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



Отчет по лабораторной работе №7
«Работа с формами, авторизация, django admin»
По курсу "Разработка интернет-приложений"

Студент группы ИУ5-51 Жизневский П.И.

#### 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.

- 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
class TeacherList(LoginRequiredMixin, ListView):
   context object name = 'teachers'
    login url = '/labs/'
    redirect field name = 'redirect to'
   model = Teacher
   template name = 'teacher list.html'
class PulpitList(ListView):
   context_object name = 'pulpits'
   model = Pulpit
   template_name = 'pulpit_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:</pre>
            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:</pre>
            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')
            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='Mmg')
```

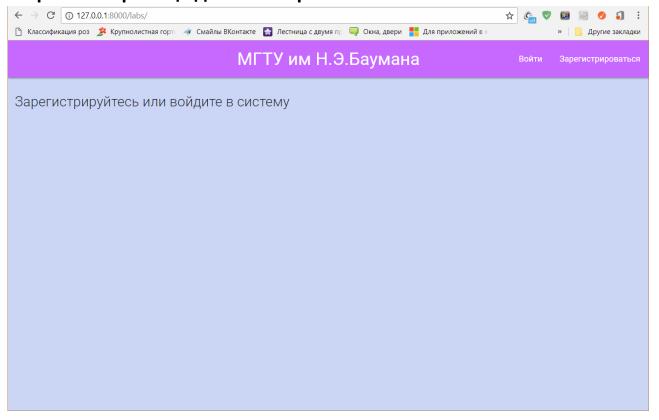
```
def registration user(request):
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        is val = form.is valid()
        data = form.cleaned data
        if data['password']!=data['password2']:
            is val = False
            form.add error('password2',['Пароли 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.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.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:</pre>
            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:</pre>
            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, 'test.html')
Urls.py
from django.conf.urls import url, include
from django.contrib import admin
from labapp.views import index
#from labapp.views import home, labapp
urlpatterns = [
    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'^teachers/$', views.TeacherList.as view(), name='teacher list'),
    url(r'^pulpit/$', views.PulpitList.as_view(), name='pulpit_list'),
    # url(r'^reviews/$',views.ReviewList.as view(),name='review list'),
    url(r'^$', views.index, name='index'),
]
```

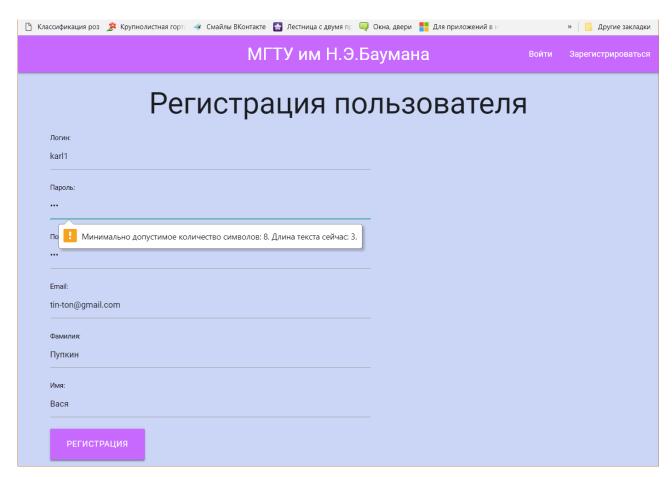
#### admin.py

