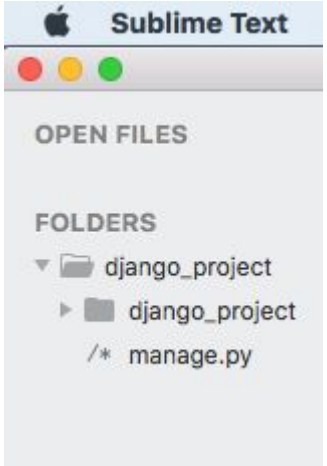
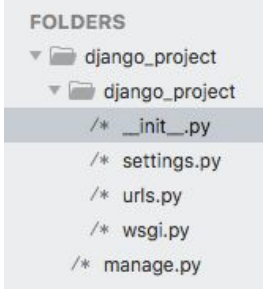
$ pip install django
$ python -m django --version
$ django-admin

(base) users-MBP:desktop user$ django-admin startproject django_project
$ cd django_project



$ brew install tree
$ tree

```
.
├── django_project
│   ├── __init__.py
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── manage.py
```



just tells python, that this is a "Python Package"

settings.py we can change different settings and configurations

```
SECRET_KEY = 'hzxv+@)f**e*l6
```

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.content
```

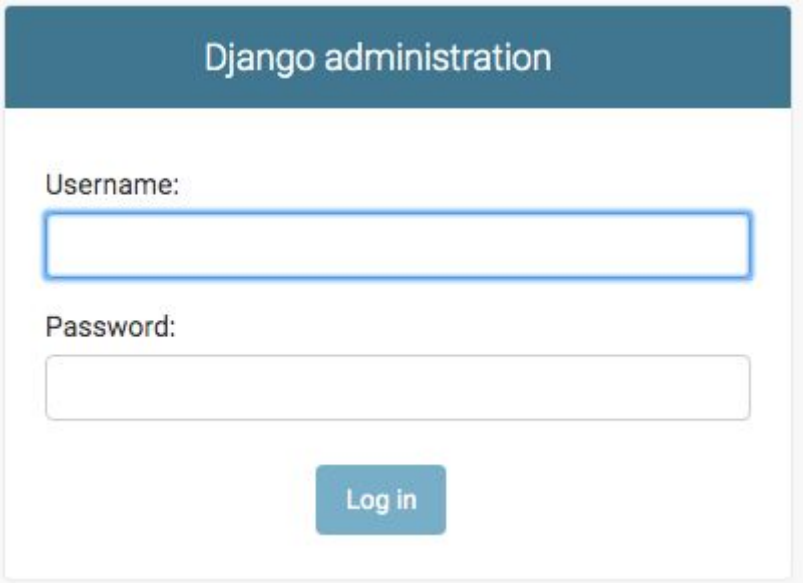
```
DATABASES = {
    'default': {
        'ENGINE': 'dj
        'NAME': os.pa
```

urls.py setup the mapping from certain URLs to where we send the user

```
urlpatterns = [
    path('admin/', admin.site.urls),
]
```

(base) users-MBP:django_project user$ python manage.py runserver

http://127.0.0.1:8000/ == http://localhost:8000/
http://localhost:8000/admin/login/?next=/admin/ == http://localhost:8000/admin

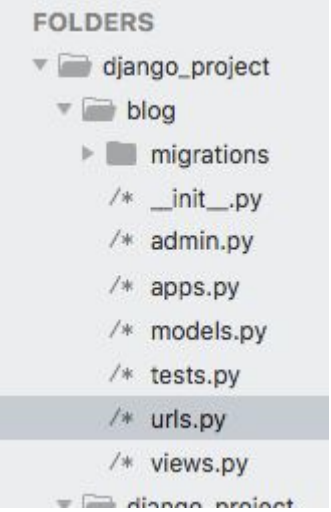


$ python manage.py startapp blog
$ tree

```
.
├── blog
│   ├── __init__.py
│   ├── admin.py
│   ├── apps.py
│   ├── migrations
│   │   └── __init__.py
│   ├── models.py
│   ├── tests.py
│   └── views.py
├── db.sqlite3
├── django_project
│   ├── __init__.py
│   ├── __pycache__
│   │   ├── __init__.cpython-37.pyc
│   │   ├── settings.cpython-37.pyc
│   │   ├── urls.cpython-37.pyc
│   │   └── wsgi.cpython-37.pyc
│   ├── settings.py
│   ├── urls.py
│   └── wsgi.py
└── manage.py
```

from /blog/views.py change like this:

```
from django.shortcuts import render
from django.http import HttpResponse

def home(request):
        return HttpResponse('<h1>Blog Home</h1>')

def about(request):
        return HttpResponse('<h1>Blog About</h1>')
```



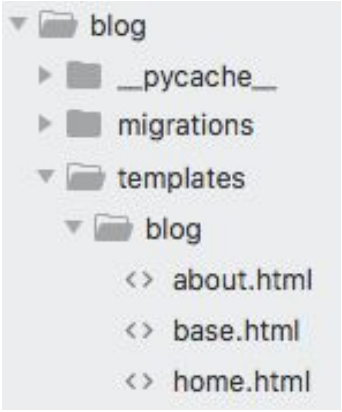
Create a new file in blog folder and named urls.py
Then go to /blog/urls.py and type code like this:

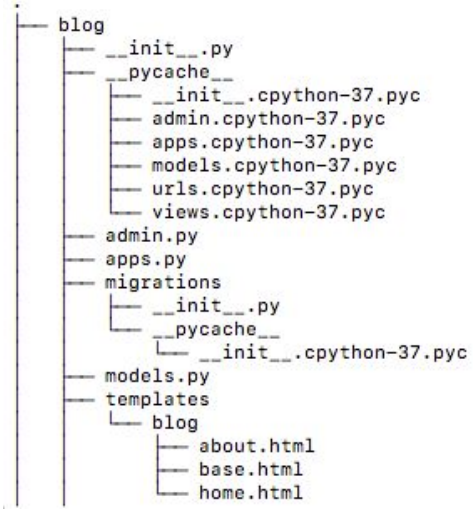
```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.home, name='blog-home'),
    path('about/', views.about, name='blog-about'),
]
```

Now go to /django_project/urls.py and change like this:

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('blog.urls')),
]
```

