Django Codes

**Blog**

**views.py**

from django.shortcuts import render, redirect, get\_object\_or\_404

from .models import Post, Comment, Like

from django.contrib import messages

from .forms import PostCreationForm, CommentForm, PostUpdateForm

from django.contrib.auth.decorators import login\_required

# Create your views here.

def no\_of\_likes(post):

occurances = Like.objects.all().filter(post=post).count()

return occurances

def home(request):

post\_dict = Post.objects.all().order\_by('-date')

for post in post\_dict:

short\_view = post.content[:100]

post.content = short\_view

post.likes = no\_of\_likes(post)

context = {'posts': post\_dict}

return render(request, 'blog/home.html', context)

@login\_required

def compose(request):

if request.method == 'POST':

form = PostCreationForm(request.POST)

if form.is\_valid():

post\_data = form.save(commit=False)

post\_data.author = request.user

post\_data.save()

messages.success(request, f'Blog Posted Successfully.')

return redirect('blog-home')

else:

form = PostCreationForm()

return render(request, 'blog/create.html', {'form': form})

@login\_required

def update\_post(request, pk):

post\_update = get\_object\_or\_404(Post, id=pk)

if request.user == post\_update.author:

if request.method == 'POST':

form = PostUpdateForm(request.POST)

if form.is\_valid():

post\_update.title = form.cleaned\_data['title']

post\_update.content = form.cleaned\_data['content']

post\_update.save()

return redirect('/')

else:

form = PostUpdateForm(instance=Post.objects.get(id=pk))

return render(request, 'blog/update.html', {'form': form})

else:

return redirect('/')

def delete\_post(request, pk):

post\_update = get\_object\_or\_404(Post, id=pk)

if request.user == post\_update.author:

post\_update.delete()

return redirect('/')

else:

return redirect('/')

def detail(request, pk):

comment\_post = get\_object\_or\_404(Post, id=pk)

comment\_post.likes = no\_of\_likes(comment\_post)

if request.method == 'POST' and "like" not in request.POST:

form = CommentForm(request.POST)

print(form)

if form.is\_valid():

comment\_data = Comment(

comment=form.cleaned\_data['comment'],

author=request.user,

post=comment\_post

)

comment\_data.save()

messages.success(request, 'Comment Submitted.')

comments = Comment.objects.all().filter(post=comment\_post).order\_by('-date')

return render(

request,

'blog/detail.html',

{

'post': comment\_post,

'form': CommentForm(),

'comments': comments

}

)

elif request.method == 'POST' and "like" in request.POST:

form = CommentForm()

comments = Comment.objects.all().filter(

post=comment\_post).order\_by('-date'

)

like\_instance = Like(author=request.user, post=comment\_post)

for like\_element in Like.objects.all().filter(post=comment\_post):

if str(like\_element) == str(like\_instance):

Like.objects.filter(author=request.user, post=comment\_post).delete()

comment\_post.likes = no\_of\_likes(comment\_post)

return render(

request, 'blog/detail.html',

{

'post': comment\_post,

'form': form,

'comments': comments

}

)

like\_instance.save()

comment\_post.likes = no\_of\_likes(comment\_post)

return render(

request, 'blog/detail.html',

{

'post': comment\_post,

'form': form,

'comments': comments

}

)

else:

form = CommentForm()

comments = Comment.objects.all().filter(

post=comment\_post).order\_by('-date'

)

return render(

request, 'blog/detail.html',

{

'post': comment\_post,

'form': form,

'comments': comments

}

)

**urls.py**

from django.urls import path

from .views import home, compose, detail, update\_post, delete\_post

APP\_NAME = 'blog'

urlpatterns = [

path('', home, name='blog-home'),

path('create/', compose, name='blog-compose'),

path('detail/<int:pk>/', detail, name='blog-detail'),

path('update/<int:pk>/', update\_post, name='blog-update'),

path('delete/<int:pk>/', delete\_post, name='blog-delete')

]

---------------------------------------xxxxxxxxxxxxxxxxxxxxxxx----------------------------------------------

**models.py**

from django.db import models

from django.contrib.auth.models import User

# Create your models here.

class Post(models.Model):

author = models.ForeignKey(User, on\_delete=models.CASCADE)

content = models.TextField()

title = models.CharField(max\_length=100)

date = models.DateTimeField(auto\_now\_add=True)

def \_\_str\_\_(self):

return self.title

class Comment(models.Model):

author = models.ForeignKey(User, on\_delete=models.CASCADE)

comment = models.TextField(max\_length=100)

date = models.DateTimeField(auto\_now\_add=True)

post = models.ForeignKey(Post, on\_delete=models.CASCADE)

def \_\_str\_\_(self):

poster = self.author

return f"{poster.username}-{self.post.title}-{self.pk}"

class Like(models.Model):

author = models.ForeignKey(User, on\_delete=models.CASCADE)

post = models.ForeignKey(Post, on\_delete=models.CASCADE)

def \_\_str\_\_(self):

return f"{self.post.title}-{self.author.username}"

---------------------------------------xxxxxxxxxxxxxxxxxxxxxxx----------------------------------------------

**forms.py**

from django.forms import ModelForm

from django import forms

from .models import Post

from allauth.account.forms import LoginForm

from django.conf import settings

from django.http import HttpResponse

class PostCreationForm(ModelForm):

class Meta:

model = Post

fields = ['title', 'content']

class CommentForm(forms.Form):

comment = forms.CharField(

label='Post your Comment:',

max\_length=100,

widget=forms.Textarea(attrs={'rows': 4, 'cols': 15})

)

class PostUpdateForm(ModelForm):

class Meta:

model = Post

fields = ['title', 'content']

