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

Гапанюк Ю.Е.

Отчет по Лабораторной работе №7 «Авторизация, работа с формами и Django Admin»

По курсу
“Разработка интернет-приложений”

Выполнил:
Постникова М.А.
Студент группы ИУ5-54

Москва 2017

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

Исходный код

ask/forms.py

```
# coding=utf-8
from django import forms
from django.core.validators import RegexValidator
from django.contrib.auth.models import User
_alphanumeric_validator = RegexValidator(r'^[0-9a-zA-Z_]*$',
                                         "Only alphabetic symbols, numbers and underscores
allowed")


# ENABLED FORMS
class RegisterForm(forms.Form):
    username = forms.CharField(max_length=150, widget=forms.TextInput(
        attrs={'class': 'form-control',
               'placeholder': 'Name, displayed to other users'}
    ), validators=[_alphanumeric_validator])
    email = forms.EmailField(max_length=254, widget=forms.EmailInput(
        attrs={'class': 'form-control',
               'placeholder': 'your@email.com'}
    ))
    password = forms.CharField(max_length=128, widget=forms.PasswordInput(
        attrs={'class': 'form-control',
               'placeholder': '**********'}
    ))
```

```
    confirm_password = forms.CharField(max_length=128,
widget=forms.PasswordInput(
        attrs={'class': 'form-control',
               'placeholder': '**********'}
    ))



    def clean(self):
        # unique email validation
        cleaned_data = super(RegisterForm, self).clean()
        if 'email' in cleaned_data:
            email = cleaned_data["email"]
            users_with_email = User.objects.filter(email=email)
            if len(users_with_email) > 0:
                raise forms.ValidationError("User with same email already registered")


        # unique username validation
        if 'username' in cleaned_data:
            username = cleaned_data["username"]
            users_with_username = User.objects.filter(username=username)
            if len(users_with_username) > 0:
                raise forms.ValidationError("User with same username already
registered")


        # password confirmation validation
        if 'password' in cleaned_data:
            password = cleaned_data["password"]
            confirm_password = cleaned_data["confirm_password"]
            if not password == confirm_password:
                raise forms.ValidationError("Passwords do not match")
            return cleaned_data


    def save(self):
        new_profile = User(username=self.cleaned_data["username"],
```

```
                        email=self.cleaned_data["email"],
                        )
        new_profile.set_password(self.cleaned_data["password"])
        new_profile.save()
        return new_profile


class LoginForm(forms.Form):
    username = forms.CharField(max_length=150, widget=forms.TextInput(
        attrs={'class': 'form-control',
               'placeholder': 'Name, displayed to other users'}
    ), validators=[_alphanumeric_validator])
    password = forms.CharField(max_length=128, widget=forms.PasswordInput(
        attrs={'class': 'form-control',
               'placeholder': '***********'}
    ))
```

base.html

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Мой блог</title>
    <!-- Bootstrap core CSS -->

    <link href="{% static "css/bootstrap.css" %}" rel="stylesheet">
    <!-- Custom styles for this template -->
    <link href="{% static "css/common.css" %}" rel="stylesheet">
</head>
<body class="bg-color">
    <nav class="navbar navbar-expand-md navbar-dark navbar-color fixed-top">
```

```
    <a class="navbar-brand" href="{% url "index" %}"><img class="logo"
src="/static/img/logo.png"></a>

    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExampleDefault" aria-controls="navbarsExampleDefault" aria-
expanded="false" aria-label="Toggle navigation">

      <span class="navbar-toggler-icon"></span>

    </button>

 {% if request.user.is_authenticated %}

      {% block user %}


         <div class="nav-account dropdown">

            <a href="#" class="hello2">

               Hello, {{ user.username }}

            </a>


                <a href="{% url 'logout' %}?continue={{ request.path }}"
class="hello3">Log out</a>


