.db import models
from django import forms
from datetime import date
from django.contrib.auth.models import User
from django.core.validators import MaxValueValidator, MinValueValidator

# Create your models here.

class CustomUser(User):
    pass


class Service(models.Model):
    title = models.CharField(max_length=254)
    category = models.CharField(max_length=254)
    company = models.CharField(max_length=254)
    price = models.DecimalField(max_digits=10, decimal_places=2, default=0)

    class Meta:
        unique_together = ('title', 'category')


class Order(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    service = models.ForeignKey(Service, on_delete=models.CASCADE)
    odate = models.DateField
```

views.py

```
from django.http import HttpResponseRedirect
from django.shortcuts import render, redirect
from django.views import View
from django.views.generic import ListView, FormView
from django.contrib import auth
from django.contrib.auth.models import User
from django.contrib.auth.forms import UserCreationForm
from services.forms import Login, RegistrationForm
from services.models import Service, Order


def services_list(request):
    return render(request, 'services/services_list.html', {})


class MainPageView(View):
    def get(self, request):
        data = Service.objects.all()
        return render(request, 'services/index.html', {'top_services': data,
'username': auth.get_user(request).username})


class ServicePageView(View):
    def get(self, request, id):
        data = Service.objects.get(pk=id)
        orders = Order.objects.filter(service=id)

        return render(request, 'services/service.html',
                      {'username': auth.get_user(request).username,
'service': data, 'orders': orders})


class ServiceList(ListView):
    model = Service
    template_name = 'services/services_list.html'
    context_object_name = 'services'


def login(request):
    args = {}
    args['form'] = Login()
    if request.method == 'POST':
        username = request.POST.get('username', '')
        password = request.POST.get('password', '')
        user = auth.authenticate(username=username, password=password)
        if user is not None:
            auth.login(request, user)
            return redirect('/')
        else:
            args['login_error'] = 'Пользователь не найден'
            return render(request, 'services/login.html', args)
    else:
        return render(request, 'services/login.html', args)


def logout(request):
    auth.logout(request)
    return redirect('/')
```

```
class UserCreateForm(object):
    pass


class SignUp(FormView):
    template_name = 'services/signin.html'
    form_class = UserCreateForm
    success_url = '/'


def signUp(request):
    # form = None;
    errors = []
    success = ''
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        print(form.is_valid())
        if form.is_valid():
            username = form.cleaned_data['username']

            users = User.objects.all()
            usernames = []
            for x in users:
                usernames.append(x.username)

            if form.cleaned_data['password'] !=
form.cleaned_data['password2']:
                errors.append('Пароли должны совпадать')
            elif usernames.count(username) != 0:
                errors.append('Такой логин уже занят')
            else:
                print("User")
                user = User.objects.create_user(
                    username=form.cleaned_data['username'],
                    # email=form.cleaned_data['email'],
                    password=form.cleaned_data['password'],
                    # first_name=form.cleaned_data['first_name'],
                    # last_name=form.cleaned_data['last_name']
                )
                user.save()
                success += 'You was successfully registered.'
                return HttpResponseRedirect('/login/')

    else:
        form = RegistrationForm()
    # form = RegistrationForm(request.POST)
    return render(request, 'services/signin.html', {'form': form, 'errors':
errors, 'success': success})
```

admin.py

```
from django.contrib import admin

from .models import CustomUser, Service, Order

class ServiceAdmin(admin.ModelAdmin):
    list_display = ('title', 'category', 'company', 'price')
    list_filter = ('company', 'title')
    search_fields = ('company', 'title')
```

```
class OrderAdmin(admin.ModelAdmin):
    list_display = ('user', 'service')
    list_filter = ('user', 'service')
    search_fields = ('user', 'service')


admin.site.register(Service, ServiceAdmin)
admin.site.register(Order, OrderAdmin)
```

auth_base.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="/static/css/bootstrap.css" rel="stylesheet">
    <link href="/static/css/sstyles.css" rel="stylesheet">
    <title>{% block title %}{% endblock %}</title>
</head>
<body>
<div class="container-fluid">
    <div class="row">
        <div class="col-md-12">
            <h1><a href="/"> SServices </a></h1>
        </div>
    </div>
        <div>{% block body %}{% endblock %}</div>
    </div>

</div>
</body>
</html>
```

login.html

```
{% extends 'services/auth_base.html' %}
{% block title %}
    Вход
{% endblock %}

{% block body %}
    <h3>Вход</h3>
    <form method="POST">
        {% csrf_token %}
        {{ form.as_ul }}
        {{ login_error }}
        <button type="submit">Войти</button>
    </form>
    <form action="/sign_in/">
        <input type="submit" value="Регистрация"/>
    </form>
{% endblock %}
```