Now create a new folder in the blog folder like this /blog/templates then create a new folder in the template named blog. Finally in the /blog/templates/blog create 3 files(*base.html*, *home.html* and *about.html*).





blog/templates/blog/***base.html***

```
{% load static%}
<!DOCTYPE html>
<html>
<head>

<!-- Required meta tags -->
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

   <!-- Bootstrap CSS -->
   <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous">

   <link rel="stylesheet" type="text/css" href="{% static 'blog/main.css' %}">

         {% if title %}
                  <title>Django Website - {{ title }}</title>
         {% else %}
                  <title>Django Website</title>
         {% endif %}
</head>
<body>
         <header class="site-header">
     <nav class="navbar navbar-expand-md navbar-dark bg-steel fixed-top">
       <div class="container">
         <a class="navbar-brand mr-4" href="{% url 'blog-home' %}">Django Blog</a>
         <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarToggle"
aria-controls="navbarToggle" aria-expanded="false" aria-label="Toggle navigation">
         <span class="navbar-toggler-icon"></span>
         </button>
         <div class="collapse navbar-collapse" id="navbarToggle">
           <div class="navbar-nav mr-auto">
             <a class="nav-item nav-link" href="{% url 'blog-home' %}">Home</a>
             <a class="nav-item nav-link" href="{% url 'blog-about' %}">About</a>
           </div>
           <!-- Navbar Right Side -->
           <div class="navbar-nav">
             <a class="nav-item nav-link" href="#">Login</a>
             <a class="nav-item nav-link" href="#">Register</a>
           </div>
         </div>
       </div>
     </nav>
   </header>
   <main role="main" class="container">
     <div class="row">
       <div class="col-md-8">
```

```
        {% if messages %}
          {% for message in messages %}
            <div class="alert alert-{{ message.tags }}">
              {{ message }}
            </div>
          {% endfor %}
        {% endif %}
        {% block content %}{% endblock %}
      </div>
      <div class="col-md-4">
        <div class="content-section">
          <h3>Our Sidebar</h3>
          <p class='text-muted'>You can put any information here you'd like.
            <ul class="list-group">
              <li class="list-group-item list-group-item-light">Latest Posts</li>
              <li class="list-group-item list-group-item-light">Announcements</li>
              <li class="list-group-item list-group-item-light">Calendars</li>
              <li class="list-group-item list-group-item-light">etc</li>
            </ul>
          </p>
        </div>
      </div>
    </div>
  </main>

          <!-- Optional JavaScript -->
  <!-- jQuery first, then Popper.js, then Bootstrap JS -->
  <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js"
integrity="sha384-J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJoZ+n"
crossorigin="anonymous"></script>
  <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
crossorigin="anonymous"></script>
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"
integrity="sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6"
crossorigin="anonymous"></script>

</body>
</html>
```

blog/templates/blog/***home.html***

```
{% extends "blog/base.html" %}
{% block content %}
  {% for post in posts %}
      <article class="media content-section">
        <div class="media-body">
```

```
            <div class="article-metadata">
              <a class="mr-2" href="#">{{ post.author }}</a>
              <small class="text-muted">{{ post.date_posted }}</small>
            </div>
            <h2><a class="article-title" href="#">{{ post.title }}</a></h2>
            <p class="article-content">{{ post.content }}</p>
          </div>
        </article>
    {% endfor %}
{% endblock content %}
```

blog/templates/blog/***about.html***

```
{% extends "blog/base.html" %}
{% block content %}
            <h1>About Page</h1>
{% endblock content%}
```

Now go to ***apps.py*** and copy this: BlogConfig

Then go to the ***settings***/INSTALLED_APPS = [

```
    'blog.apps.BlogConfig',
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
```

 # class BlogConfig(AppConfig):

 # Add this line : 'blog.apps.BlogConfig',

Open ***views.py*** and change this function:

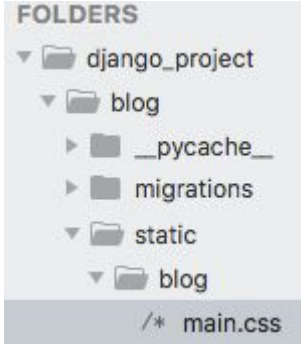
 # Old function: def home(request):

return HttpResponse('<h1>Blog Home</h1>')

```
from django.shortcuts import render

posts = [
        {
                'author': 'Farshid',
                'title': 'Blog Post 1',
                'content': 'First post content',
                'date_posted': 'August 29, 2019'
        },
        {
                'author': 'Elham',
                'title': 'Blog Post 2',
                'content': 'Second post content',
                'date_posted': 'August 30, 2019'
        }
```

```
]

def home(request):
	context = {
		'posts': posts
	}
	return render(request, 'blog/home.html', context)

def about(request):
	return render(request, 'blog/about.html')
```

Now create a new folder in the ***blog*** named ***static***, then create another new folder in the static named ***blog***. Finally create a new file in the blog named ***main.css***.



Copy these codes into the ***main.css***.

```
body {
  background: #fafafa;
  color: #333333;
  margin-top: 5rem;
}

h1, h2, h3, h4, h5, h6 {
  color: #444444;
}

ul {
  margin: 0;
}

.bg-steel {
  background-color: #5f788a;
}

.site-header .navbar-nav .nav-link {
  color: #cbd5db;
}

.site-header .navbar-nav .nav-link:hover {
  color: #ffffff;
```

```
}

.site-header .navbar-nav .nav-link.active {
  font-weight: 500;
}

.content-section {
  background: #ffffff;
  padding: 10px 20px;
  border: 1px solid #dddddd;
  border-radius: 3px;
  margin-bottom: 20px;
}

.article-title {
  color: #444444;
}

a.article-title:hover {
  color: #428bca;
  text-decoration: none;
}

.article-content {
  white-space: pre-line;
}

.article-img {
  height: 65px;
  width: 65px;
  margin-right: 16px;
}

.article-metadata {
  padding-bottom: 1px;
  margin-bottom: 4px;
  border-bottom: 1px solid #e3e3e3
}

.article-metadata a:hover {
  color: #333;
  text-decoration: none;
}

.article-svg {
  width: 25px;
  height: 25px;
  vertical-align: middle;
}
```

