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

Факультет «Системы обработки информации и управления»

Кафедра ИУ5. Курс «РИП»

Отчет по лабораторной работе №7

«Авторизация, работа с формами и Django Admin.»

Выполнил:
студент группы ИУ5-53
Белков А.Д.

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’a 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

models.py

```python
from django.db import models
from django.contrib.auth.models import User
# -*- coding: utf-8 -*-


class Product(models.Model):
    class Meta:
        db_table = 'lab_app_product'

    # Название товара
    name = models.CharField(max_length=255)
```

```
    # Описание товара
    description = models.CharField(max_length=1000)

    # Продавец
    seller = models.CharField(max_length=255)

    # Ссылка на картинку товара
    image = models.ImageField(upload_to='lab_app/static/product_images',

default='lab_app/static/product_images/default.png')

    # Полный путь до картинки товара
    def image_path(self):
        return self.image.name.replace('lab_app/', '')

    # Короткое описание товара
    def short_description(self):
        return self.description[:126]

    def __str__(self):
        return ' '.join([
            self.name,
            ' from ',
            self.seller,
        ])


class Review(models.Model):
    class Meta:
        db_table = 'lab_app_review'

    # Пользователь, который оставил отзыв
    user = models.ForeignKey(User, on_delete=models.CASCADE)

    # Товар, под которым оставлен отзыв
    product = models.ForeignKey(Product, on_delete=models.CASCADE)

    # Текст отзыва
    description = models.CharField(
        max_length=500,
    )

    def __str__(self):
        return ' '.join([
            'review \'',
            str(self.description),
            ' \' from user @',
            str(self.user.username),
        ])
```

urls.py

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

from lab_app.views import ProductView, ListProductView, AddProductView, \
    SignUpView, LoginView, LogoutView, create_review, AboutView

urlpatterns = [
    url(r'^$', ListProductView.as_view()),
    url(r'^page=(?P<page>\d+)', ListProductView.as_view()),
    url(r'^product/(?P<product_id>\d+)', ProductView.as_view()),
```

