**Пользовательские модели**

Встроенная в Django модель User позволяет нам сразу начать работать с пользователями, как мы только что делали с нашим приложением блога в предыдущих главах. Однако официальная документация Django настоятельно рекомендует использовать пользовательскую модель для новых проектов. Причина в том, что, если вы хотите внести какие-либо изменения в пользовательскую модель в будущем, например, добавив поле возраста с помощью custom user model с самого начала это довольно просто. Но если вы не создадите custom user model, обновление модели пользователя по умолчанию в существующем проекте Django будет очень и очень сложным.

Поэтому всегда используйте **пользовательскую модель** для всех новых проектов Django. Однако пример официальной документации не совсем то, что рекомендуют многие эксперты Django. Он использует довольно сложный AbstractBaseUser, чем если мы просто используем AbstractUser, все гораздо проще и все еще настраиваемо. Это подход, который мы будем использовать в этой практической работе, где мы запустим новое приложение Newspaper (Газета) с пользовательской моделью.

**Создайте и настройке новый проект. Приложение назовите users.**

**Не выполняйте миграции на данном этапе.**

Обратите внимание, что мы не выполняли миграцию для настройки базы данных. Важно подождать, пока мы не создадим нашу новую пользовательскую модель, прежде чем делать это, учитывая, насколько тесно связана пользовательская модель с остальной частью Django.

Проверьте что создали и настроили проект верно.

**Custom User Model**

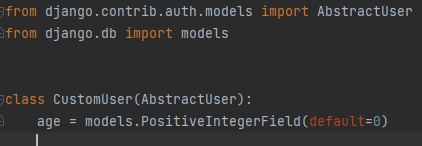
Создание нашей собственной модели пользователя требует четыре шага:

* обновить settings.py
* создание новой CustomUser модели
* создание новой формы для UserCreation и UserChangeForm
* обновить админку сайта

В settings.py в нижней части файла используйте конфигурацию AUTH\_USER\_MODEL, чтобы указать Django использовать нашу новую пользовательскую модель вместо встроенной модели User. Мы будем называть нашу пользовательскую модель CustomUser так, поскольку она существует в нашем users приложении, мы называем ее users.CustomUser.



Теперь обновите users / models.py новой моделью User, которую мы будем называть CustomUser. Мы также добавим наше первое дополнительное поле для «age» (возраста) наших пользователей. Мы можем использовать PositiveIntegerField от Django, что означает, что целое число должно быть либо положительным, либо нулевым.

****

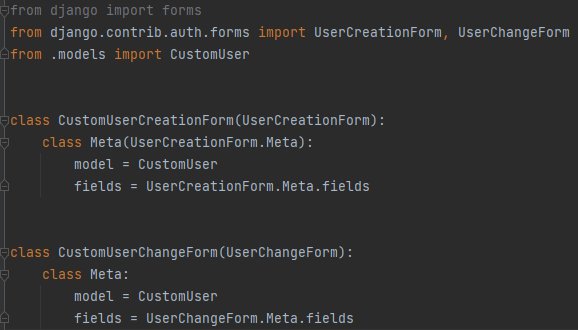
Это весь код, который нам нужен. Поскольку мы расширяем AbstractUser, наш CustomUser является в основном копией модели User по умолчанию. Единственное обновление - наше поле age.

**Формы**

Если мы на шаг отступим назад, как мы будем взаимодействовать с нашей новой моделью CustomUser? Первый случай, когда пользователь регистрирует новую учетную запись на нашем сайте. Другой находится в приложении администратора, которое позволяет нам, как суперпользователям, изменять существующих пользователей. Поэтому нам нужно обновить две встроенные формы для этой функции: UserCreationForm и UserChangeForm.

Cоздайте новый файл в приложении users с именем forms.py.