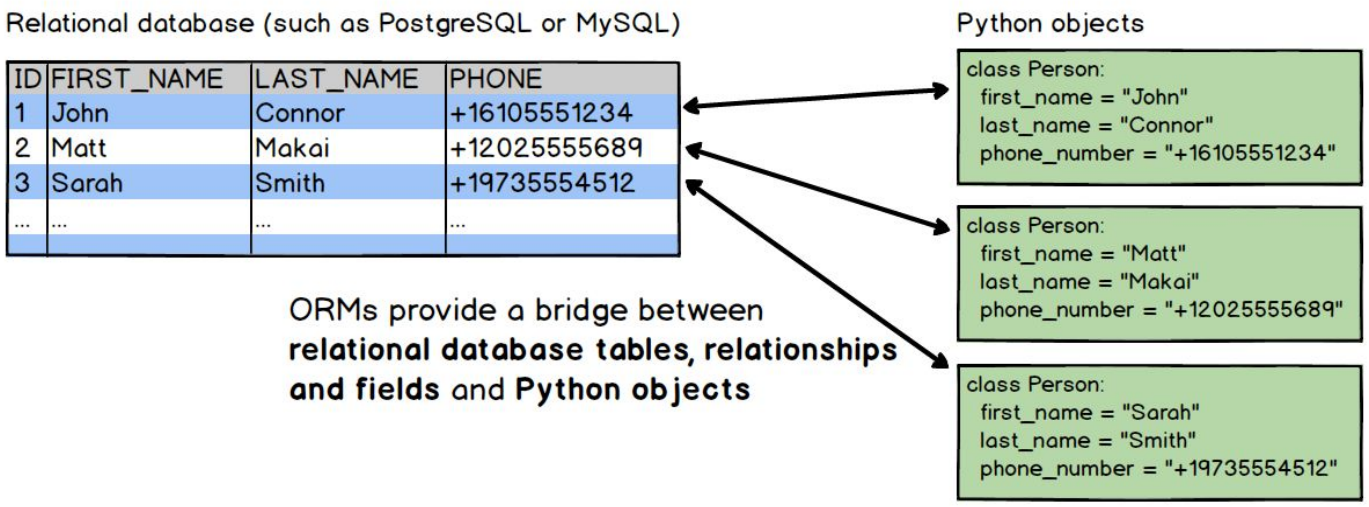
```
.account-img {
  height: 125px;
  width: 125px;
  margin-right: 20px;
  margin-bottom: 16px;
}

.account-heading {
  font-size: 2.5rem;
}
```

First stop server by press ***control^c*** in the terminal $
After that type in terminal like this:
$ python manage.py makemigrations
$ python manage.py migrate
$ python manage.py createsuperuser

Username (leave blank to use 'user'): ***farshid***
Email address: ***farshid.farnia@gmail.com***
Password: ********
Password (again): ********
Superuser created successfully

$ python manage.py runserver

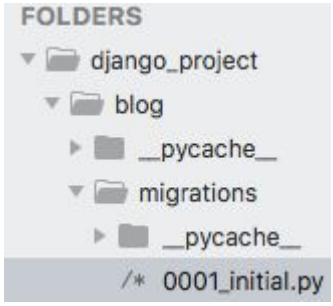
Database and Migrations



models.py

```
from django.db import models
from django.utils import timezone
from django.contrib.auth.models import User

class Post(models.Model):
        title = models.CharField(max_length=100)
        content = models.TextField()
        date_posted = models.DateTimeField(default=timezone.now)
        author = models.ForeignKey(User, on_delete=models.CASCADE)

    def __str__(self):
        return self.title
```

```
$ python manage.py makemigrations
Migrations for 'blog':
  blog/migrations/0001_initial.py
    - Create model Post
```




```
$ python manage.py sqlmigrate blog 0001
$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, blog, contenttypes, sessions
```

```
Running migrations:
  Applying blog.0001_initial... OK
```

$ python manage.py shell

```
In [1]: from blog.models import Post
In [2]: from django.contrib.auth.models import User
In [3]: User.objects.all()
Out[3]: <QuerySet [<User: farshid>]>
In [4]: User.objects.first()
Out[4]: <User: farshid>
In [5]: User.objects.filter(username='farshid')
Out[5]: <QuerySet [<User: farshid>]>
In [6]: user = User.objects.filter(username='farshid').first()
                                    # this is the same =>          user = User.objects.get(id=1)
In [7]: user
Out[7]: <User: farshid>
In [8]: user.id                     # this is the same =>    user.pk
Out[8]: 1
In [9]: Post.objects.all()
Out[9]: <QuerySet [ ]>
In [10]: post_1 = Post(title='Blog 1', content='First Post Content!', author=user)
In [11]: post_1.save()              # for see the result, type=> Post.objects.all()
In [12]: user.post_set.create(title='Blog 2', content='Second Post Content!')
Out[12]: <Post: Post object (2)>
In [13]: user.post_set.create(title='Blog 3', content='Third Post Content!')
Out[13]: <Post: Post object (3)>     # for see the result, type=> Post.objects.all()
In [14]: exit()
```

Back to the sublime text editor and use the database queries, use this real data that was added to the database instead of the dummy data that we have right now. So we can edit this post data within the admin page of our site.

Now you can delete dummy data dictionaries, from ***views.py*** :

```
posts = [
        {
                'author': 'Farshid',
                'title': 'Blog Post 1',
                'content': 'First post content',
                'date_posted': 'August 29, 2019'
        },
        {
                'author': 'Elham',
                'title': 'Blog Post 2',
                'content': 'Second post content',
                'date_posted': 'August 30, 2019'
        }
]
```

/blog/views.py

```
from .models import Post

def home(request):
        context = {
                'posts': Post.objects.all()
        }
        return render(request, 'blog/home.html', context)
```

$ python manage.py runserver

Go to ***home.html*** and change this line:

```
<small class="text-muted">{{ post.date_posted |date:"F d,Y" }}</small>
```

Ok, now go to ***admin.py*** page and add these lines:

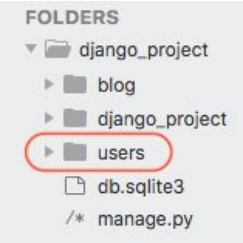
```
from django.contrib import admin

from .models import Post
admin.site.register(Post)
```

