

МГТУ им. Н. Э. Баумана, кафедра ИУ5
курс “Разработка интернет-приложений”

Лабораторная работа №6
Работа с формами, авторизация и модуль
администрирования в Django

ВЫПОЛНИЛ:

Матюнин да Вейга Р.А.

Группа: ИУ5-51Б

ПРОВЕРИЛ:

Гапанюк Ю.Е.

Москва 2019

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'a через команду manage.py

11. Подключить django.contrib.admin и войти в панель администрирования.

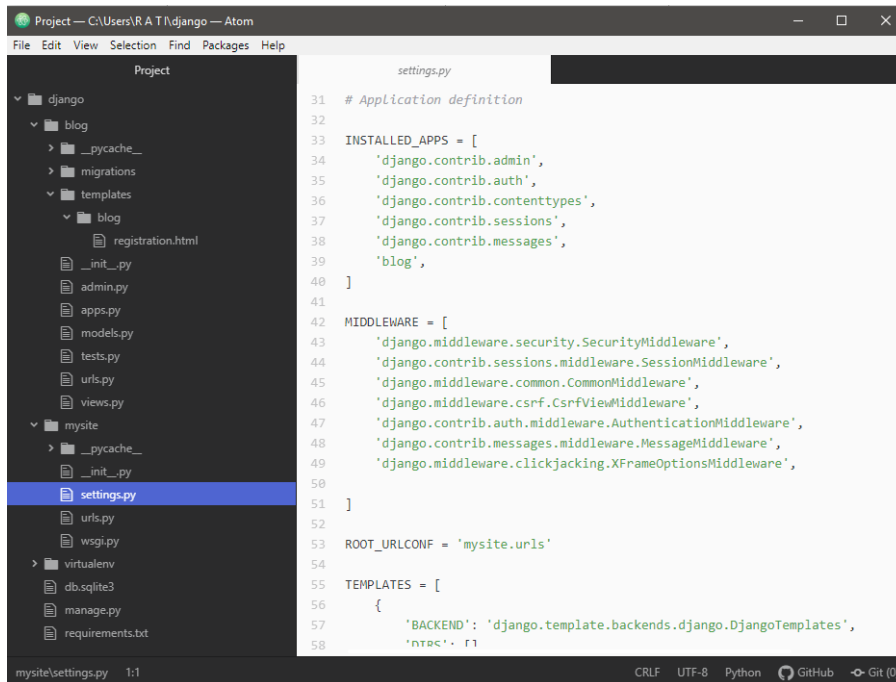
12. Зарегистрировать все свои модели в django.contrib.admin

13. Для выбранной модели настроить страницу администрирования:

- Настроить вывод необходимых полей в списке
- Добавить фильтры
- Добавить поиск
- Добавить дополнительное поле в список

