

## Aplicación “registration”

1. Crear la aplicación “registration”
2. En el archivo settings
  - a. Agregarla al inicio de las aplicaciones con la finalidad de que sea la de mayor prioridad
  - b. Colocar los redirects

```
LOGIN_REDIRECT_URL = 'core:home'
LOGOUT_REDIRECT_URL = 'core:home'
```

3. Dentro de la aplicación registration, crear el archivo login.html

```
<> login.html U X
DAW0822 > Practicas > solutions > django > webrestaurant > registration > templates > registration > <> login.html
1  {% extends "core/base.html" %}
2  {% block content %}
3  {% load static %}
4  <div class="container">
5      <div class="about-heading-content mbtm">
6          <div class="row">
7              <div class="col-xl-9 col-lg-10 mx-auto">
8                  <div class="bg-faded rounded p-5 forced">
9                      <h2 class="section-heading mb-4">
10                         Autenticación
11                     </h2>
12                     <div class="section-content">
13                         <form action="" method="post">{% csrf_token %}
14                             {% if form.non_field_errors %}
15                             <p style="color:red">Usuario o contraseña incorrectos, prueba de nuevo.</p>
16                             {% endif %}
17                         <p>
18                             <input type="text" name="username" autofocus maxlength="254" required
19                             id="id_username" class="form-control" placeholder="Nombre de usuario"/>
20                         </p>
21                         <p>
22                             <input type="password" name="password" required
23                             id="id_password" class="form-control" placeholder="Contraseña"/>
24                         </p>
25                         <p><input type="submit" class="btn btn-primary btn-block" value="Acceder"></p>
26                     </form>
27                 </div>
28             </div>
29         </div>
30     </div>
31 </div>
32 </div>
33 {% endblock %}
```

- a.
4. En el archivo base.html

```
</li>
{% if not request.user.is_authenticated %}
<li class="nav-item">
    <a class="nav-link" href="{% url 'login' %}">Acceder</a>
</li>
{% else %}
<li class="nav-item">
    <form action="{% url 'logout' %}" method="post">
        {% csrf_token %}
        <button class='btn btn-outline-secondary btn-sm' type="submit">Salir</button>
    </form>
</li>
{% endif %}
```

- a.
5. En el archivo urls.py

```

12 urlpatterns = [
13     path('accounts/', include('django.contrib.auth.urls')),
14     path('admin/', admin.site.urls),
15     path('', include(core_urlpatterns)),

```

a.

6. Verificar funcionamiento de login y logout

## Registro de usuarios

1. En el archivo webrestaurante/urls.py

```

12 urlpatterns = [
13     path('accounts/', include('django.contrib.auth.urls')),
14     path('accounts/', include('registration.urls')),

```

a.

2. Crear el archivo urls.py

```

1 from django.urls import path
2 from .views import SignUpView
3
4 urlpatterns = [
5     path('signup/', SignUpView.as_view(), name='signup' ),
6 ]

```

a.

3. Crear el archivo signup.html

```

DAW0822 > Practicas > solutions > django > webrestaurante > registration > templates > registration > <> signup.html
1  {% extends "core/base.html" %}
2  {% block contentStyle %}
3      label{display:none}
4  {%endblock%}
5  {% block content %}
6  {% load static %}
7  <div class="container">
8      <div class="about-heading-content mbtm">
9          <div class="row">
10             <div class="col-xl-9 col-lg-10 mx-auto">
11                 <div class="bg-faded rounded p-5 forced">
12                     <h2 class="section-heading mb-4">
13                         Registro
14                     </h2>
15                     <div class="section-content">
16                         <form action="" method="post">{% csrf_token %}
17                             {{form.as_p}}
18                             <p><input type="submit" class="btn btn-primary btn-block" value="Confirmar"></p>
19                         </form>
20                     </div>
21                 </div>
22             </div>
23         </div>
24     </div>
25 </div>
26 {% endblock %}

```

a.