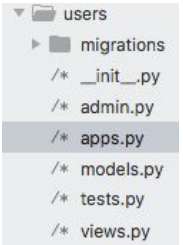
Then see the result in http://localhost:8000/admin :

User Registration

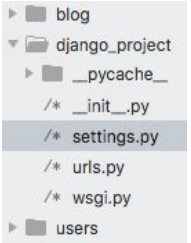
First of all, let's go ahead and create a new app.

django_project user$ python manage.py startapp users






```
class UsersConfig(AppConfig):
    name = 'users'
```




```
INSTALLED_APPS = [
    'blog.apps.BlogConfig',
    'users.apps.UsersConfig',
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
```

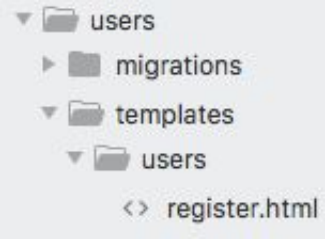
Change the ***/users/views.py*** like this:

```
views.py

def register(request):
    if request.method == 'POST':
        form = UserCreationForm(request.POST)
        if form.is_valid():
            username = form.cleaned_data.get('username')
            messages.success(request, f'Account created for {username}!')
            return redirect('blog-home')
    else:
        form = UserCreationForm()
    return render(request, 'users/register.html', {'form': form})
```

def register(request):

```
    if request.method == 'POST':
        form = UserCreationForm(request.POST)
        if form.is_valid():
            form.save()
            username = form.cleaned_data.get('username')
            messages.success(request, f'Account created for {username}!')
            return redirect('blog-home')
    else:
        form = UserCreationForm()
    return render(request, 'users/register.html', {'form': form})
```

Now create a new folder in the ***users*** named ***templates***, then create another new folder in the templates named ***users***. Finally create a new file in the blog named ***register.html***.



After that added these lines into the ***register.html***:

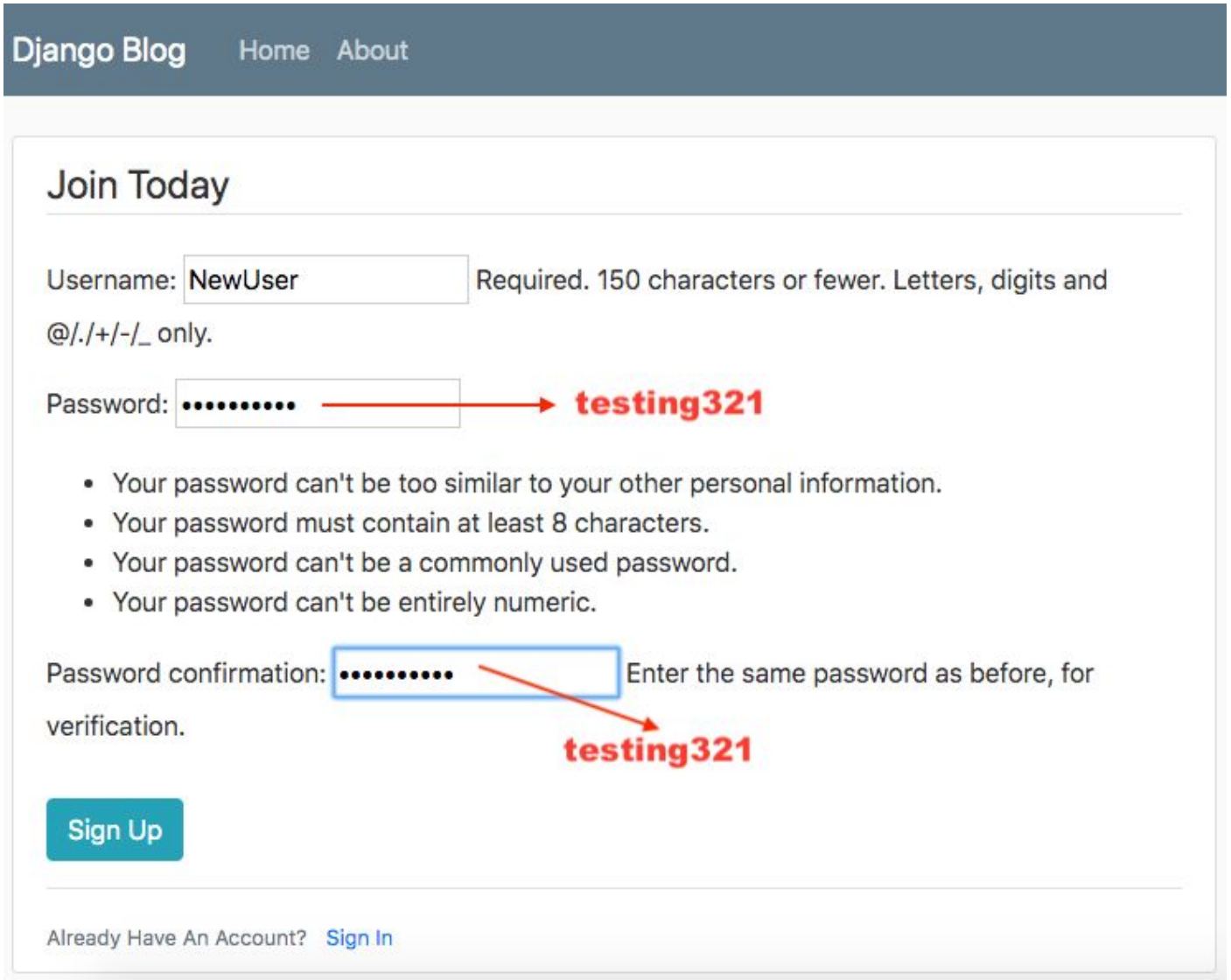
```
{% extends "blog/base.html" %}
{% block content %}
    <div class="content-section">
        <form method="POST">
            {% csrf_token %}
            <fieldset class="form-group">
                <legend class="border-bottom mb-4">Join Today</legend>
                {{ form.as_p }}
            </fieldset>
            <div class="form-group">
                <button class="btn btn-outline-info" type="submit">Sign Up</button>
            </div>
        </form>
        <div class="border-top pt-3">
            <small class="text-muted">
                Already Have An Account? <a class="ml-2" href="#">Sign In</a>
            </small>
        </div>
    </div>
{% endblock content%}
```

Well done!