Выполненная работа

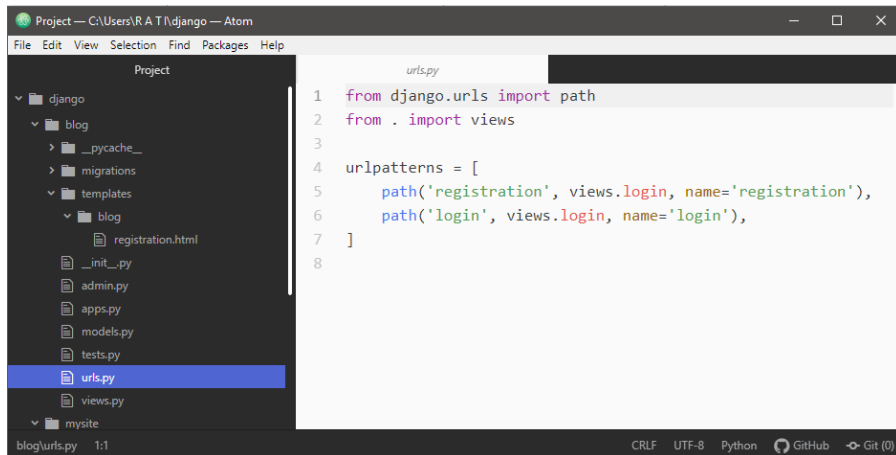
settings.py



The screenshot shows the Atom editor interface with a project named 'Project - C:\Users\RA T\I\django'. The left sidebar displays a file tree with folders 'django' and 'mysite'. The 'django' folder is expanded, showing subfolders like 'blog', 'migrations', and 'templates', along with files like 'registration.html', 'urls.py', and 'views.py'. The 'mysite' folder is also expanded, showing 'urls.py' and 'wsgi.py'. The 'settings.py' file is selected and highlighted in blue. The main editor area shows the content of 'settings.py', which includes application definition, installed apps, middleware, root URLconf, and templates.

```
31 # Application definition
32
33 INSTALLED_APPS = [
34     'django.contrib.admin',
35     'django.contrib.auth',
36     'django.contrib.contenttypes',
37     'django.contrib.sessions',
38     'django.contrib.messages',
39     'blog',
40 ]
41
42 MIDDLEWARE = [
43     'django.middleware.security.SecurityMiddleware',
44     'django.contrib.sessions.middleware.SessionMiddleware',
45     'django.middleware.common.CommonMiddleware',
46     'django.middleware.csrf.CsrfViewMiddleware',
47     'django.contrib.auth.middleware.AuthenticationMiddleware',
48     'django.contrib.messages.middleware.MessageMiddleware',
49     'django.middleware.clickjacking.XFrameOptionsMiddleware',
50 ]
51
52
53 ROOT_URLCONF = 'mysite.urls'
54
55 TEMPLATES = [
56     {
57         'BACKEND': 'django.template.backends.django.DjangoTemplates',
58         'DIRS': [
59             os.path.join(BASE_DIR, 'templates'),
60             os.path.join(BASE_DIR, 'blog', 'templates'),
61         ],
62         'APP_DIRS': True,
63         'OPTIONS': {
64             'context_processors': [
65                 'django.template.context_processors.debug',
66                 'django.template.context_processors.request',
67                 'django.contrib.auth.context_processors.auth',
68                 'django.contrib.messages.context_processors.messages',
69             ],
70         },
71     },
72 ]
```

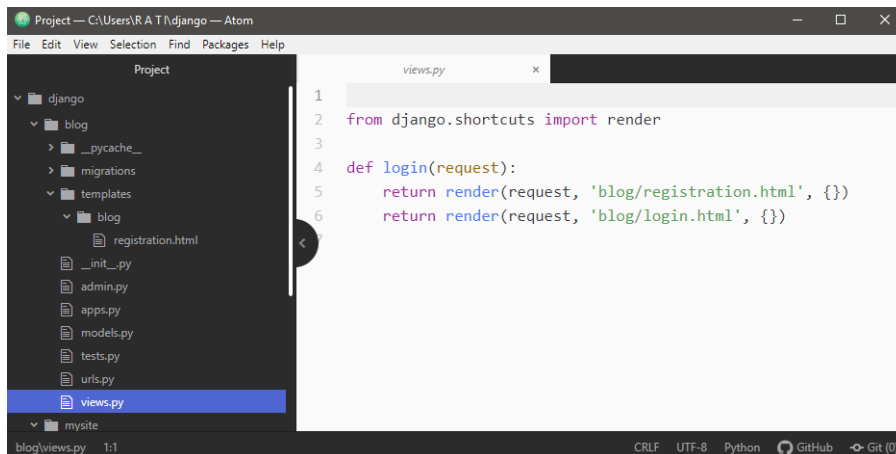
urls.py



The screenshot shows the Atom editor interface with the same project. The left sidebar shows the file tree with 'urls.py' selected and highlighted in blue. The main editor area shows the content of 'urls.py', which includes imports for path and views, and a list of URL patterns for 'registration' and 'login'.

```
1 from django.urls import path
2 from . import views
3
4 urlpatterns = [
5     path('registration', views.login, name='registration'),
6     path('login', views.login, name='login'),
7 ]
8
```