```
from django.contrib import admin
from .models import *
# Register your models here.
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)
@admin.register(Pulpit)
class PulpitAdmin(admin.ModelAdmin):
    empty value display = '-empty-'
@admin.register(Teacher)
class PulpitAdmin(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 Teacher(models.Model):
    name = models.CharField(max length=30)
    second name = models.CharField(max length=30)
    third name = models.CharField(max length=30)
    phone = models.BigIntegerField()
    mail = models.CharField(max length=30)
class Pulpit(models.Model):
    name = models.CharField(max length=5)
    year = models.IntegerField()
    objects = models.Manager()
class Membership(models.Model):
    teacher = models.ForeignKey(Teacher, on delete=models.CASCADE)
    pulpit = models.ForeignKey(Pulpit, on delete=models.CASCADE)
```

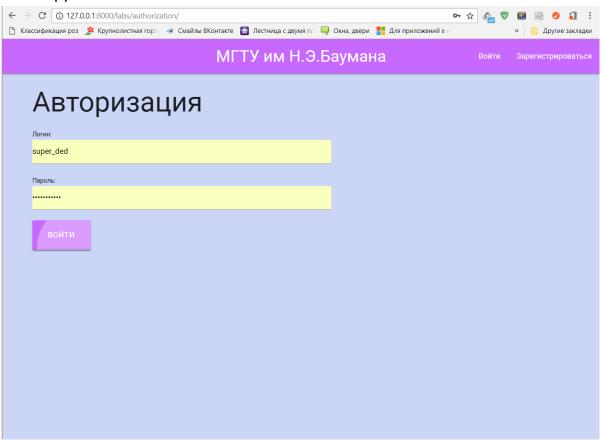
## Стартовая страница для неавторизованных пользователей



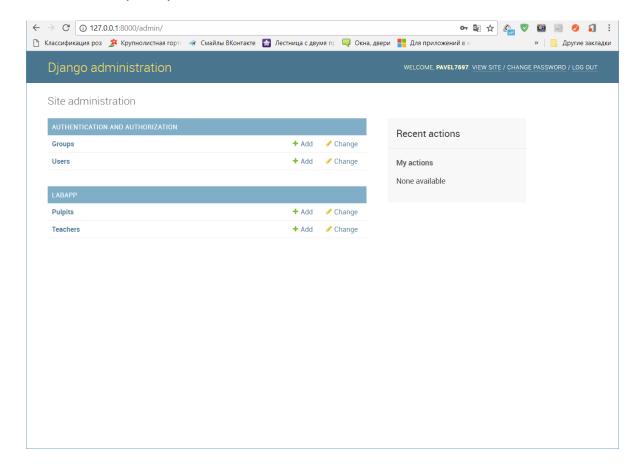
### Страница регистрации



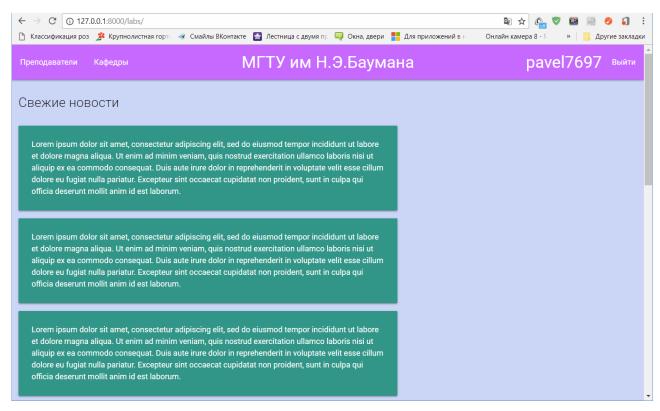
## Форма входа



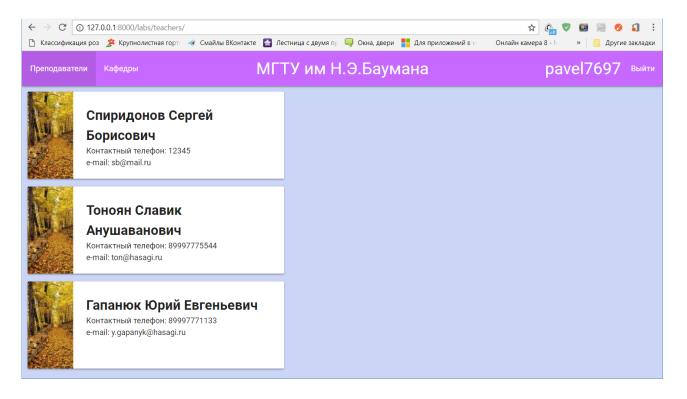
# Меню администратора



## Стартовая страница для авторизованных пользователей



#### Список преподавателей



# Список кафедр