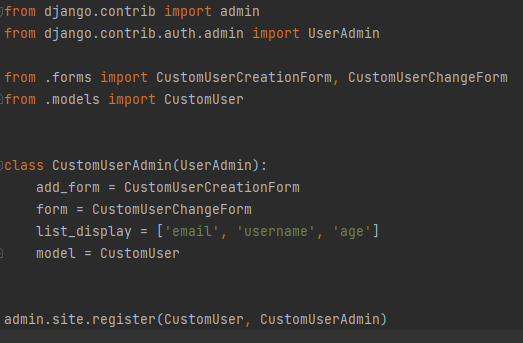
Мы обновим его следующим кодом, чтобы расширить существующие формы UserCreationForm и UserChangeForm:



Для обеих новых форм мы устанавливаем модель CustomUser и используем поля по умолчанию с помощью Meta.fields.

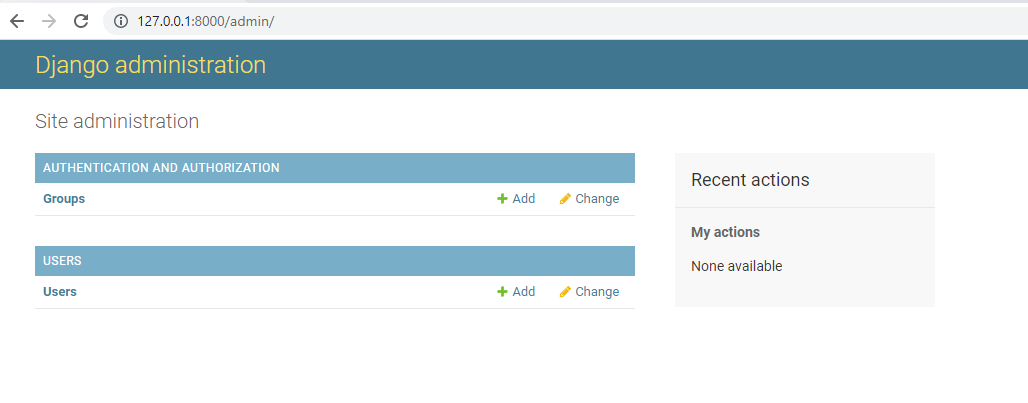
Поначалу понятие полей в форме может сбивать с толку, поэтому давайте уделим немного времени, чтобы изучить его дальше. Наша модель CustomUser содержит все поля модели пользователя по умолчанию и наше дополнительное поле возраста, которое мы установили. Но что это за поля по умолчанию? Оказывается, есть много, включая имя пользователя, имя, фамилия, адрес электронной почты, пароль, группы и многое другое. Тем не менее, когда пользователь регистрирует новую учетную запись в Django, форма по умолчанию запрашивает только имя пользователя, адрес электронной почты и пароль.

Это говорит нам о том, что настройка по умолчанию для полей в UserCreationForm — это просто имя пользователя, адрес электронной почты и пароль, хотя доступно еще много полей. Это может быть не понятно, так как для правильного понимания форм и моделей требуется некоторое время. В следующей работе мы создадим наши собственные страницы регистрации, входа и выхода, которые будут более четко связывать нашу модель CustomUser и формы. Так что держитесь! Последний шаг — это обновить наш файл admin.py, так как Admin тесно связан с моделью пользователя по умолчанию. Мы расширим существующий класс UserAdmin для использования нашей новой модели CustomUser и двух новых форм.



Обратите внимание, что наш CustomUserAdmin также имеет параметр list\_display, чтобы он отображал только поля email, username и age, даже если на данный момент в модели CustomUser их гораздо больше.

Выполите миграции чтобы создать базу данных использующую пользовательскую модель. Далее создайте суперпользователя (установите имя и пароль как root) и запустите сервер.



Если вы нажмете на ссылку "Users", вы увидите свою учетную запись суперпользователя. Обратите внимание, что мы видим три поля: электронная почта, имя пользователя и возраст, так как мы установили их в list-display в нашем CustomUserAdmin.