In this time you should be added URL path of this form into the ***/django_project/urls.py*** :

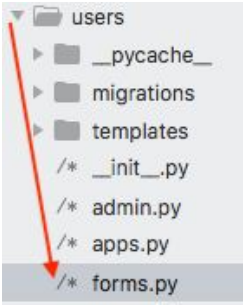
```
from django.contrib import admin
from django.urls import path, include
from users import views as user_views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('register/', user_views.register, name='register'),
    path('', include('blog.urls')),
]
```

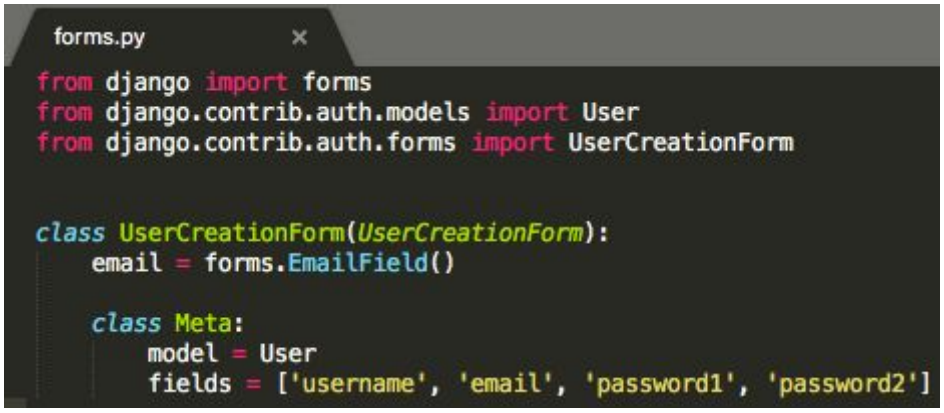
$ python manage.py runserver
http://localhost:8000/register/



If you want to see all users already exist, you should go to the link below:
http://localhost:8000/admin/auth/user/

We need to create a new form (user creation form). So to do this we're going to actually need to create a new form that inherits from our user creation form. So to do this, we need to first create a file where we can put these new forms.
So we create a new file in our user application directory and I'll call this *forms.py* in the users folder.






```
from django import forms
from django.contrib.auth.models import User
from django.contrib.auth.forms import UserCreationForm


class UserCreationForm(UserCreationForm):
        email = forms.EmailField()

        class Meta:
                model = User
                fields = ['username', 'email', 'password1', 'password2']
```

Now go back to ***users/views.py*** and added new path, like this:

```
from .forms import UserCreationForm
```

And delete unnecessary path :

from django.contrib.auth.forms import UserCreationForm

Now we should see "in this page (http://localhost:8000/register/)":

Username:

@/./+/-/_ only.

Email:

Password:

Ok, for better result styling forms, we install django-crispy-forms:
$ pip install django-crispy-forms

After installed crispy_forms, added to INSTALLED_APPS in the ***settings.py:***

```
    'users.apps.UsersConfig',
    'crispy_forms',
    'django.contrib.admin',
```

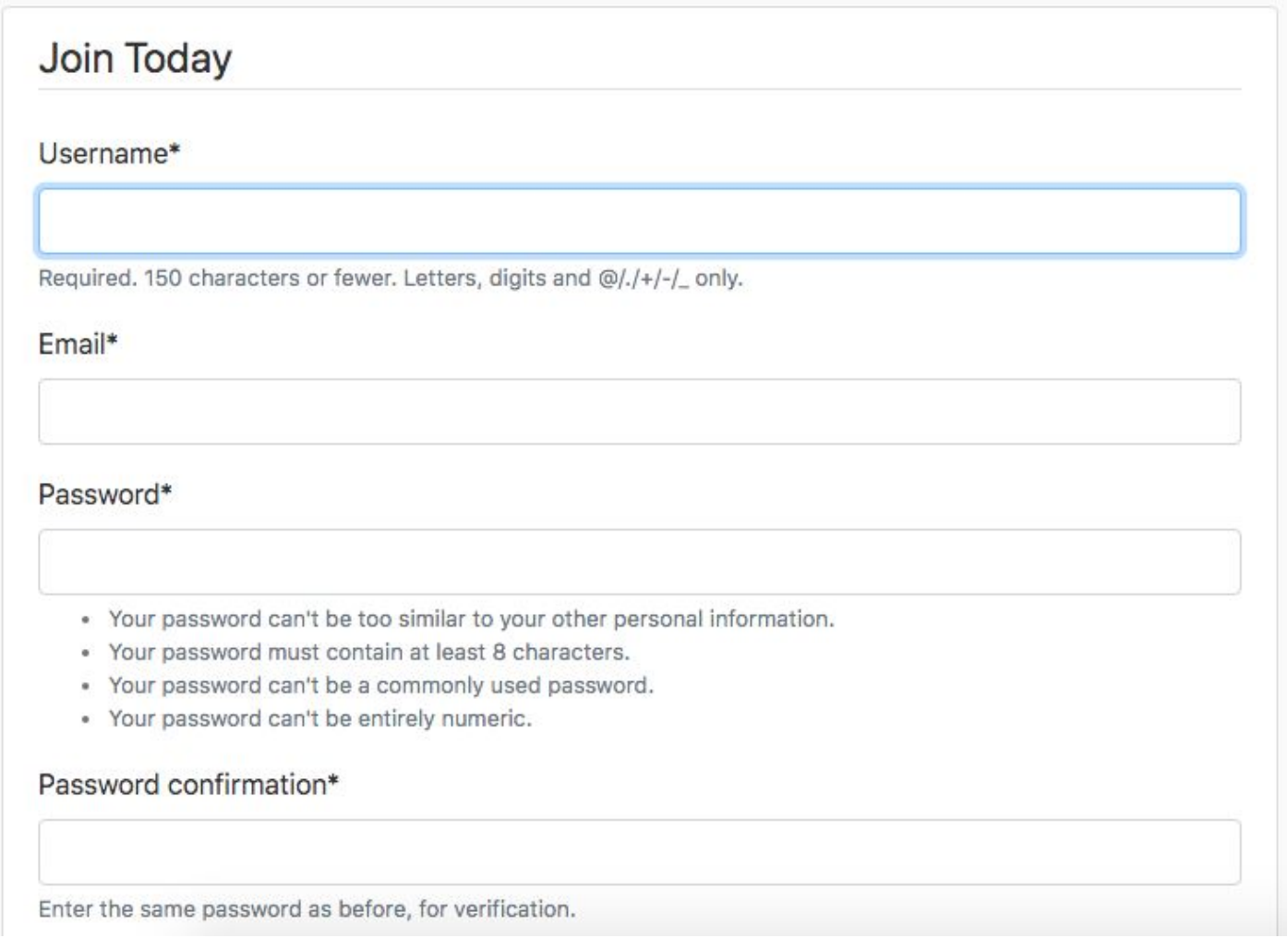
Alright , now go to the very bottom of our ***settings*** file and add this line:

```
STATIC_URL = '/static/'
CRISPY_TEMPLATE_PACK = 'bootstrap4'
```