         </div>


      {% endblock %}

   {% else %}

      <a href="{% url 'login' %}?continue={{ request.path }}" class="hello2">

         Log in

      </a>


      <a href="{% url 'singup' %}?continue=/" class="hello3">

         Register

      </a>

   {% endif %}

   </nav>

<!-- /.container -->

   {% block content %}

   {%  endblock %}

<footer class="footer">
```

```
    <hr>
    <div class="row">
    <div class="col-md-9">
    &copy; Постникова Мария АПО-13
        </div>
    <div class="col-md-3 social">
        <a href="http://vk.com"><img src="/static/img/vk.png"></a>
        <a href=#><img src="/static/img/instagram.png"></a>
        <a href="#"><img src="/static/img/twitter.png"></a>
        <a href="#"><img src="/static/img/google-plus.png"></a>
        <a href="#"><img src="/static/img/facebook.png"></a>
        </div>
        </div>
</footer>
<script type="text/javascript" src="/static/js/jquery-3.2.1.slim.min.js"></script>
<script type="text/javascript" src="/static/js/popper.min.js"></script>
<script type="text/javascript" src="/static/js/bootstrap.js"></script>
</body>
</html>
```

singup.html

```
{% extends "base.html" %}
{% block content %}
<div class="article-body">
<h2 class="singup">Регистрация</h2>
    {% if errors %}
        {% for error in errors %}
            <div class="error">
                {{error}}
            </div>
        {% endfor %}
    {% endif %}
```

```
    <form method="POST">
        {% csrf_token %}
        <div class="content">
        <table align="center">
            <tr>
                <td>
                    <label>
                        Имя пользователя (логин):
                    </label>
                </td>
                <td>
                    <input type="text" name="username"
value="{{formdata.username}}">
                </td>
            </tr>
            <tr>
                <td>
                    <label>
                        Электронная почта:
                    </label>
                </td>
                <td>
                    <input type="text" name="email" value="{{formdata.email}}">
                </td>
            </tr>
            <tr>
                <td>
                    <label>
                        Имя:
                    </label>
                </td>
                <td>
```

```
                <input type="text" name="firstname"
value="{{formdata.firstname}}">
            </td>
        </tr>
        <tr>
            <td>
                <label>
                    Фамилия:
                </label>
            </td>
            <td>
                <input type="text" name="lastname"
value="{{formdata.lastname}}">
            </td>
        </tr><tr>
            <td>
                <label>
                    Пароль:
                </label>
            </td>
            <td>
                <input type="text" name="password"
value="{{formdata.password}}">
            </td>
        </tr><tr>
            <td>
                <label>
                    Введите пароль ещё раз:
                </label>
            </td>
            <td>
                <input type="text" name="confirmpass">
            </td>
        </tr>
```

```
      </table>


         <button action="submit" class="btn btn-primary btn-singup-
pos">Зарегистрироваться</button>
         </div>


   </form>


   </div>


{% endblock %}
```

login.html

```
{% extends "base.html" %}
{% block content %}
   <h2>Log In</h2>
   <div class="question-single">
      <form method="POST">
         {% csrf_token %}
         {{ form.as_table }}
         <button class="btn btn-primary btn-top" type="submit">Log in</button>
      </form>
      <p></p>
      <a href="{% url 'singup' %}">New here? Create an account.</a>
   </div>
{% endblock %}
```

ask/urls.py

```
from django.conf.urls import url
from . import views

urlpatterns = [
   url(r'^$', views.index, name='index'),
   url(r'^questions/$', views.index, name='index'),
   url(r'^questions/(?P<id>(\d+))$', views.question, name='question'),
   url(r'^singup/$', views.singup, name='singup'),
   url(r'^login/$', views.singin, name='login'),
   url(r'^logout/$', views.log_out, name="logout")
]
```

ask/views.py

