**Аутентификация пользователя**

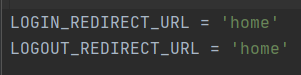
Теперь, когда у нас есть рабочая пользовательская модель, мы можем добавить функциональность, которая нужна каждому веб-сайту: возможность регистрироваться, входить и выходить из системы. Django предоставляет все необходимое для входа и выхода, но нам нужно будет создать собственную форму для регистрации новых пользователей. Мы также создадим базовую домашнюю страницу со ссылками на все три функции, поэтому нам не придется каждый раз вводить URL вручную.

**Templates(Шаблоны)**

По умолчанию загрузчик шаблонов Django ищет шаблоны в вложенной структуре в каждом приложении. Таким образом, шаблон home.html для пользователей должен быть расположен по адресу users / templates / users / home.html. Но подход к папке с шаблонами на уровне проекта более понятен и лучше масштабируется, поэтому мы будем использовать его. Давайте создадим новый каталог шаблонов и внутри него папку registration, и в этой папке шаблон login.html.

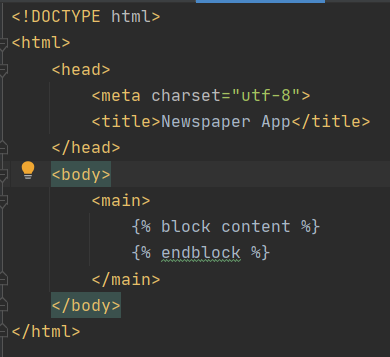
Теперь нам нужно сообщить Django об этом новом каталоге, обновив конфигурацию для 'DIRS' в settings.py. Сделайте это самостоятельно.

Если вы думаете о том, что происходит, когда вы входите или выходите из сайта, вы сразу же перенаправляетесь на следующую страницу. Нам нужно указать Django, куда отправлять пользователей в каждом конкретном случае. Делают это настройки LOGIN\_REDIRECT\_URL и LOGOUT\_REDIRECT\_URL. Мы настроим оба, чтобы перенаправить на нашу домашнюю страницу, которая будет иметь названный URL «home». Помните, что когда мы создаем наши маршруты URL, у нас есть возможность добавить имя для каждого из них. Поэтому, когда мы создадим URL домашней страницы, мы обязательно назовем его «home». Добавьте эти две строки внизу файла settings.py.

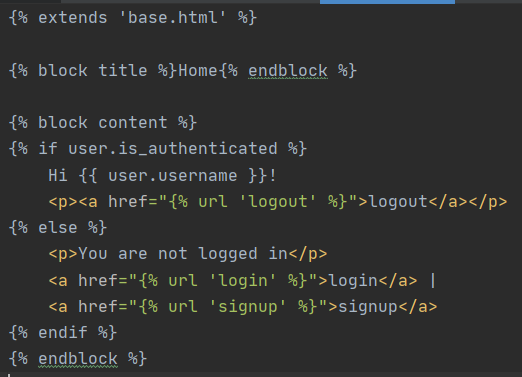


Теперь создайте еще три шаблона, но не в папке registration (так как эти шаблоны не будут относится напрямую к регистрации), а в папке templates: base.html, home.html, signup.html.

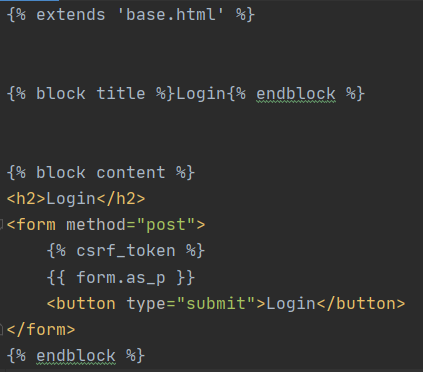
base.html будет унаследован каждым другим шаблоном в нашем проекте. Используя блок типа   
{% block content %}, мы можем позже переопределить содержимое только в этом месте в других шаблонах.