--------------------------------------------xxxxxxxxxxxxxxxxxxxxx--------------------------------------------

**Profiles**

**views.py**

from django.shortcuts import render, redirect

from .models import Profile, Post, Like

from django.contrib import messages

from django.contrib.auth.decorators import login\_required

from .forms import UserCreationForm, ProfileCreationForm

from PIL import Image

def home(request, username):

post\_dict = Post.objects.all().filter(author\_\_username=username).order\_by('-date')[:5]

for post in post\_dict:

short\_view = post.content[:100]

post.content = short\_view

post.likes = no\_of\_likes(post)

for profile in Profile.objects.all().filter(user\_\_username=username):

profile.update()

profile.save()

req\_prof = Profile.objects.all().filter(user\_\_username=username)

print(req\_prof)

context = {'profiles': req\_prof, 'posts': post\_dict}

return render(request, 'profiles/home.html', context)

@login\_required

def create(request):

try:

prof = request.user.profile

if request.method == 'POST':

u\_form = UserCreationForm(request.POST, instance=request.user)

p\_form = ProfileCreationForm(request.POST, request.FILES, instance=request.user.profile)

if u\_form.is\_valid() and p\_form.is\_valid():

usr\_profile\_img = p\_form.cleaned\_data['image']

u\_form.save()

prof = Profile.objects.get(user=request.user)

prof.image = usr\_profile\_img

prof.save()

img\_save(prof, request)

prof.save()

messages.success(request, f'Your account has been updated.')

return redirect('blog-home')

else:

u\_form = UserCreationForm(instance=request.user)

p\_form = ProfileCreationForm(instance=request.user.profile)

context = {

'u\_form': u\_form,

'p\_form': p\_form

}

return render(request, 'profiles/create.html', context)

except:

if request.method == 'POST':

u\_form = UserCreationForm(request.POST, instance=request.user)

p\_form = ProfileCreationForm(request.POST, request.FILES)

if u\_form.is\_valid() and p\_form.is\_valid():

usr\_profile\_img = p\_form.cleaned\_data['image']

u\_form.save()

prof = Profile(user=request.user, image=usr\_profile\_img)

prof.save()

img\_save(prof, request)

prof.save()

messages.success(request, f'Your account has been updated.')

return redirect('blog-home')

else:

u\_form = UserCreationForm(instance=request.user)

p\_form = ProfileCreationForm()

context = {

'u\_form': u\_form,

'p\_form': p\_form

}

return render(request, 'profiles/create.html', context)

def img\_save(prof, request):

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

if img.height != 300 or img.width != 300:

output\_size = (300, 300)

img.thumbnail(output\_size)

img.save(prof.image.path)

def no\_of\_likes(post):

occurances = Like.objects.all().filter(post=post).count()

return occurances

@login\_required

def base\_create(request):

try:

prof = request.user.profile

return redirect('/')

except:

if request.method == 'POST':

u\_form = UserCreationForm(request.POST, instance=request.user)

p\_form = ProfileCreationForm(request.POST, request.FILES)

if u\_form.is\_valid() and p\_form.is\_valid():

usr\_profile\_img = p\_form.cleaned\_data['image']

u\_form.save()

prof = Profile(user=request.user, image=usr\_profile\_img)

prof.save()

img\_save(prof, request)

prof.save()

messages.success(request, f'Your account has been updated.')

return redirect('blog-home')

else:

u\_form = UserCreationForm(instance=request.user)

p\_form = ProfileCreationForm()

context = {

'u\_form': u\_form,

'p\_form': p\_form

}

return render(request, 'profiles/create.html', context)

@login\_required

def my\_post(request, pk):

post\_dict = []

for post in Post.objects.all().order\_by('-date'):

if post.author == request.user:

short\_view = post.content[:100]

post.content = short\_view

post.likes = no\_of\_likes(post)

post\_dict.append(post)

context = {'posts': post\_dict, 'act': True}

return render(request, 'blog/home.html', context)

---------------------------------------------xxxxxxxxxxxxxxxxxxx----------------------------------------------

**urls.py**

from django.urls import path

from .views import home, create, my\_post, base\_create

APP\_NAME = 'profile'

urlpatterns = [

path('<username>', home, name='profile-home'),

path('create/', create, name='profile-create'),

path('first\_create/', base\_create, name='profile-base-create'),

path('posts/<int:pk>', my\_post, name='profile-posts')

# path('create/', compose, name='blog-compose'),

# path('detail/<int:pk>/', detail, name='blog-detail'),

]

---------------------------------------xxxxxxxxxxxxxxxxxxxxxxx----------------------------------------------

**models.py**

from blog.models import Post, Like

from django.contrib.auth.models import User

from django.db import models

# Create your models here.

class Profile(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE)

image = models.ImageField(default='default.jpg', upload\_to='profile\_pics')

post\_counter = models.PositiveIntegerField(default=0)

likes\_counter = models.PositiveIntegerField(default=0)

def update(self):

self.likes\_counter = Like.objects.all().filter(post\_\_author=self.user).count()

self.post\_counter = Post.objects.all().filter(author=self.user).count()

def \_\_str\_\_(self):

return f'{self.user.first\_name} Profile'

---------------------------------------xxxxxxxxxxxxxxxxxxxxxxx----------------------------------------------

**forms.py**

from django import forms

from django.contrib.auth.models import User

from .models import Profile

class UserCreationForm(forms.ModelForm):

email = forms.EmailField()

class Meta:

model = User

fields = ['username', 'email', 'first\_name', 'last\_name']

class ProfileCreationForm(forms.ModelForm):

class Meta:

model = Profile

fields = ['image']