```
# coding=utf-8
from django.shortcuts import render
from .models import Question
from django.shortcuts import render
from django.shortcuts import render_to_response
from django.template.loader import render_to_string
from django.http import HttpResponse
from django.http import HttpResponseRedirect
from django.template import RequestContext, loader
from django.contrib.auth.models import User
from django.contrib.auth import authenticate, login, logout
from django.contrib.auth.decorators import login_required
from django.core.paginator import Paginator
from django.http.response import Http404, JsonResponse
# django imports
from django.shortcuts import render, redirect
from .forms import RegisterForm, LoginForm
# Create your views here..
def index(request):
  questions = Question.objects.all()
  content = {
    'questions': questions
  }
  for q in questions:
    print(q)
  return render(request, 'index.html', content)
def question(request, id):
  q = Question.objects.get(id=int(id))
  content = {
    'question' : q
```

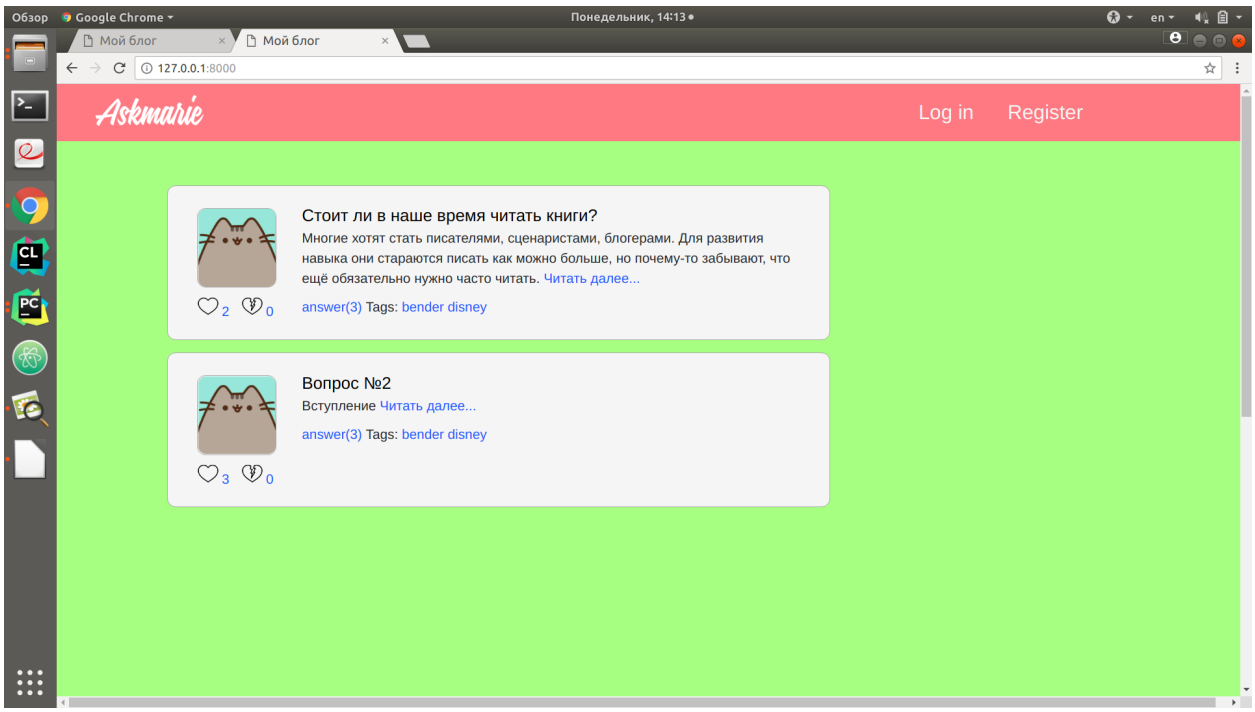
```
    }
    return render(request, 'questions.html', content)

def singup(request):
    errors = []
    formdata = {}
    if request.method == 'POST':
        username = request.POST.get('username')
        if not username:
            errors.append("Введите имя пользователя")
        elif len(username) < 5:
            errors.append("Имя пользователя должно содержать не менее 5
символов")


        email = request.POST.get('email')
        if not email:
            errors.append("Введите адрес эл. почты")


        firstname = request.POST.get('firstname')
        if not firstname:
            errors.append("Введите своё имя")
        else:
            formdata['firstname'] = firstname


        lastname = request.POST.get('lastname')
        if not lastname:
            errors.append("Введите своё фамилию")
        else:
            formdata['lastname'] = lastname


