**Lab 5**

**Custom User Model**

Django’s built-in User model allows us to start working with users right away, as we just did with our Blog app in the previous lab. However, the official Django documentation highly recommends using a custom user model for new projects. The

reason is that if you want to make any changes to the User model down the road–-for

example adding an age field-–using a custom user model from the beginning makes

this quite easy. But if you do not create a custom user model, updating the default User model in an existing Django project is very difficult.

**Initial Set Up**

Refer to earlier lab sheets if you don’t remember the commands

* Create a folder called lab5
* Create & Activate your virtual environment & install Django
* Create a new project called newspaperproject
* Create a new app called accounts and register this app in settings.py

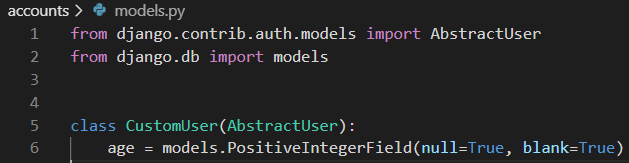
**Custom User Model**

In **settings.py** add the users app to the INSTALLED\_APPS.

At the bottom of the **settings.py** file use the AUTH\_USER\_MODEL config to tell Django to use our new custom user model in place of the built-in User model. We will call our custom user model CustomUser so, since it exists within our users app we refer to it as users.CustomUser.



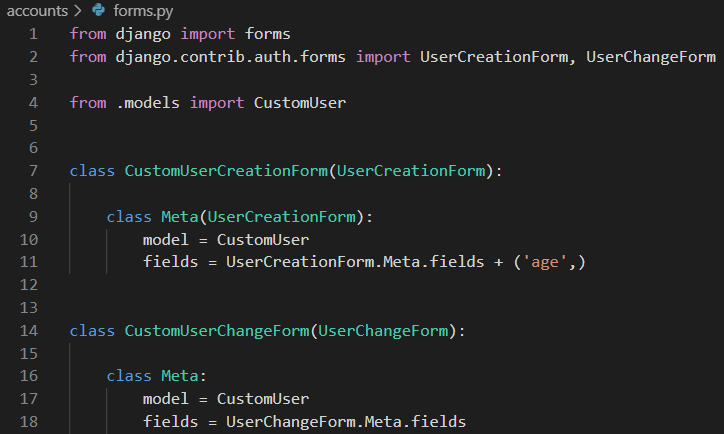
Update **accounts/models.py** with a new User model called CustomUser that extends the existing AbstractUser. We also include our first custom field, age, here.



**Forms**

There are two ways in which we can interact with our new CustomUser model. One case is when a user signs up for a new account on our website. The other is within the admin app which allows us, as superusers, to modify existing users. We will need to update the two built-in forms for this functionality: UserCreationForm and UserChangeForm.

Stop the local server with Control+BREAK and create a new file in the accounts app called forms.py. Update this file with the following code:



For both new forms we are setting the model to our CustomUser and using the default

fields via Meta.fields which includes all default fields. To add our custom age field we simply tack it on at the end and it will display automatically on our future sign up page.

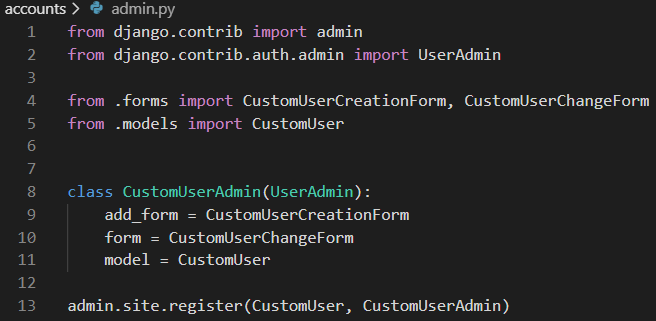
Our CustomUser model contains all the fields of the default User model and our additional age field which we set.

There are many default fields including username, first\_name, last\_name, email, password, groups, and more. Yet when a user signs up for a new account on Django the default form only asks for a username, email, and password. This tells us that the default setting for fields on UserCreationForm is just username, email, and password even though there are many more fields available.

Understanding forms and models properly takes some time. In the next exercise we will create our own sign up, log in, and log out pages which will tie together our CustomUser model and forms more clearly.

The only other step we need is to update our admin.py file since Admin is tightly coupled to the default User model. We will extend the existing UserAdmin class to use our new CustomUser model.

Open admin.py and enter the following code:



Type Control+BREAK to stop the local server and run makemigrations and migrate for the first time to create a new database that uses the custom user model.

Command Line

(env) djangoprojects\ lab5 >python manage.py makemigrations accounts

(env) djangoprojects\ lab5 >python manage.py migrate

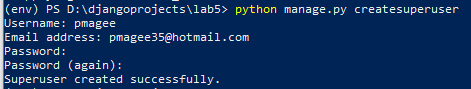
**Superuser**

Create a superuser account to confirm that everything is working as expected. On the command line type the following command and go through the prompts.