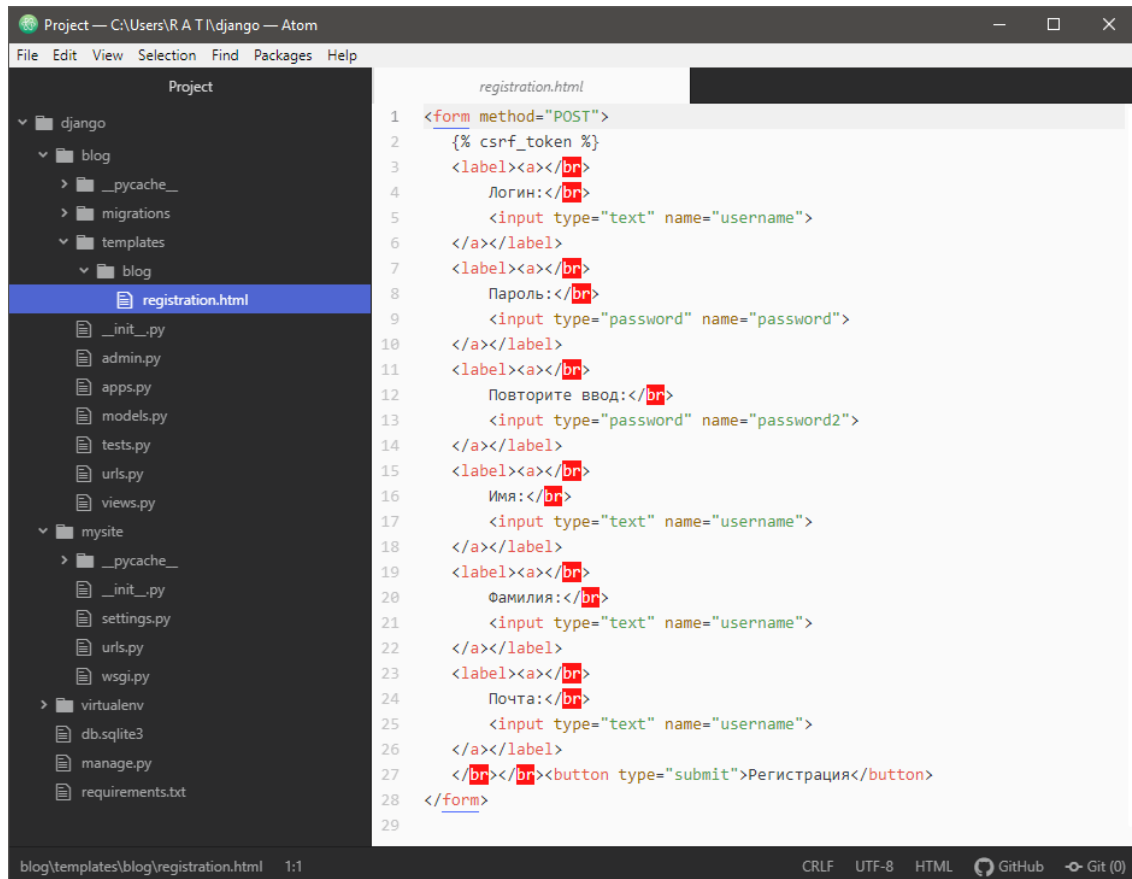
views.py



The screenshot shows the Atom editor interface with the same project. The left sidebar shows the file tree with 'views.py' selected and highlighted in blue. The main editor area shows the content of 'views.py', which includes an import for render and a function 'login' that returns rendered HTML templates.

```
1
2 from django.shortcuts import render
3
4 def login(request):
5     return render(request, 'blog/registration.html', {})
6     return render(request, 'blog/login.html', {})
```

