**Subcribe Feature**

**urls.py**

app\_name = 'newsletter'

urlpatterns = [

    path('newsletter-subscribe/', newsletter\_subscribe, name='subscribe'),

    path('newsletter-unsubscribe/', newsletter\_unsubscribe, name='unsubscribe'),

    path('newsletter-unsubscribe-send/',

         send\_newsletter, name='send\_newsletter'),

]

**models.py**

class NewsLetterUserList(models.Model):

    email = models.EmailField(max\_length=254)

    subscription\_date = models.DateTimeField(auto\_now\_add=True)

    def \_\_str\_\_(self):

        return self.email

class SendEmailToNewsLetterUser(models.Model):

    EMAIL\_STATUS\_CHOICES = (

        ("Draft", "Draft"),

        ("Published", "Published")

    )

    subject = models.CharField(max\_length=250)

    body = models.TextField()

    status = models.CharField(max\_length=10, choices=EMAIL\_STATUS\_CHOICES)

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

    updated = models.DateTimeField(auto\_now=True)

    def \_\_str\_\_(self):

        return self.subject

**forms.py**

from django import forms

from .models import NewsLetterUserList, SendEmailToNewsLetterUser

from tinymce import TinyMCE

class TinyMCEWidget(TinyMCE):

    def use\_required\_attribute(self, \*args):

        return False

class NewsLetterUserListForm(forms.ModelForm):

    class Meta:

        model = NewsLetterUserList

        fields = ['email']

class SendEmailToNewsLetterUserForm(forms.ModelForm):

    body = forms.CharField(widget=TinyMCEWidget(

        attrs={'required': False, 'cols': 10, 'rows': 10})

    )

    class Meta:

        model = SendEmailToNewsLetterUser

        fields = ['subject', 'body', 'status']

**views.py**

from django.shortcuts import render, redirect, from django.conf import settings

from .models import NewsLetterUserList, SendEmailToNewsLetterUser

from .forms import NewsLetterUserListForm, SendEmailToNewsLetterUserForm

from django.contrib import messages

from django.core.mail import send\_mail, EmailMultiAlternatives

from django.template.loader import get\_template

from django.utils.safestring import mark\_safe

def newsletter\_subscribe(request):

    if request.method == 'POST':

        form = NewsLetterUserListForm(request.POST)

        if form.is\_valid():

            instance = form.save(commit=False)

            if NewsLetterUserList.objects.filter(email=instance.email).exists():

                messages.success(request, 'This email is already registered!')

            else:

                instance.save()

                subject, recipient\_list = 'Subscribed Successfully', [instance.email]

                from\_email = settings.EMAIL\_HOST\_USER

                with open(settings.BASE\_DIR + "/newsletter/templates/newsletter/email\_txt.txt") as f:

                    body = f.read()

                 message = EmailMultiAlternatives(subject=subject, body=body, from\_email=from\_email, to=recipient\_list)

                temp = get\_template('newsletter/subscribe-email.html').render()

                message.attach\_alternative(temp, 'text/html')

                message.send()

                 messages.success(request, 'You successfully subscribed to our newsletter, please check your email')

                return redirect('blog-home')

    else:

        form = NewsLetterUserListForm()

    return render(request, 'blogApp/home.html', {'subscribe\_form': form})

def newsletter\_unsubscribe(request):

    form = NewsLetterUserListForm(request.POST or None)

    if form.is\_valid():

        instance = form.save(commit=False)

        if NewsLetterUserList.objects.filter(email=instance.email).exists():

            NewsLetterUserList.objects.filter(email=instance.email).delete()

            messages.success(

                request, 'We\'re sad to see you go :(')

            return redirect('blog-home')

        else:

            messages.success(request, 'Email doesn\'t exists!')

    return render(request, 'newsletter/unsubscribe.html', {'unsubscribe\_form': form})

def send\_newsletter(request):

    form = SendEmailToNewsLetterUserForm(request.POST or None)

    if form.is\_valid():

        instance = form.save()

        if instance.status == 'Published':

            subject = instance.subject

            body = instance.body

            from\_email = settings.EMAIL\_HOST\_USER

            for newsletter\_obj in NewsLetterUserList.objects.all():

                msg = EmailMultiAlternatives(subject, body, from\_email, [newsletter\_obj.email])

                msg.attach\_alternative(body, "text/html")

                msg.send()

    return render(request, 'newsletter/newsletter-email.html', {'form': form})

**subscribe-email.html**

<!DOCTYPE html>

<html lang="en">

<head>

</head>

<body>

    <h1 style='color:red;'>Welcome to our website</h1>

    <p>Lorem ipsum dolor, sit amet consectetur adipisicing elit. Voluptatem voluptate voluptatibus sequi odit. Fu laboriosam libero eaque repellat natus modi reiciendis velit maxime omnis voluptatum inventore, placeat debitis voluptates eveniet?</p>

    <small>Click here to <a href="{% url 'newsletter:unsubscribe' %}">unsubscribe</a></small>

</body>

</html>

**unsubscribe.html**

{% extends 'blogApp/base.html' %}

{% load crispy\_forms\_tags %}

{% block main %}

<div class="content-section">

  <fieldset class="form-group">

    <legend class="border-bottom mb-4">Unsubscribe</legend>

    <form action="#" method="POST">

      {% csrf\_token %}

      {{ unsubscribe\_form|crispy }}

      <input type="submit" class="btn btn-outline-success" value="Unsubscribe" />

    </form>

  </fieldset>

</div>

{% endblock main %}

**newsletter-email.html**

{% extends 'blogApp/base.html' %}

{% load crispy\_forms\_tags %}

{% block main %}

    <div class="content-section">

        <form action="." method="POST">

            {% csrf\_token %}

            {{form|crispy}}

            <input type="submit" value="Send NewsLetter" class='btn btn-primary'>

        </form>

    </div>

{% endblock main %}