Next open up ***register.html*** page and added new line, like this:

```
{% extends "blog/base.html" %}
{% load crispy_forms_tags %}
{% block content %}
# now changing this line :
{{ form.as_p }}          =>        {{ form|crispy }}
```

Our page (http://localhost:8000/register/) should be changed like this:

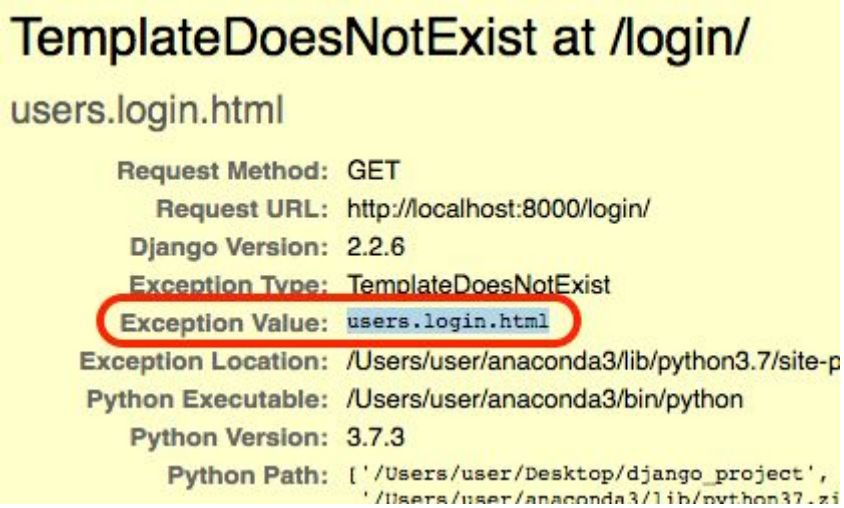


Login and Logout System

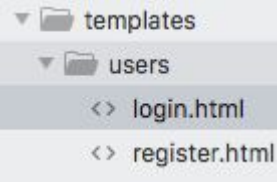
Go to ***urls.py*** and add these lines:

```
from django.contrib import admin
from django.contrib.auth import views as auth_views
. . .
    path('register/', user_views.register, name='register'),
    path('login/', auth_views.LoginView.as_view(template_name='users/login.html'), name='login'),
    path('logout/', auth_views.LogoutView.as_view(template_name='users/logout.html'), name='logout'),
```

If you visited this link: http://localhost:8000/login/
You see this error, because we don't create login.html in the Template folder:



So, go to ***templates/users/*** and create a new file, named: ***login.html***



Now, go to ***register.html*** and copy all of the codes and paste that in the ***login.html*** but changing something like legend, button names and href linked:

```
                    <fieldset class="form-group">
                        <legend class="border-bottom mb-4">Log In</legend>
. . .
                    <div class="form-group">
                        <button class="btn btn-outline-info" type="submit">Login</button>
. . .
                    <small class="text-muted">
                        Need An Account? <a class="ml-2" href="{% url 'register' %}">Sign Up
Now</a>
```

Now go back to ***register.html*** and change this line:

```
            <small class="text-muted">
                Already Have An Account? <a class="ml-2" href="{% url 'login' %}">Sign In</a>
```

Welldone!
Now go ***settigs.py*** and wright this line in the below of page:

```
. . .
CRISPY_TEMPLATE_PACK = 'bootstrap4'

LOGIN_REDIRECT_URL = 'blog-home'
```

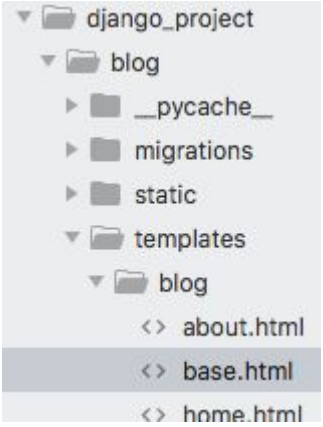
Then go to ***views.py*** and change these lines:

```
messages.success(request, f'Your account has been created! \nYou are able to log in')
return redirect('login')
```

After that, go to ***templates/users/*** and create a new file, named: ***logout.html*** and type codes like this:

```
{% extends "blog/base.html" %}
{% block content %}
        <h2>You have been logged out</h2>
        <div class="border-top pt-3">
                <small class="text-muted">
                        <a class="ml-2" href="{% url 'login' %}">Log In Again</a>
                </small>
        </div>
{% endblock content%}
```

Okey now we need to attach a register, login and logout to the navigation bar. So go to the ***base.html*** and changing the path:




```
        <!-- Navbar Right Side -->
        <div class="navbar-nav">
         {% if user.is_authenticated %}
           <a class="nav-item nav-link" href="{% url 'logout' %}">Logout</a>
         {% else %}
           <a class="nav-item nav-link" href="{% url 'login' %}">Login</a>
           <a class="nav-item nav-link" href="{% url 'register' %}">Register</a>
         {% endif %}
        </div>
```

Now we want anybody to be able to edit their own profile. So let's do something like that "create a page for users profile in ***users/views.py***":

```
from django.contrib import messages
from django.contrib.auth.decorators import login_required
. . .
def profile(request):
        return render(request, 'users/profile.html')
```