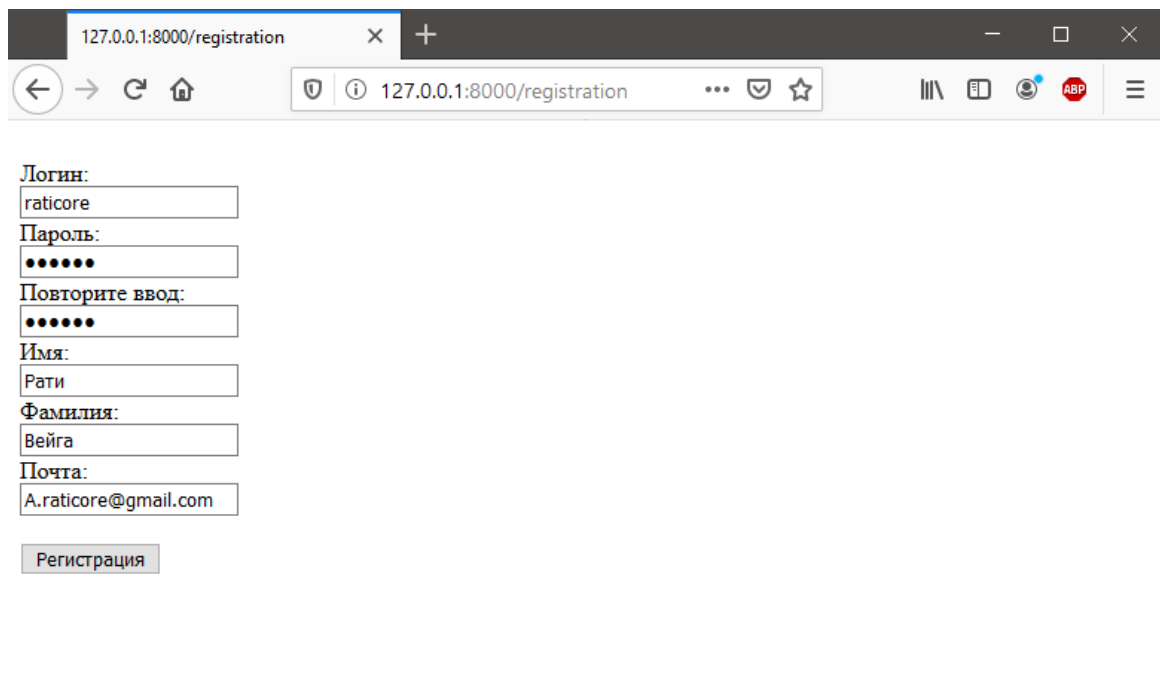
Registration.html



```
1 <form method="POST">
2   {% csrf_token %}
3   <label><a></br>
4   Логин:</br>
5   <input type="text" name="username">
6 </a></label>
7 <label><a></br>
8   Пароль:</br>
9   <input type="password" name="password">
10 </a></label>
11 <label><a></br>
12   Повторите ввод:</br>
13   <input type="password" name="password2">
14 </a></label>
15 <label><a></br>
16   Имя:</br>
17   <input type="text" name="username">
18 </a></label>
19 <label><a></br>
20   Фамилия:</br>
21   <input type="text" name="username">
22 </a></label>
23 <label><a></br>
24   Почта:</br>
25   <input type="text" name="username">
26 </a></label>
27 </br></br><button type="submit">Регистрация</button>
28 </form>
29
```