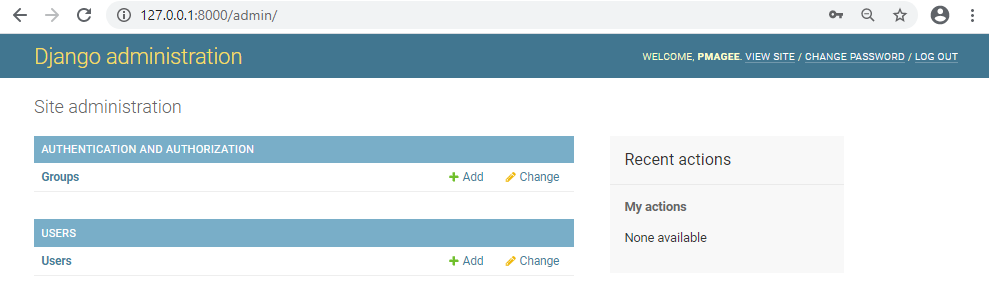
**Command Line**

(env) djangoprojects\ lab5 >python manage.py createsuperuser

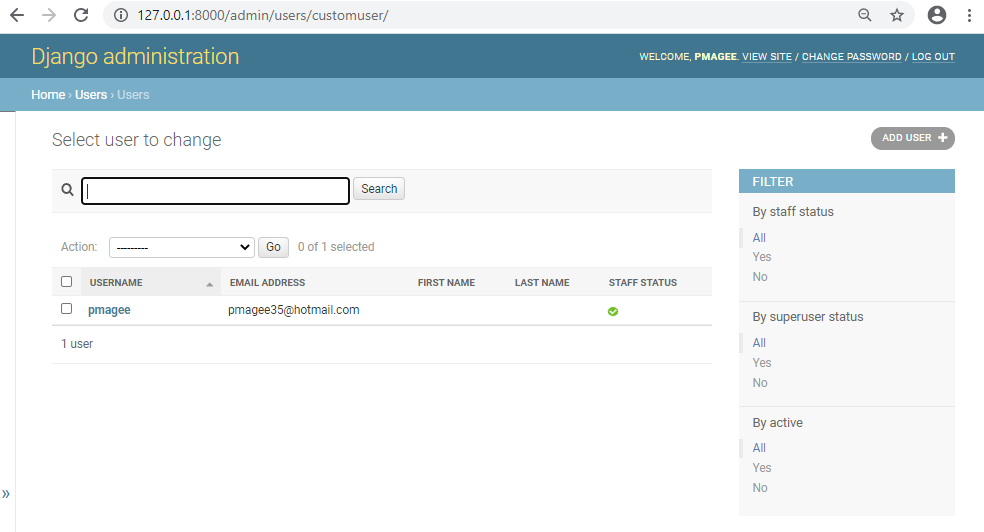
The fact that this works is the first proof our custom user model works as expected.



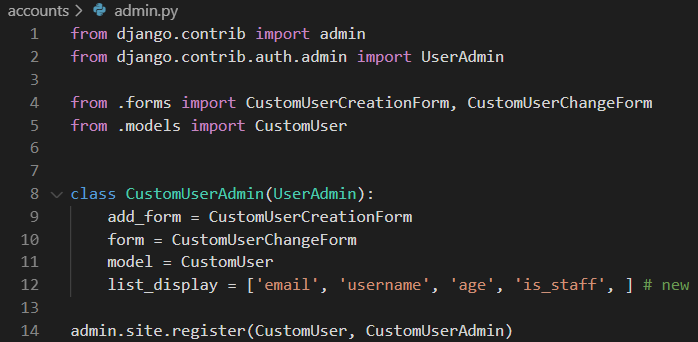
Start up the web server and navigate to the admin at http://127.0.0.1:8000/admin and log in.



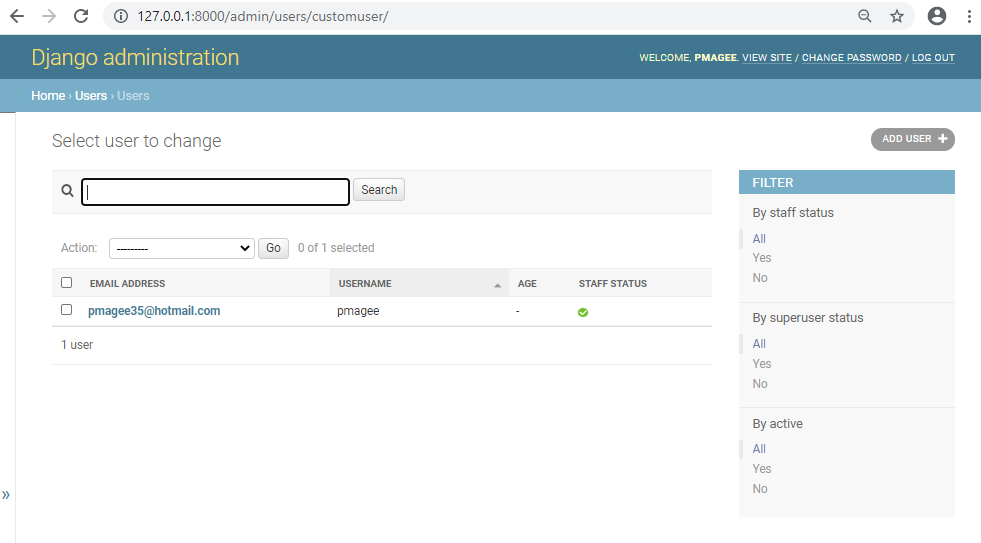
If you click on the link for “**Users**” you should see your superuser account as well as the default fields of Username, Email Address, First Name, Last Name, and Staff Status.



We can control the fields listed here via the list\_display setting for CustomUserAdmin. Add the following line of code to admin.py:



Refresh the page and you should see the update. Now we see the fields email, username, age and is\_staff:



Create and initialise a local repo for your lab 5 project and follow the steps set out in lab 2 (page 11-14) to push your code to the Classroom GitHub repository – see link for lab 5 on Moodle