Lab 9 Part 1

Creation of a User Profile

Step 1 - Initial Set Up

1. Go to www.github.com and log in to your GitHub account.
2. Go to Moodle and click on the link for Lab 9.
3. Once you have accepted the assignment you are asked to refresh the page, and you are then presented with a link to your repository for lab 9.
4. When you click on the repository link, you are taken to the repository where you see a readme file has already been added:
5. In Windows Command line make sure that you are in the djangoprojects folder and type the suitable command to activate the virtual environment:
6. Use the git clone command to clone the repo to your local computer:
7. Move into this lab-9-username folder using the cd command.

The repository contains starter code which has an **accounts** app with a **CustomUser** model defined as well as the features, Sign Up, Sign In, Log Out, Password Change and Password Reset. There is also a **pages** app created which has a home page. The home page is shown below with a navbar with items **BookStore** which just directs to the home page, and **Books** which doesn't do anything yet but will display a list of books when we implement the functionality by creating a new app called **books**.

Run the server, sign up with a new account, log in and make sure that you can view the home page as shown below:

The first part of this exercise is to create a user profile which we will represent in our model as a 1-1 relationship with **CustomUser** i.e., a user has just one profile.

Open **accounts/models.py** and add in the code shown below:

```
accounts > models.py > ...
1   from django.db import models
2   from django.contrib.auth.models import AbstractUser
3   from django.contrib.auth import get_user_model
4   from django.urls import reverse
5
6   # Create your models here.
7
8   class CustomUser(AbstractUser):
9       age = models.PositiveBigIntegerField(null=True, blank = True)
10
11
12  class Profile(models.Model):
13      user = models.OneToOneField(
14          get_user_model(),
15          null=True,
16          on_delete=models.CASCADE,
17          )
18      date_of_birth = models.DateField(blank=False, null=False)
19      fav_author = models.CharField(max_length=255)
20
21      def __str__(self):
22          return str(self.user)
23
24      def get_absolute_url(self):
25          return reverse('show_profile', args=[str(self.id)])
```

The line of code on line 11 defines a field called user which is a one to mapping with **CustomUser** defined in our model. We set null to True to deal with situations where we already have users created in our system with no profiles. The on_delete field is set to make sure that if a user is deleted from the system the associated profile is also deleted.

Use the following commands to create this new database table:

---

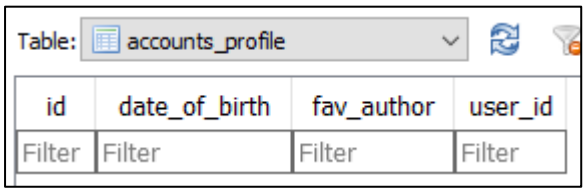
python manage.py makemigrations accounts

python manage.py migrate

We need to register the new Profile model in **admin**. Open **accounts/admin.py** and add the code highlighted below:

```
accounts > admin.py > ...
1   from django.contrib import admin
2   from django.contrib.auth.admin import UserAdmin
3   from .forms import CustomUserCreationForm, CustomUserChangeForm
4   from .models import CustomUser, Profile
5
6
7   # Register your models here.
8   class CustomUserAdmin(UserAdmin):
9       add_form = CustomUserCreationForm
10      form = CustomUserChangeForm
11      model = CustomUser
12      list_display = ['email', 'username','age','is_staff',]
13
14
15  admin.site.register(CustomUser, CustomUserAdmin)
16  admin.site.register(Profile)
```

Open the **db.sqlite3** file in DB Browser and check that a new table called **accounts_profile** has been created.



Create a super user account, run the server, and log in to Django Admin and **create a new profile** for your new account.

Open the file **accounts/views.py** and add in the following imports and code for two class-based views. One view is to display the user profile and the other view is to edit the user profile.

accounts > views.py > ...

```python
from django.urls import reverse_lazy
from django.views.generic import CreateView, UpdateView, DetailView
from .forms import CustomUserCreationForm
from .models import Profile

# Create your views here.

class SignUpView(CreateView):
    form_class = CustomUserCreationForm
    success_url = reverse_lazy('login')
    template_name = 'registration/signup.html'

class ProfileEditView(UpdateView):
    model = Profile
    template_name = 'registration/edit_profile.html'
    fields = ['fav_author']


class ProfilePageView(DetailView):
    model = Profile
    template_name = 'registration/user_profile.html'
```

Edit the **accounts/urls.py** file to include paths for the two new views:

accounts > urls.py > ...

```python
from django.urls import path
from .views import SignUpView, ProfileEditView, ProfilePageView

urlpatterns=[
    path('signup/', SignUpView.as_view(), name='signup'),
    path('edit_profile/<int:pk>/', ProfileEditView.as_view(), name='edit_profile'),
    path('profile/<int:pk>/', ProfilePageView.as_view(), name='show_profile')
]
```

Create a file called **user_profile.html** in the **templates/registration** folder and copy & paste in the following code:

```
{% extends 'base.html' %}
{% block title %}User Profile{% endblock title %}
{% block content %}
    Date Of Birth:{{ user.profile.date_of_birth }}
    <br/>
    Favourite Author: {{ user.profile.fav_author }}
{% endblock content %}
```

Create a file called **edit_profile.html** in the **templates/registration** folder and copy & paste in the following code:

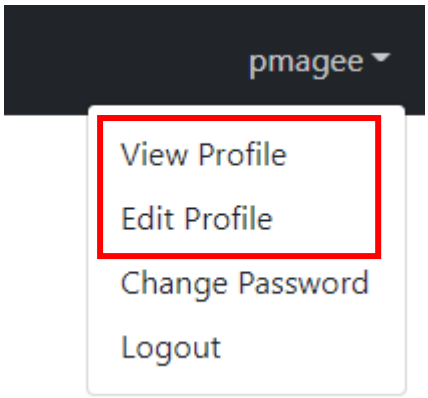
```
{% extends 'base.html' %}
{% block title %}Edit Profile{% endblock title %}
{% block content %}

{% if user.is_authenticated %}
  <h1>Edit Profile</h1>
  <br/>

  <div class="form-group">
    <form method="POST">
      {% csrf_token %}
      {{ form.as_p }}
      <input class="btn btn-success" type="submit" value="Update Profile">
    </form>
    </div>

{% else %}
You're not allowed here
{% endif %}
{% endblock content %}
```

We now need to provide a link in our nav bar for viewing and editing the user profile, but these menu items should only be available if the user has a profile. Open **templates/base.html** and modify the code for the drop down menu to include the following. You can copy-paste the following code and place the existing links for Change password and Logout into the else statement.

```
26          {% if user.is_authenticated %}
27          <ul class="nav navbar-nav ms-auto">
28            <li class="nav-item dropdown">
29              <a href="#" class="nav-link dropdown-toggle" data-bs-toggle="dropdown">{{user.username}}</a>
30              <div class="dropdown-menu dropdown-menu-end">
31                <a href="{% url 'password_change'%}" class="dropdown-item">Change Password</a>
32                <a href="{% url 'logout' %}" class="dropdown-item">Logout</a>
33              </div>
34            </li>
35          </ul>
36        </div>
```

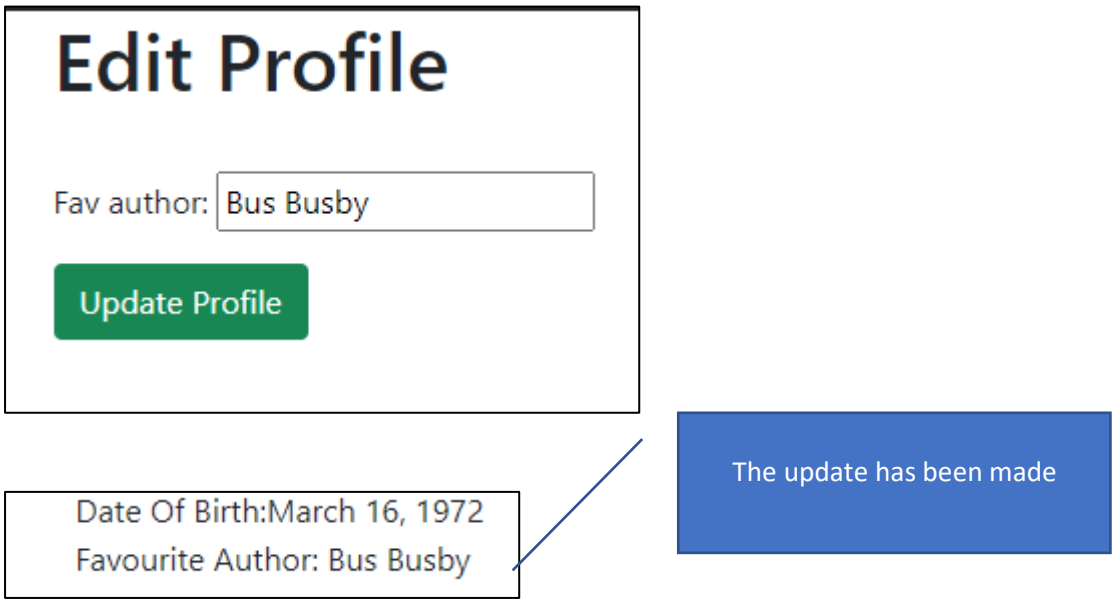
```
<div class="dropdown-menu dropdown-menu-end">
  {% if user.profile %}
    <a href="{% url 'show_profile' user.profile.pk %}" class="dropdown-item">View Profile</a>
    <a href="{% url 'edit_profile' user.profile.pk %}" class="dropdown-item">Edit Profile</a>
    <a href="{% url 'password_change'%}" class="dropdown-item">Change Password</a>
    <a href="{% url 'logout' %}" class="dropdown-item">Logout</a>
  {% else %}
    <a href="{% url 'password_change'%}" class="dropdown-item">Change Password</a>
    <a href="{% url 'logout' %}" class="dropdown-item">Logout</a>
  {% endif %}
</div>
```

We have just added two new menu items to the drop-down menu in the nav bar. We have included an if else statement to check if a profile exists for a user. If there is no profile, then the menu items to view and edit the profile are not displayed:



Create a superuser account, run the server, log into Django admin, and add a profile for the user you created earlier.

Run the server and access the home page. Log in and try the View Profile and Edit Profile options. Try and change the favourite author in the Edit Profile menu and view the profile again to make sure that the update has happened.



Run the following git commands to update the local and remote repositories:

git add -A

git commit -m “lab 9 Part 1 commit”

git push -u origin main