        password = request.POST.get('password')
        if not password:
            errors.append("Введите пароль")
        elif len(password) < 8:
```

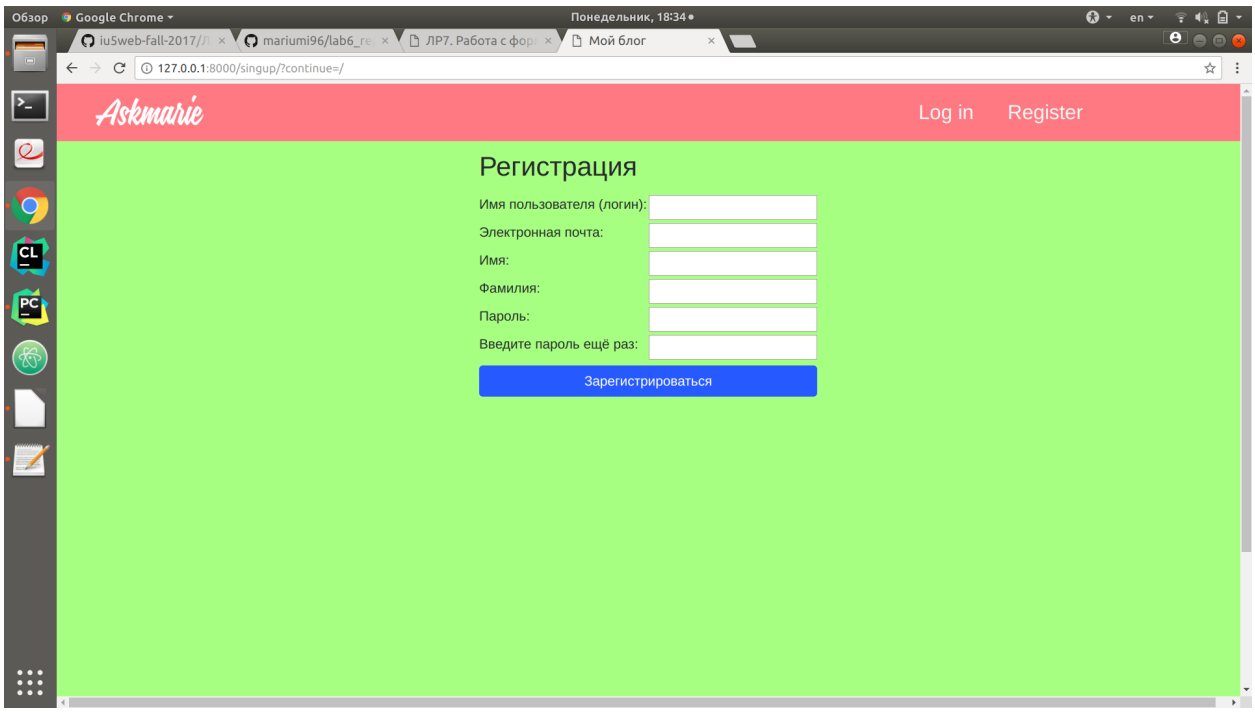
```
            errors.append("Пароль должен содержать не менее 8 символов")
        else:
            confirmpass = request.POST.get('confirmpass')
            if not confirmpass:
                errors.append("Подтвердите пароль")
            elif password != confirmpass:
                errors.append("Пароли не совпадают")
                formdata['confirmpass'] = confirmpass
            formdata['password'] = password

        sameusers = []
        try:
            sameusers.append(User.objects.get(username=username))
        except User.DoesNotExist:
            formdata['username'] = username
        try:
            sameusers.append(User.objects.get(email=email))
        except User.DoesNotExist:
            formdata['email'] = email

        if sameusers:
            errors.append("Пользователь с таким именем или адресом эл. почты
уже существует")

        if errors:
            return render(request, 'singup.html', {'errors': errors, 'formdata': formdata})

        User.objects.create_user(username=username, email=email,
password=password)
        return HttpResponseRedirect("/login/")