```
    url(r'^product/create_review/$', create_review, name='create_review'),
    url(r'^product/add_product/$', AddProductView.as_view()),
    url(r'^signup/$', SignUpView.as_view()),
    url(r'^login/$', LoginView.as_view()),
    url(r'^logout/$', LogoutView.as_view()),
    url(r'^about/', AboutView.as_view()),
    url(r'^admin/', admin.site.urls),
]
```

views.py

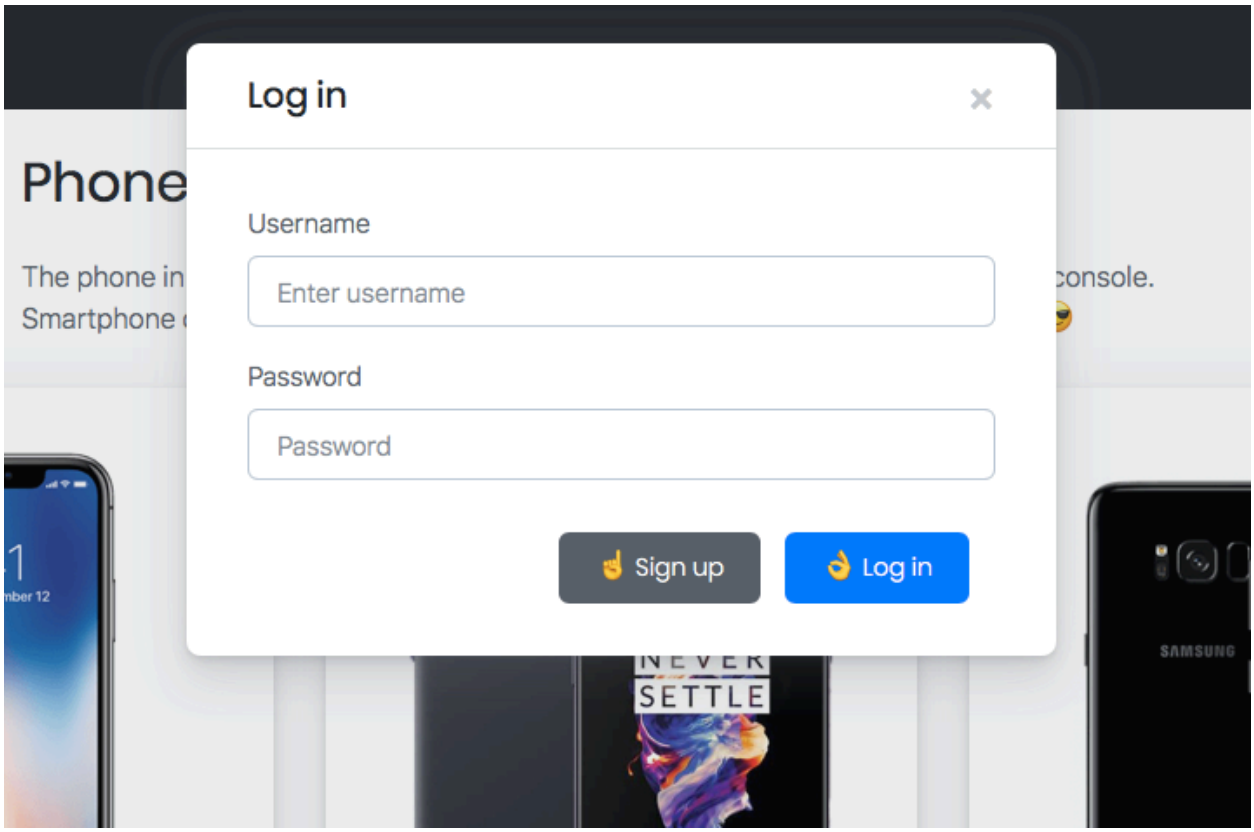
```
from django.contrib.auth import authenticate
from django.contrib.auth.models import User
from django.contrib.auth.views import logout, login
from django.shortcuts import render, redirect

from django.http import HttpResponse
from django.views.generic import View, ListView

from lab_app.models import Product, Review

from django.views.decorators.csrf import csrf_exempt
import json
import math

# TODO: Добавить проверку на superuser для отображения кнопки 'Добавить
продукт'


# Список продуктов
class ListProductView(ListView):

    model = Product
    template_name = 'product_list.html'
    context_object_name = 'products'
    paginate_by = 3

    def get(self, request, page=1):

        # Количество продуктов на странице
        elements_on_page = 9

        # Количество продуктов в строке
        elements_in_row = 3

        products = Product.objects.all()
        pages_count = math.ceil(len(products) / elements_on_page)

        start_index = (int(page) - 1)*elements_on_page
        end_index = start_index + elements_on_page
        products = products[start_index:end_index]

        index = 1
        rows = []
        row = []
        for product in products:
            row.append(product)