After that, create a new file "***profile.html***" in the ***users/templates/users***:

```
{% extends "blog/base.html" %}
{% load crispy_forms_tags %}
```

```
{% block content %}
        <h1>{{ user.username }}</h1>
{% endblock content%}
```

Next define a new path for user profile in the ***django_project/urls.py***:

```
. . .
   path('register/', user_views.register, name='register'),
   path('profile/', user_views.profile, name='profile'),
. . .
```

Next go to the ***base.html*** and added this line:

```
        {% if user.is_authenticated %}
          <a class="nav-item nav-link" href="{% url 'profile' %}">Profile</a>
. . .
```

Finally go to ***settings.py*** and added this line:

```
LOGIN_REDIRECT_URL = 'blog-home'
LOGIN_URL = 'login'
```

User Profile and Picture

First of all, go to ***models.py*** and define a class named Profile and picture and write a function for user profile, like this:

```
from django.db import models
from django.contrib.auth.models import User

class Profile(models.Model):
        user = models.OneToOneField(User, on_delete=models.CASCADE)
        image = models.ImageField(default='default.jpg', upload_to='profile_pics')

        def __str__(self):
                return f'{self.user.username} Profile'
```

Next go to the terminal and install pillow package(**Python Imaging Library**) after that run (**makemigrations**) and (**migrate**):
$ pip install pillow
$ python manage.py makemigrations
$ python manage.py migrate

Next go to ***admin.py*** import Profile and use it like this:

```
from django.contrib import admin
from .models import Profile

admin.site.register(Profile)
```

Now go to the terminal and run server:
$ python manage.py runserver

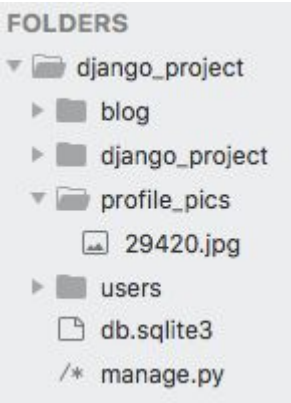
Alright go to the admin page (http://localhost:8000/admin/) then click the Profile and push ADD PROFILE + Now you are able to choose a user and set a picture for profile.

Run the python shell:

```
In [1]: from django.contrib.auth.models import User
In [2]: user = User.objects.filter(username='farshid').first()
In [3]: user.profile
Out[3]: <Profile: farshid Profile>
In [4]: user.profile.image
Out[4]: <ImageFieldFile: profile_pics/29420_K9Srsqf.jpg>
# you can use this commands :     >>>user.profile.image.width          >>>user.profile.image.url
In [5]: user = User.objects.filter(username='NewUser').first()
In [6]: user
Out[6]: <User: NewUser>
In [7]: exit()
```

$ python manage.py runserver

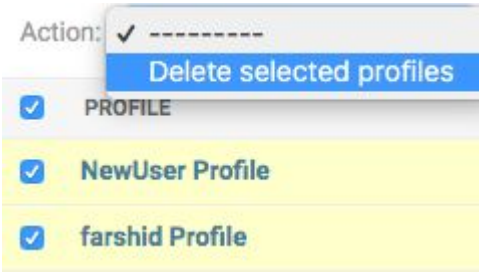
If you came back to your editor, see a new folder named “profile” :



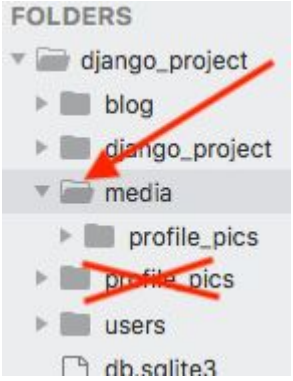
Okey, now go to ***settings.py*** and set media direction and media URLs:

```
STATIC_URL = '/static/'
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
MEDIA_URL = 'media'
```

Good job, now back to (http://localhost:8000/admin/users/profile/) and delete any users added before:



Then create a new profile (ADD PROFILE **+**) for users, and select an image.
If you back to editor “sublime text” see a new folder named media:



And delete the old folder (profile pics).

Now go to ***/users/profile.html*** and paste these lines:

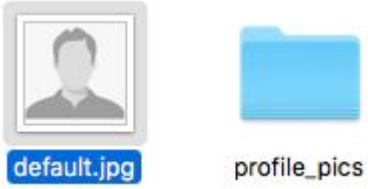
```
{% extends "blog/base.html" %}
{% load crispy_forms_tags %}
{% block content %}
  <div class="content-section">
    <div class="media">
      <img class="rounded-circle account-img" src="{{ user.profile.image.url }}">
      <div class="media-body">
        <h2 class="account-heading">{{ user.username }}</h2>
```

```
        <p class="text-secondary">{{ user.email }}</p>
      </div>
    </div>
    <!-- FORM HERE -->
  </div>
{% endblock content %}
```

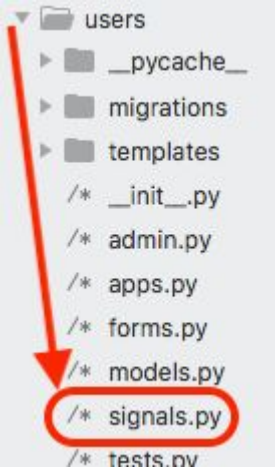
Open the urls.py and added new lines like this:

```
from django.contrib import admin
from django.contrib.auth import views as auth_views
from django.urls import path, include
from django.conf import settings
from django.conf.urls.static import static
from users import views as user_views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('register/', user_views.register, name='register'),
    path('profile/', user_views.profile, name='profile'),
    path('login/', auth_views.LoginView.as_view(template_name='users/login.html'), name='login'),
    path('logout/', auth_views.LogoutView.as_view(template_name='users/logout.html'),
name='logout'),
    path('', include('blog.urls')),
]

if settings.DEBUG:
    urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

For default image (any user don't set image for own profile) doing like this:
django_project/media/default.jpg



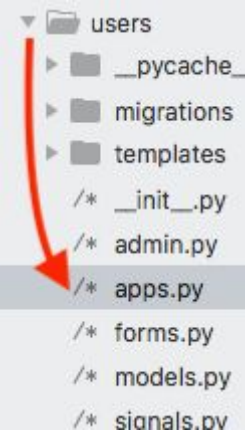
Well done!
Now create a new file in this route: ***django_project/users/signals.py***



So write these codes for saving the signals(sender, instance, ...):

