

Programacion web II

Segundo trabajo



Alumno : Gonzalo sampini

10/09/2024

Prof. y Lic. Julio César Casco

INTRODUCCIÓN

Se desarrolló un proyecto Django denominado **TableroMensajes2** el cual permite crear mensajes con un nombre de remitente y un nombre destinatario otra de las funciones implementadas son la de buscar los mensajes de un usuario en particular tanto como los recibidos como los enviados por el mismo y se le da la posibilidad de borrarlos. Se utilizó el patrón diseño modelo-vista-plantilla (MVT) utilizando vistas basadas en funciones (su gran mayoría) y vistas basadas en clases .

Estructura

DESARROLLO:

El proyecto cuenta con la funcionalidad de crear mensajes a través de un formulario basado en modelo Tablero el cual cuenta con los campos de remitente destinatario y texto. Además cuenta con la hora en la cual se guarda el mensaje pero ese campo se lo sacamos ya que lo hace internamente y no depende del usuario.

```
def home_view (request):  
  
    if request.method == 'POST':  
  
        form = TableroForm(request.POST)  
  
        if form.is_valid():  
  
            mensaje = form.save() # Guarda el objeto en la base de datos  
  
            return redirect('ver',mensaje_id= mensaje.id )  
  
        else:  
  
            form = TableroForm()  
  
    return render(request, 'home.html', {'form':form})
```

Cuenta también con la funcionalidad de buscador de mensajes tanto como de los enviados como de los recibidos en esta pantalla se pueden eliminar uno por uno los mensajes. para lograr esta busqueda

```
def ver_mensajes_view (request, usuario):

    mensajes_recividos = Tablero.objects.filter(destinatario=usuario)

    mensajes_enviados = Tablero.objects.filter(remitente=usuario)

    mensajes = []

    for msj in mensajes_recividos:

        mensaje = {'usuario_2': msj.remitente, 'texto': msj.texto, 'tipo':
"recivido", 'id': msj.id}

        mensajes.append(mensaje)

    for msj in mensajes_enviados:

        mensaje = {'usuario_2': msj.destinatario, 'texto': msj.texto,
'tipo': "enviado", 'id': msj.id}

        mensajes.append(mensaje)

    return render (request, 'ver_mensajes.html', {'mensajes': mensajes,
'usuario':usuario})
```

esto se encarga de filtrar los mensajes por determinado usuario guardandolos en una lista y mostrandolos en ver_mensajes.html

Buscar mensajes recibe la peticion y devuelve la lista de mensajes con la opcion de eliminar individualmente

```

<p>Buscar mensajes</p>

<form action="{% url 'ver_mensajes' %}" method="POST">

    <input type = "text" name="usuario" required value="{% usuario %}">

    {% csrf_token %}

    <button type="submit">Buscar</button>

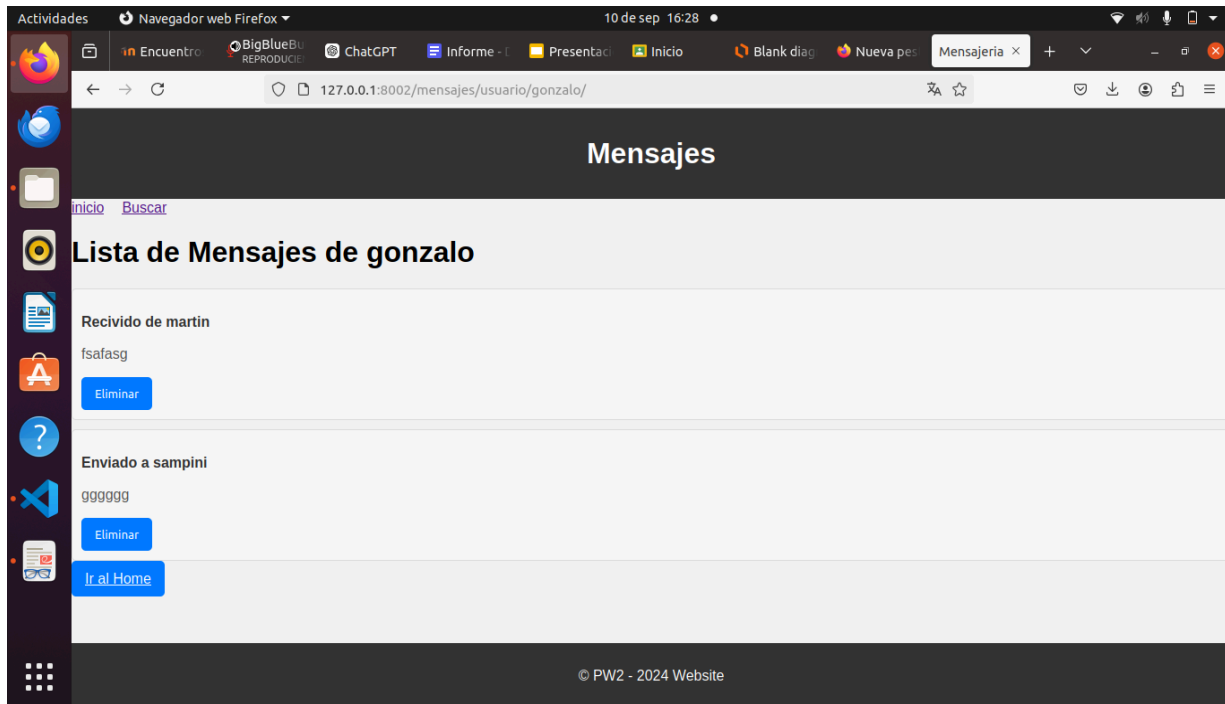
</form>

{% endblock %}