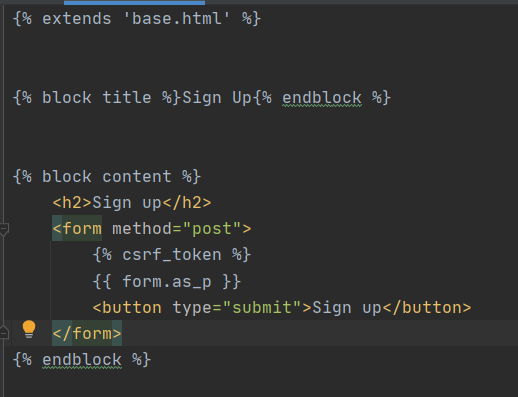
В home.html пропишите следующий код для домашней страницы:



Файл login.html заполните формой для входа:



И шаблон signup.html:

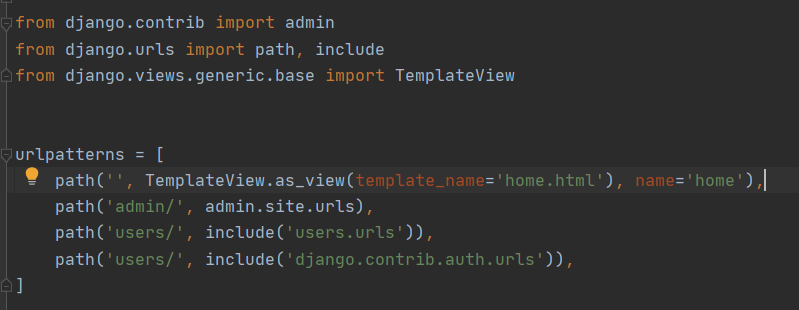


**URLs**

Давайте начнем с URL-маршрутов. В нашем файле urls.py проекта мы хотим, чтобы наш шаблон home.html отображался в качестве домашней страницы. Но мы пока не хотим создавать приложение для выделенных страниц, поэтому мы можем использовать быстрый доступ импортируя TemplateView и установки имени шаблона прямо в нашем шаблоне URL.

Далее мы хотим «включить» и приложение пользователя, и встроенное приложение аутентификации. Причина в том, что встроенное приложение авторизации уже предоставляет представления и URL для входа и выхода из системы. Но для регистрации нам нужно создать наш собственное представление и URL. Чтобы обеспечить согласованность наших URL маршрутов, мы размещаем в users/, поэтому возможными URL будут / users / login, / users / logout и / users / signup.

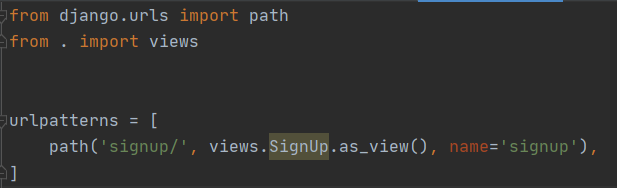
Отредактируем файл urls.py, находящийся в папке проекта:



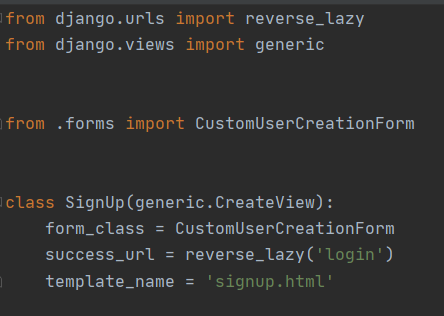
Первый path ссылается на домашнюю страницу благодаря встроенной функциональности django.

Второй path обеспечивает доступ к админ панели. Третий path пересылает все url начинающиеся на users/ в файл urls.py, находящийся в папке приложения users. Четвертый при отсутствии нужного url в приложении пересылает запрос на библиотеку django.

Создайте файл urls.py в приложении users и обновите его следующим образом:



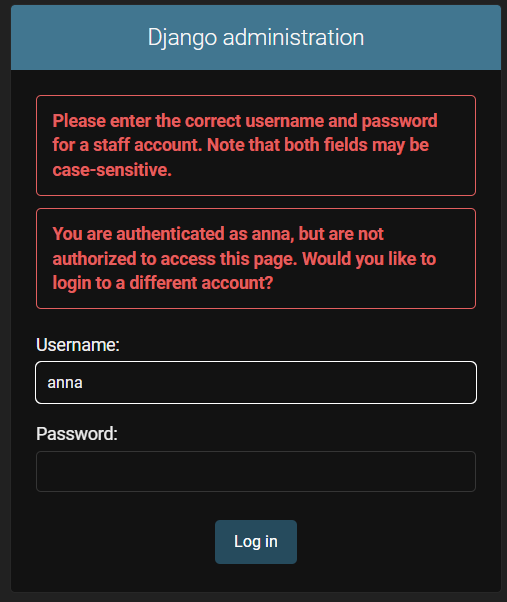
Последний шаг — это наш файл views.py, который будет содержать логику для нашей формы регистрации. Здесь мы используем общий CreateView от Django и говорим ему использовать нашу CustomUserCreation-Form и перенаправлять на вход в систему после успешной регистрации пользователя, и что наш шаблон называется signup.html.



Запустите сервер и проверьте работу входа и выхода из системы, а также регистрацию.

**Admin**

Зайдите в admin под учетной записью нового созданного вами при помощи формы регистрации (на вашем сайте) пользователя.



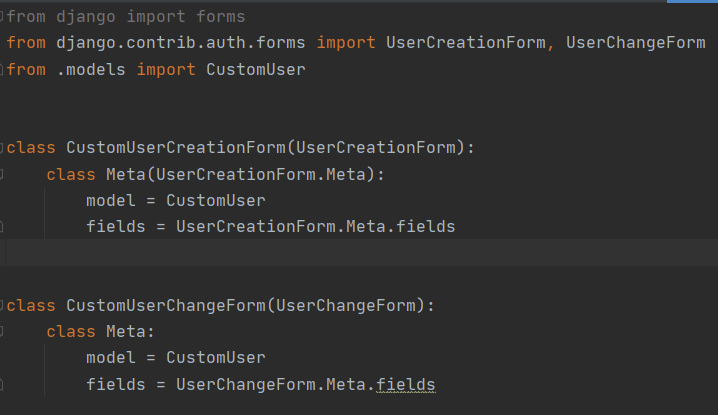
Что это такое? Почему мы не можем войти?

Потому что мы вошли в систему с нашей новой учетной записью тестового пользователя, а не с учетной записью суперпользователя. Только учетная запись суперпользователя имеет разрешения для входа в систему администратора! Поэтому используйте свою учетную запись суперпользователя для входа.

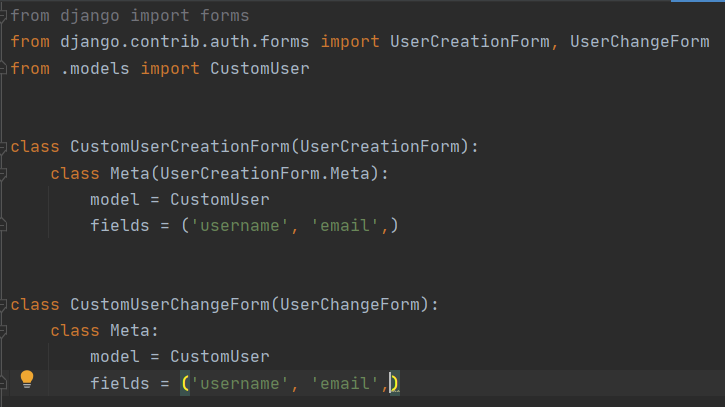
После этого вы увидите обычную домашнюю страницу администратора. Нажмите на Users и вы увидите наших двух пользователей: того, которого мы только что создали, и ваше предыдущее имя суперпользователя.

Все работает, но вы можете заметить, что для нашего тестового пользователя нет поля электронной почты. Почему так? Ну, оглянитесь на страницу регистрации в users/signup/, и вы увидите, что она запрашивает только имя пользователя и пароль, а не электронную почту! Именно так работает настройка Django по умолчанию. Однако мы можем легко изменить это. Вернемся к нашему файлу users/forms.py.

Сейчас это выглядит так:



Мы используем Meta.fields, которые просто отображают настройки по умолчанию: имя пользователя/пароль. Но мы также можем явно установить, какие поля мы хотим отобразить, поэтому давайте обновим его, чтобы запросить имя пользователя/адрес электронной почты/пароль, установив его в ('username', 'email',). Нам не нужно включать поле пароля, потому что оно необходимо! Но все остальные поля могут быть настроены, как мы выберем.



Теперь зайдя на страницу регистрации вы увидите поле Email address. Проверьте что все работает хорошо.

Аутентификации пользователя Django требует немного настройки, но вы должны начать видеть, что он также предоставляет нам невероятную гибкость для настройки регистрации и входа в систему именно так, как мы хотим.