```
from django.db.models.signals import post_save
from django.contrib.auth.models import User
from django.dispatch import receiver
from .models import Profile


@receiver(post_save, sender=User)
def create_profile(sender, instance, created, **kwargs):
    if created:
        Profile.objects.create(user=instance)


@receiver(post_save, sender=User)
def save_profile(sender, instance, **kwargs):
    instance.profile.save()
```

Okey, now go to ***apps.py*** and added these lines:




```
class UsersConfig(AppConfig):
    name = 'users'

    def ready(self):
        import users.signals
```

Update User Profile

First of all open ***forms.py*** and create a model form:

```
. . .
from django.contrib.auth.forms import UserCreationForm
from .models import Profile

. . .
                    fields = ['username', 'email', 'password1', 'password2']

class UserUpdateForm(forms.ModelForm):
    email = forms.EmailField()

    class Meta:
        model = User
        fields = ['username', 'email']


class ProfileUpdateForm(forms.ModelForm):
    class Meta:
        model = Profile
        fields = ['image']
```

Okey, now we want add and import created forms to ***views.py*** :

```
. . .
from .forms import UserCreationForm, UserUpdateForm, ProfileUpdateForm
. . .
@login_required
def profile(request):
    if request.method == 'POST':
        u_form = UserUpdateForm(request.POST, instance=request.user)
        p_form = ProfileUpdateForm(request.POST,
                                   request.FILES,
                                   instance=request.user.profile)
        if u_form.is_valid() and p_form.is_valid():
            u_form.save()
            p_form.save()
            messages.success(request, f'Your account has been updated!')
            return redirect('profile')

    else:
        u_form = UserUpdateForm(instance=request.user)
        p_form = ProfileUpdateForm(instance=request.user.profile)

    context = {
        'u_form': u_form,
        'p_form': p_form
    }

        return render(request, 'users/profile.html', context)
```

Now open up ***register.html*** and copy part of codes (<form> section) then open up ***profile.html*** and paste it behind the lasted after that, add user form {{ u_form|crispy }} and profile form {{ p_form|crispy }} finally added the enctype and changing the legend and button named:

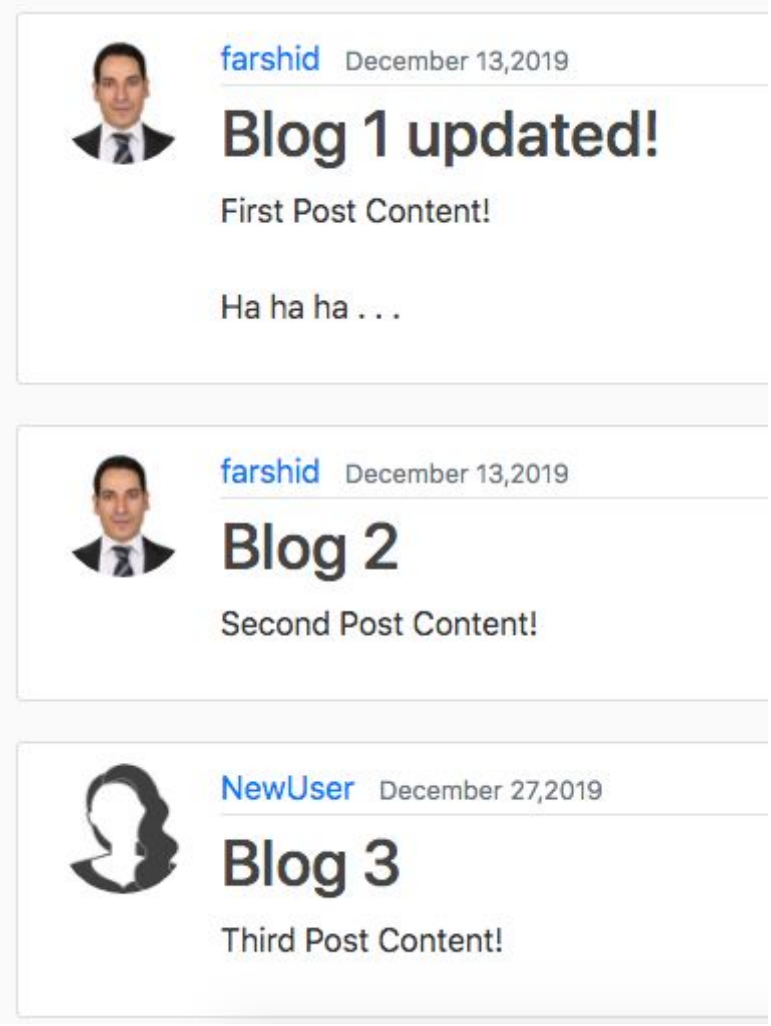
```
  </div>
    <form method="POST" enctype="multipart/form-data">
    {% csrf_token %}
    <fieldset class="form-group">
      <legend class="border-bottom mb-4">Profile Info</legend>
      {{ u_form|crispy }}
      {{ p_form|crispy }}
    </fieldset>
    <div class="form-group">
      <button class="btn btn-outline-info" type="submit">Update</button>
    </div>
  </form>
 </div>
{% endblock content %}
```

We will also want to know how to resize this image when it is uploaded to save space on our web server. For this purpose go to models.py and define a new function like this:

```
from django.contrib.auth.models import User
from PIL import Image
. . .
    def save(self):
        super().save()

        img = Image.open(self.image.path)

        if img.height > 300 or img.width > 300:
            output_size = (300, 300)
            img.thumbnail(output_size)
            img.save(self.image.path)
```

Now we want the ability to add a custom user image author beside each post. For this purpose, go to the ***home.html*** and added this line:

```
<article class="media content-section">
  <img class="rounded-circle article-img" src="{{ post.author.profile.image.url }}">
```



Create, Update, and Delete Posts

First of all import list view in ***blog/views.py*** :

```
from django.shortcuts import render
from django.views.generic import ListView
. . .
class PostListView(ListView):
        model = Post
```

Next go to ***blog/urls.py*** and add this path: