

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



Отчет
Лабораторная работа № 6
По курсу «Разработка интернет-приложений»

**«Работа с формами, авторизация и модуль администрирования в
Django»**

ИСПОЛНИТЕЛЬ:

Матиенко Андрей

Группа ИУ5-51

"__" _____ 2019 г.

ПРЕПОДАВАТЕЛЬ:

Гапанюк Ю.Е.

"__" _____ 2019 г.

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

Регистрация:

[←](#) [Я](#) [↻](#) [🌐](#) 127.0.0.1:8000 127.0.0.1:8000/registration

Username:

Password:

Password confirmation: Enter the same password as before, for verification.

Имя:

Фамилия:

Почта:

Password: MOleva9999

Views.py:

Registration.html

#Регистрация

```
class RegistrationForm(FormView):
    form_class = RegistrationForm
    success_url = "/login/"
    template_name = "registration.html"

    def form_valid(self, form):
        form.save()
        return super(RegistrationForm, self).form_valid(form)
```

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

Forms.py:

```
class RegistrationForm(UserCreationForm):
    email = forms.EmailField(max_length=30, label="Почта")
    first_name = forms.CharField(max_length=30, label="Имя")
    last_name = forms.CharField(max_length=30, label="Фамилия")

    def __init__(self, *args, **kwargs):
        super(UserCreationForm, self).__init__(*args, **kwargs)
        self.fields['username'].help_text = ''
        self.fields['password1'].help_text = ''

    class Meta:
        model = User
        fields = ('username', 'password1', 'password2', 'first_name', 'last_name', 'email')
```

url.py:

```
url(r'^registration/', views.RegistrationForm.as_view(), name="registration"),
```

Авторизация:

Views.py:

```
class LoginForm(FormView):
    form_class = LoginForm
    success_url = "/profile/"
    template_name = "login.html"

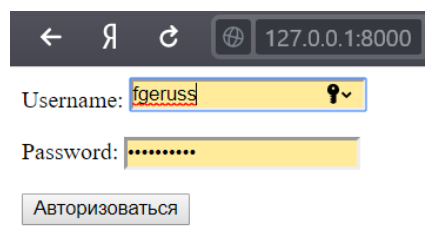
    def form_valid(self, form):
        self.user = form.get_user()
        login(self.request, self.user)
        return super(LoginForm, self).form_valid(form)
```

Forms.py:

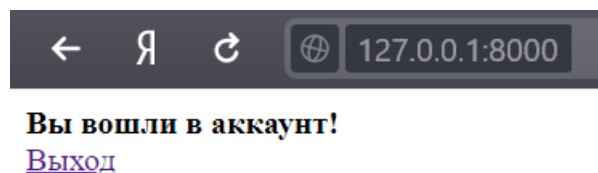
```
class LoginForm(AuthenticationForm):
    pass
```

Urls.py:

```
url(r'^login/', views.LoginForm.as_view(), name="login"),
```



Вход в профиль и выход из него:



url.py:

```
url(r'^profile', views.user_profile, name="profile")
```

Profile.html:

```
<b>Вы вошли в аккаунт!</b><br>
<a href="/logout">Выход</a>
```

Views.py:

```
#Валидация и вход
@login_required(login_url='/login/')
def user_profile(request):
    return render(request, 'profile.html')

class LogoutForm(View):
    def get(self, request):
        logout(request)
        return HttpResponseRedirect("/")
```

Вывод:

Научились работе в Django по созданию авторизации.