            if index == elements_in_row:
                rows.append(row)
                row = []
                index = 1
```

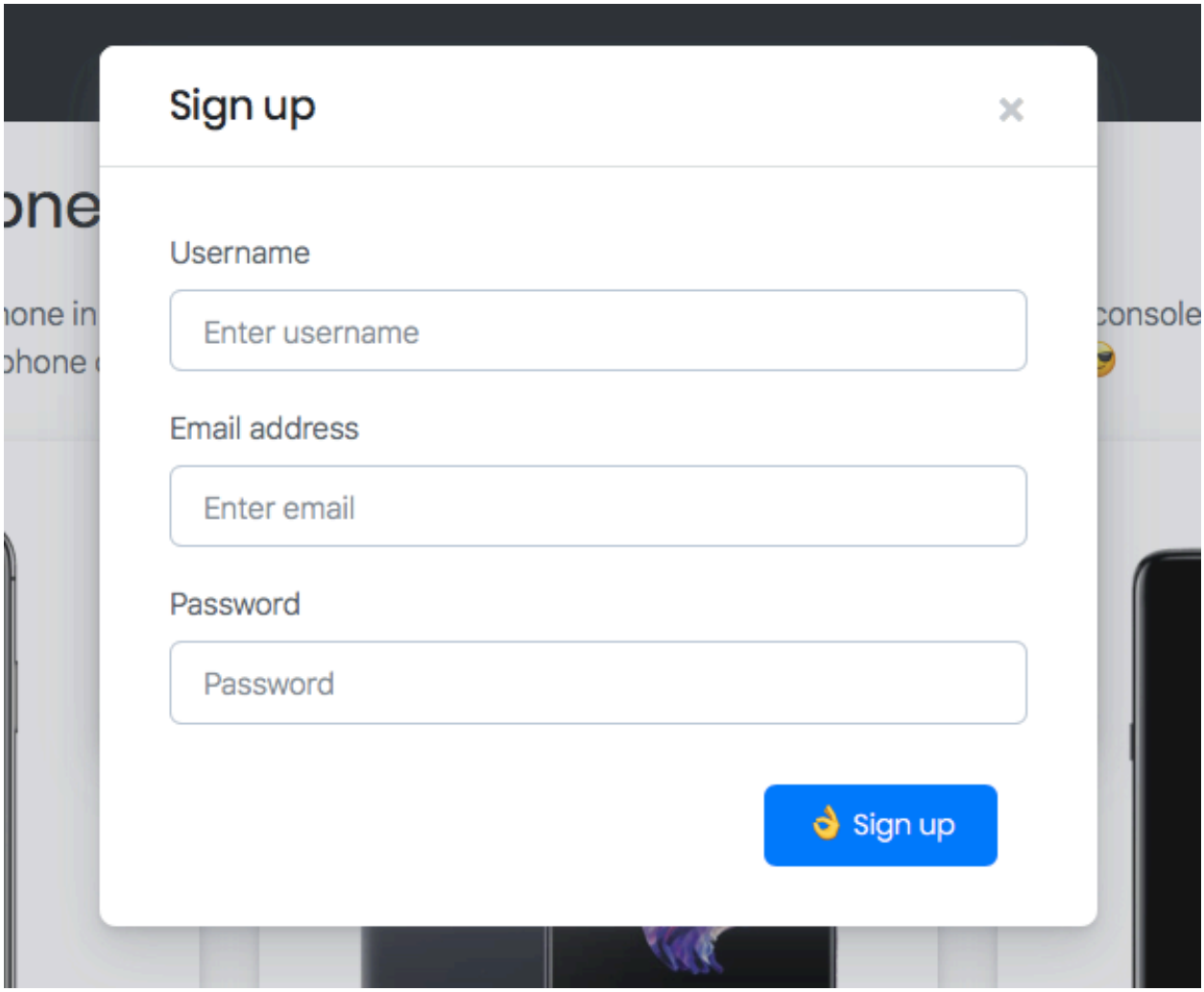
```
            else:
                index += 1

        if len(row) > 0:
            rows.append(row)

        return render(request, 'product_list.html',  {"products": rows,
"page": page, "pages_count": pages_count})


# Страница добавления продукта
class AddProductView(View):

    def post(self, request):
        if request.POST:
            name = request.POST['productName']
            description = request.POST['productDescription']
            seller = request.POST['productSeller']
            image = request.FILES['productImage']

            product = Product(name=name, description=description,
seller=seller, image=image)
            product.save()
            if product is not None:
                return redirect("/")

        return redirect("/invalidProduct")