Результаты

Добавление нового пользователя



127.0.0.1:8000/registration

Логин:
raticore

Пароль:
●●●●●●

Повторите ввод:
●●●●●●

Имя:
Рати

Фамилия:
Вейга

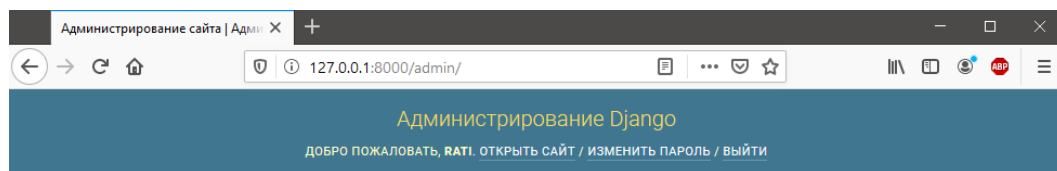
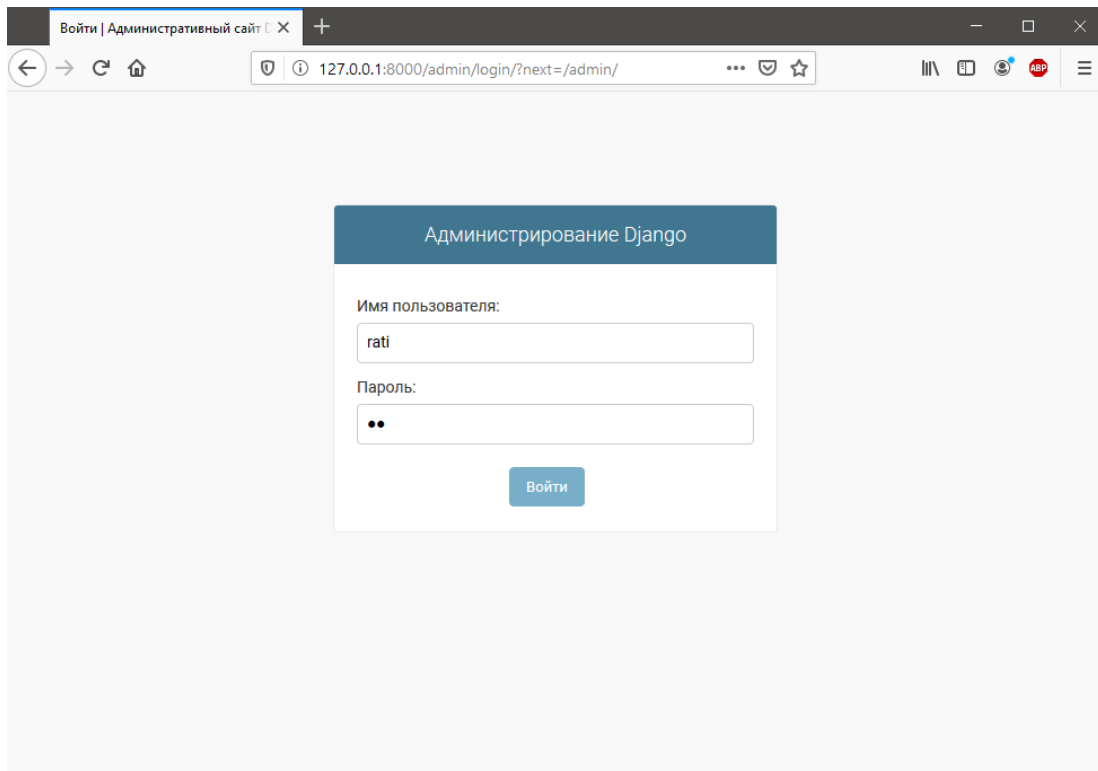
Почта:
A.raticore@gmail.com

Регистрация

Суперпользователь и администрирование в Django

```
Командная строка

(virtualenv) C:\Users\R A T I\django>python manage.py createsuperuser
Имя пользователя (leave blank to use 'rati'): rati
Адрес электронной почты: admin@mail.com
Password:
Password (again):
Введённый пароль слишком короткий. Он должен содержать как минимум 8 символов.
Введённый пароль состоит только из цифр.
Bypass password validation and create user anyway? [y/N]: n
Password:
Password (again):
Введённый пароль слишком короткий. Он должен содержать как минимум 8 символов.
Введённый пароль слишком широко распространён.
Bypass password validation and create user anyway? [y/N]: n
Password:
Password (again):
Введённый пароль слишком короткий. Он должен содержать как минимум 8 символов.
Введённый пароль состоит только из цифр.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```



Администрирование сайта

BLOG		
Posts	+ Добавить	✎ Изменить
ПОЛЬЗОВАТЕЛИ И ГРУППЫ		
Группы	+ Добавить	✎ Изменить
Пользователи	+ Добавить	✎ Изменить

Последние действия

Мои действия

Недоступно