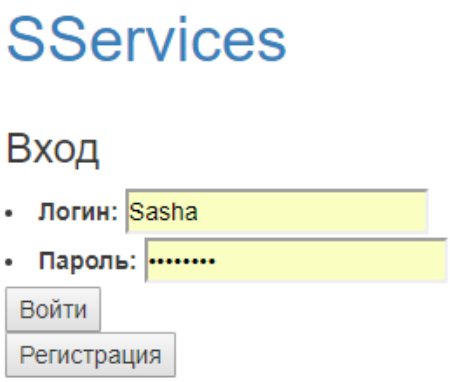
signin.html

```
{% extends 'services/auth_base.html' %}
{% block title %}Регистрация{% endblock %}
{% block body %}
```

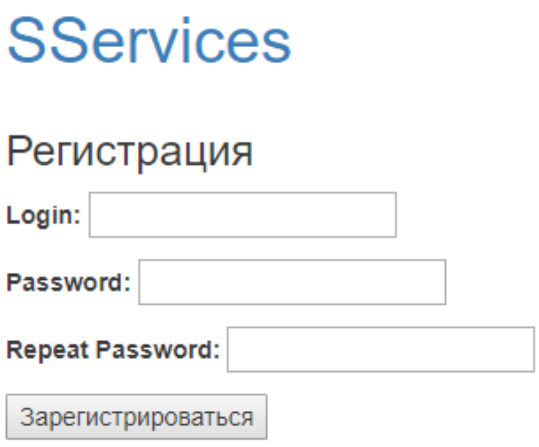
```
    <h3>Регистрация</h3>
    <form method="POST">
            {% csrf_token %}
            {{ form.as_p }}
            <button type="submit">Зарегистрироваться</button>
        </form>
{% endblock %}
```

Скрины выполнения

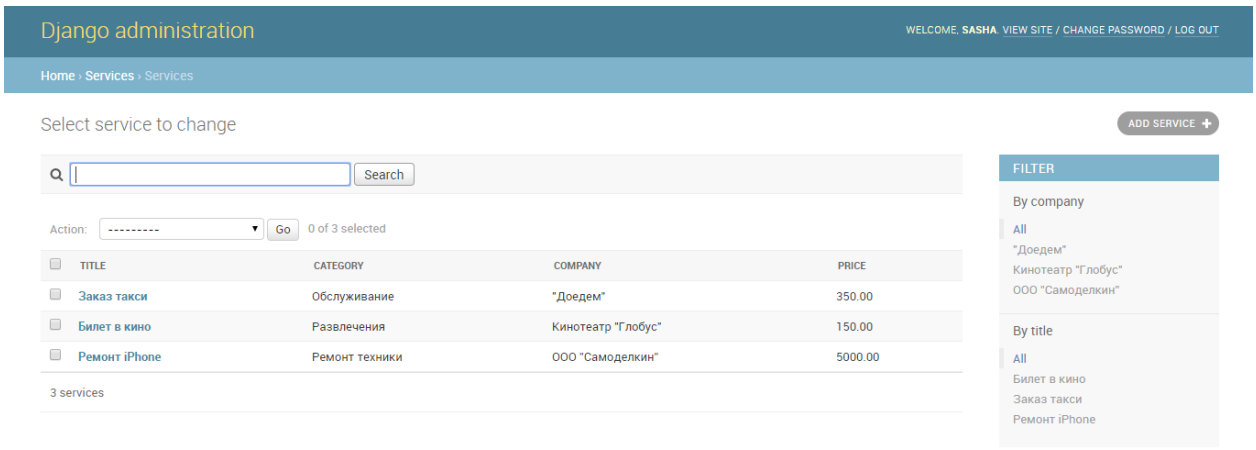
Авторизация



Регистрация



Настроенная админка


Django administration
WELCOME, SASHA. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Services › Services
Select service to change
ADD SERVICE
Search
Action: --------- Go 0 of 3 selected
TITLE
CATEGORY
COMPANY
PRICE
Заказ такси
Обслуживание
"Доедем"
350.00
Билет в кино
Развлечения
Кинотеатр "Глобус"
150.00
Ремонт iPhone
Ремонт техники
ООО "Самоделкин"
5000.00
3 services
FILTER
By company
All
"Доедем"
Кинотеатр "Глобус"
ООО "Самоделкин"
By title
All
Билет в кино
Заказ такси
Ремонт iPhone