    return render(request, 'singup.html', {'errors': [], 'formdata': formdata})
```

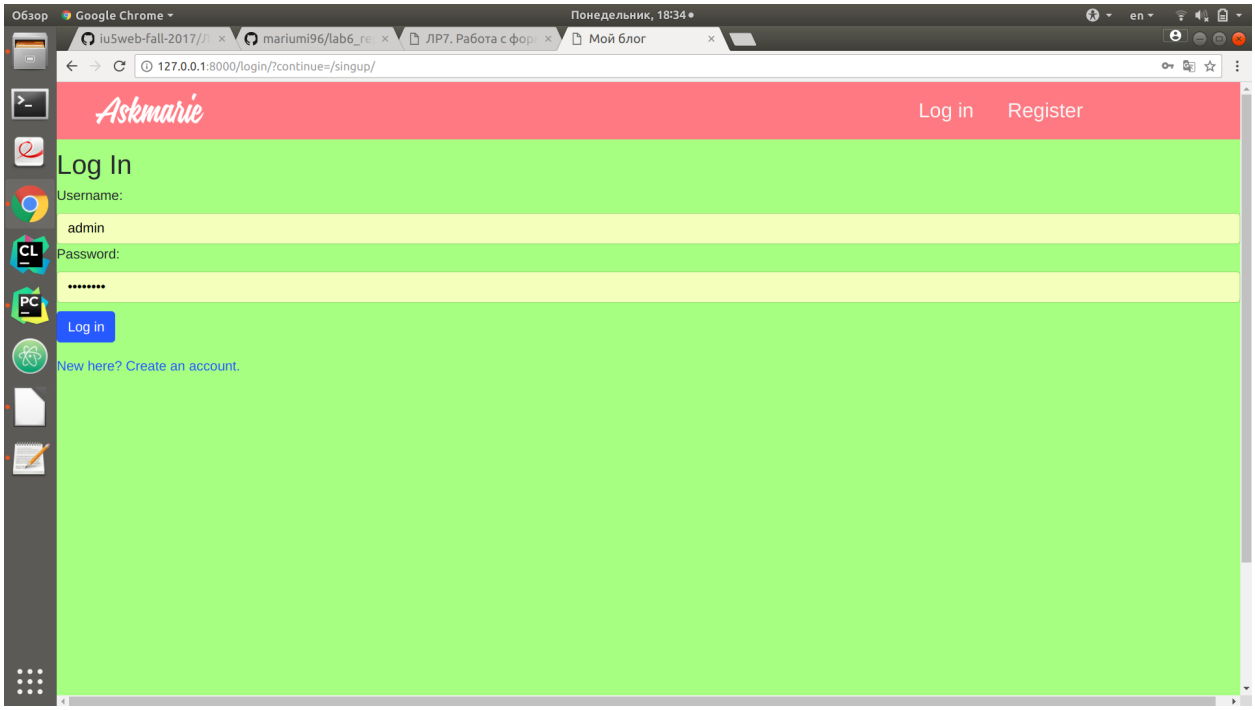
```
#def login(request):
#    return render(request, 'login.html')



def add_user(request, context):
    usr = User.objects.filter(id=request.user.id).last()
    context['userinfo'] = usr
    return context

def singin(request):
    # check if continue exists
    continue_path = request.GET.get("continue", '/')
    # if user is authenticated, then redirect him to info page
    if request.user.is_authenticated:
        context = {}
        context = add_user(request, context)
        # add a continue path for the user to go back
        context["continue"] = continue_path