```

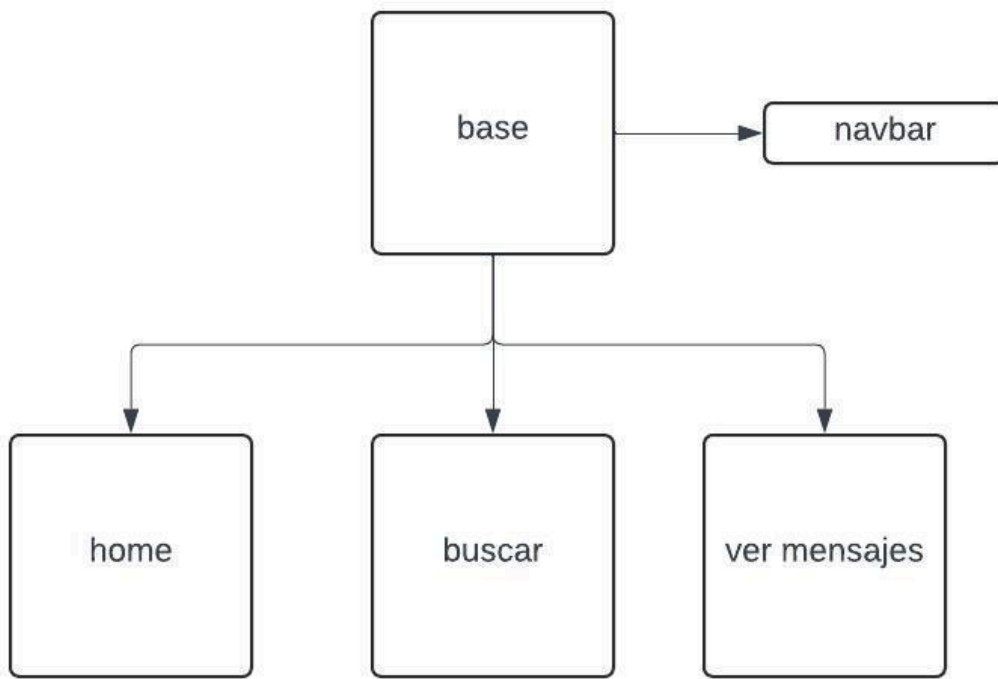
vista buscar





```
class MensajeDeleteView(DeleteView):  
    model = Tablero  
  
    success_url = reverse_lazy('ver_mensajes')
```

HERENCIA Y REUTILIZACION DE PLANTILLAS



dtl

para el navbar

```
<li style="margin-right: 20px;"><a href="{% url 'home' %}">inicio</a></li>

    <li style="margin-right: 20px;"><a href="{% url 'buscar_mensajes' %}">Buscar</a></li>
```

para tokens de seguridad en formularios

```
        <input type="hidden" name="usuario" value="{% usuario %}">

        {% csrf_token %}

        <button type="submit" class="btn btn-danger">Eliminar</button>

    </form>
```

para incluir estilo css al proyecto a través de la carpeta static/css/style

```
<link rel="stylesheet" href="{% static 'css/styles.css' %}">

</head>
```

MODELOS:

El modelo utilizado fue

```
from django.db import models

# Create your models here.

class Tablero(models.Model):

    texto = models.TextField()

    remitente = models.CharField(max_length=50)

    destinatario = models.CharField(max_length=50)

    def __str__(self):

        return self.remitente
```

de este modelo se creo un formulario basado en modelo.

```
from django import forms

from .models import Tablero

class TableroForm(forms.ModelForm):

    class Meta:

        model = Tablero

        fields = ['remitente', 'destinatario', 'texto']
```

DIFICULTADES:

Problemas encontrados : comunicacion entre las vistas y las plantillas .

el trabajo lo puede realizar con muchos videos tutoriales, y yendo a clases particulares que me explicaran un poco más todo esto.

Uno de los problemas es la falta de tiempo que tengo ya que cualquier contratiempo que me surja corro contra el reloj.


CONCLUSION:

Se pudo llegar a cumplir con los objetivos del práctico ya que me afiance un poco más en el tema , con algunos contratiempos pero se pudieron solucionar .El trabajo podria tener un mejor resultado.

REFERENCIAS:

Casco, Julio. (2024). *Plantillas django* [Apunte de clase]. Universidad del chubut.

Casco, Julio. (2024). *Patrones mvt y mvc* [Apunte de clase]. Universidad del chubut.

Generación de formularios basados en modelos con django 4.1 y Crispy-Forms [video] youtube  Generación de formularios basados en modelos con django 4.1 y Crispy-Fo...

Django Software Foundation. (2024). working whits forms. Django Documentation.
<https://docs.djangoproject.com/es/5.1/topics/forms/>