# Страница с информацией о продукте и отзывами
class ProductView(View):

    def get(self, request, product_id):

        elements_in_row = 2
        product = Product.objects.get(id=product_id)
        reviews = Review.objects.filter(product_id=product_id)
        reviews_count = len(reviews)

        index = 1
        rows = []
        row = []
        for review in reviews:
            row.append(review)

            if index == elements_in_row:
                rows.append(row)
                row = []
                index = 1
            else:
                index += 1

        if len(row) > 0:
            rows.append(row)

        if len(rows) == 0:
            rows = None

        return render(request, 'product.html',  {"product": product,
"reviews": rows, "reviews_count": reviews_count})
```

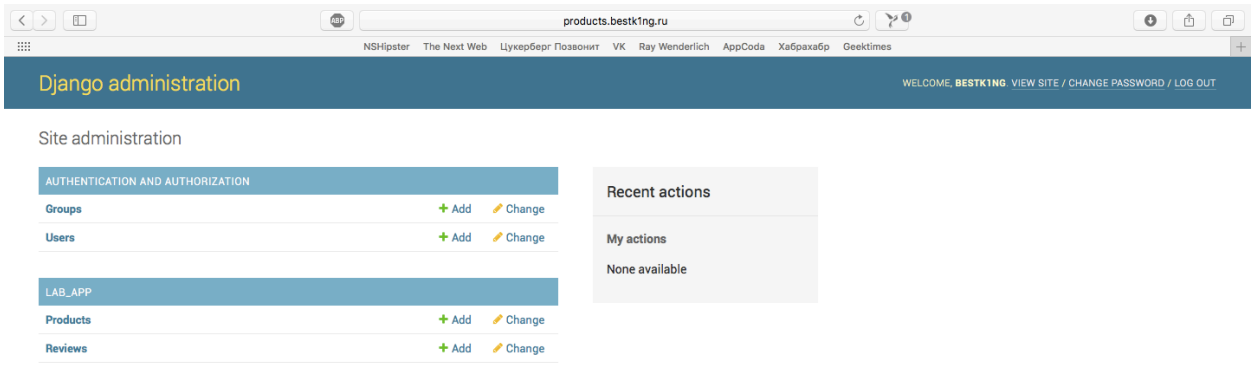
```python
# Страница регистрации
class SignUpView(View):

    def post(self, request):
        logout(request)
        if request.POST:
            user = User.objects.create_user(
                username=request.POST['username'],
                password=request.POST['password'],
                email=request.POST['email']
            )

            if user is not None:
                login(request, user)
                return redirect("/")

        return redirect("/invalidUser")


# Страница авторизации
class LoginView(View):

    def post(self, request):
        logout(request)
        if request.POST:
            username = request.POST['username']
            password = request.POST['password']

            user = authenticate(username=username, password=password)
            if user is not None:
                login(request, user)
                return redirect("/")

        return redirect("/invalidUser")


# Страница выхода
class LogoutView(View):

    def post(self, request):
        logout(request)
        return redirect("/")


# Создание и сохранение отзыва
@csrf_exempt
def create_review(request):
    if request.method == 'POST':
        review_text = request.POST.get('review_text')
        product_id = request.POST.get('product_id')

        # Создаем отзыв и сохраняем в БД
        user = User.objects.get(id=request.user.id)
        product = Product.objects.get(id=product_id)
        review = Review(description=review_text, user=user, product=product)
        review.save()

        # Формируем json с отзывом для обновления страницы
        response_data = dict()
        response_data["review_description"] = review.description
        response_data["product_id"] = review.product_id
        response_data["user_name"] = review.user.username
```

```
        response_data["reviews_count"] =
int(request.POST.get('reviews_count')) + 1

        return HttpResponse(
            json.dumps(response_data),
            content_type="application/json"
        )
    else:
        return HttpResponse(
            json.dumps({"nothing to see": "this isn't happening"}),
            content_type="application/json"
        )


class AboutView(View):

    def get(self, request):

        # ..

        return render(request, 'about.html')
```

Скришоты выполнения




Sign up
Username
Enter username
Email address
Enter email
Password
Password
Sign up


products.bestk1ng.ru
NSHipster
The Next Web
Цукерберг Позвонит
VK
Ray Wenderlich
AppCoda
Хабрахабр
Geektimes
Django administration
WELCOME, BESTK1NG. VIEW SITE / CHANGE PASSWORD / LOG OUT
Site administration
AUTHENTICATION AND AUTHORIZATION
Groups + Add Change
Users + Add Change
LAB_APP
Products + Add Change
Reviews + Add Change
Recent actions
My actions
None available