    # if a user just wants to login
    if request.method == 'GET':
        form = LoginForm()
    else: # else, if he sends some data in POST
        form = LoginForm(request.POST) # initialize the form with POST data
        if form.is_valid():
            username = form.cleaned_data["username"]
            password = form.cleaned_data["password"]
            user = authenticate(request=request, username=username,
password=password) # try auth
            if user is not None: # if auth is success
                login(request, user) # start session
                return redirect(continue_path) #redirect to continue or to /
            else: # else, auth gone wrong
                form.add_error(None, "Username or password is incorrect")
```

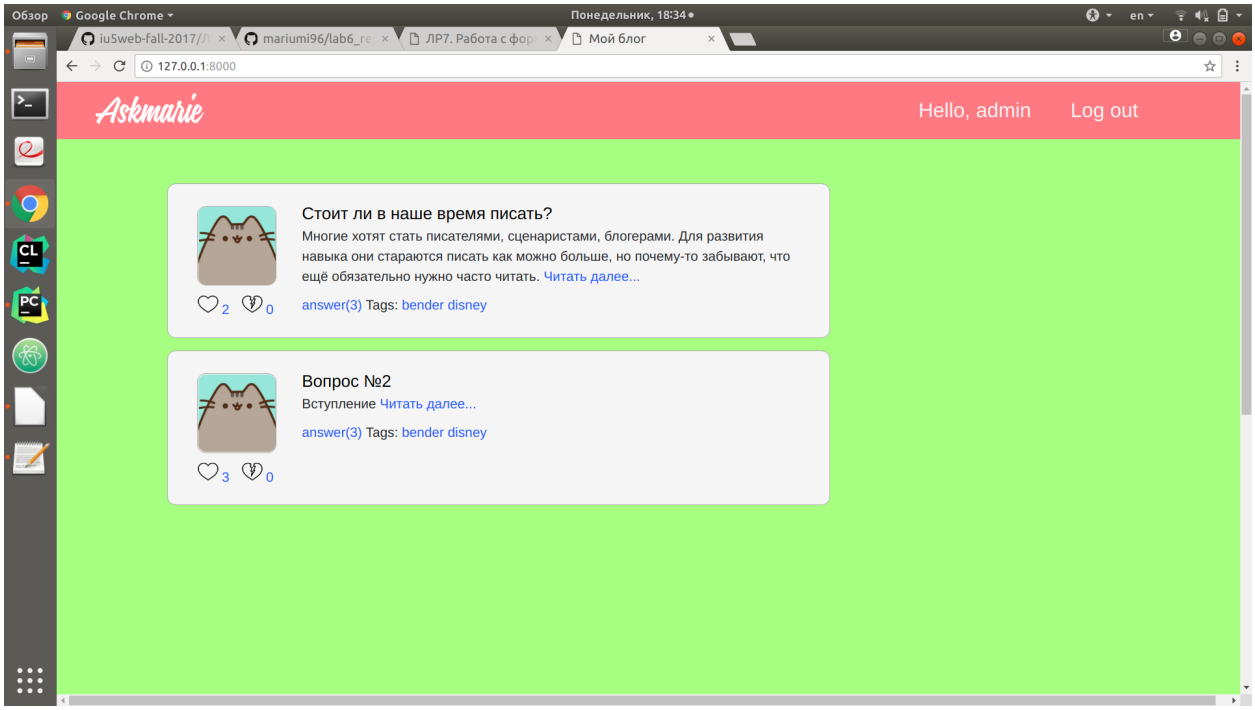
```
    return render(request, 'login.html', {'form': form})


def signup(request):
    if request.user.is_authenticated:
        context = {}
        context = add_user(request, context)
        return render(request, 'questions.html', context)#singed_in
    if request.method == 'GET':
        register_form = RegisterForm()
    else:
        register_form = RegisterForm(request.POST, request.FILES)
        if register_form.is_valid():
            new_profile = register_form.save()
            login(request, new_profile)
            return redirect('index')
    return render(request, 'singup.html', {'form': register_form})



@login_required()
def log_out(request):
    continue_path = request.GET.get("continue", '/questions')
    logout(request)
    return redirect('/questions')
```

Результаты работы программы


Обзор
Google Chrome
Понедельник, 14:13
Мой блог
Мой блог
127.0.0.1:8000
Askmarie
Log in
Register
Стоит ли в наше время читать книги?
Многие хотят стать писателями, сценаристами, блогерами. Для развития навыка они стараются писать как можно больше, но почему-то забывают, что ещё обязательно нужно часто читать. Читать далее...
2
0
answer(3) Tags: bender disney
Вопрос №2
Вступление Читать далее...
answer(3) Tags: bender disney
3
0


Обзор
Google Chrome
Понедельник, 18:34
Мой блог
127.0.0.1:8000/singup/?continue=/
Askmarie
Log in
Register
Регистрация
Имя пользователя (логин):
Электронная почта:
Имя:
Фамилия:
Пароль:
Введите пароль ещё раз:
Зарегистрироваться


Обзор
Google Chrome
Понедельник, 18:34
Мой блог
127.0.0.1:8000/login/?continue=/singup/
Askmarie
Log in
Register
Log In
Username:
admin
Password:
Log in
New here? Create an account.


Обзор
Google Chrome
Понедельник, 18:34
Мой блог
127.0.0.1:8000
Askmarie
Hello, admin
Log out
Стоит ли в наше время писать?
Многие хотят стать писателями, сценаристами, блогерами. Для развития навыка они стараются писать как можно больше, но почему-то забывают, что ещё обязательно нужно часто читать. Читать далее...
2
0
answer(3) Tags: bender disney
Вопрос №2
Вступление Читать далее...
answer(3) Tags: bender disney
3
0