4. Para dar formato a la forma, es necesario hacerlo en tiempo real en el archivo views.py

```

DAW0822 > Practicas > solutions > django > webrestaurante > registration > views.py
1  from django.views.generic.edit import CreateView
2  from django.contrib.auth.forms import UserCreationForm
3  from django import forms
4
5  from django.shortcuts import render
6
7  class SignUpView(CreateView):
8      form_class = UserCreationForm
9      template_name = 'registration/signup.html'
10
11     def get_success_url(self):
12         return reverse_lazy('login') + '?register'
13
14     def get_form(self, form_class=None):
15         form = super(SignUpView, self).get_form()
16         #Modificar en tiempo real
17         form.fields['username'].widget = forms.TextInput(attrs={'class':'form-control mb-2', 'placeholder':'Nombre de usuario'})
18         form.fields['password1'].widget = forms.PasswordInput(attrs={'class':'form-control mb-2', 'placeholder':'Contraseña'})
19         form.fields['password2'].widget = forms.PasswordInput(attrs={'class':'form-control mb-2', 'placeholder':'Repite la contraseña'})
20         return form

```

5. En el archivo base.html

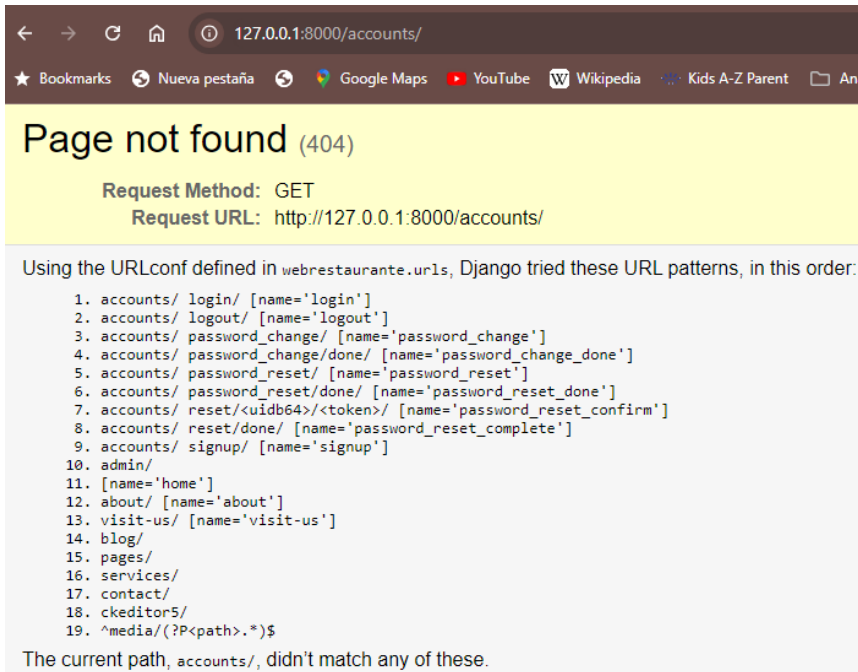
```

27     <style>
28         ul.errorlist {
29             color:red;
30         }
31         {% block contentStyle %} {%endblock%}
32     </style>
33 </head>

```

a.

6. Verificar /accounts ...



Page not found (404)

Request Method: GET  
Request URL: http://127.0.0.1:8000/accounts/

Using the URLconf defined in webrestaurante.urls, Django tried these URL patterns, in this order:

1. accounts/ login/ [name='login']
2. accounts/ logout/ [name='logout']
3. accounts/ password\_change/ [name='password\_change']
4. accounts/ password\_change/done/ [name='password\_change\_done']
5. accounts/ password\_reset/ [name='password\_reset']
6. accounts/ password\_reset/done/ [name='password\_reset\_done']
7. accounts/ reset/<uidb64>/<token>/ [name='password\_reset\_confirm']
8. accounts/ reset/done/ [name='password\_reset\_complete']
9. accounts/ signup/ [name='signup']
10. admin/
11. [name='home']
12. about/ [name='about']
13. visit-us/ [name='visit-us']
14. blog/
15. pages/
16. services/
17. contact/
18. ckeditor5/
19. ^media/(?P<path>.\*)\$

The current path, accounts/, didn't match any of these.

a.

7. Agregar los links correspondientes, en el archivo base.html

```

</li>
{% if not request.user.is_authenticated %}
<li class="nav-item">
| <a class="nav-link" href="{% url 'login' %}">Acceder</a>
</li>
<li class="nav-item">
| <a class="nav-link" href="{% url 'signup' %}">Registro</a>
</li>
{% else %}
<li class="nav-item">
| <span class="nav-link">{{ user.username }}</span>
</li>
<li class="nav-item">
| <form action="{% url 'logout' %}" method="post">
| | {% csrf_token %}
| | <button class="btn btn-outline-secondary btn-sm" type="submit">Salir</button>
| </form>
</li>
{% endif %}
</ul>

```

a.

## Extensión con Email

Ya que por default, la clase user cuenta con el campo de "email" es posible extender en el formulario dicho campo.

1. Crear el archivo forms.py y crear la clase **UserCreationFormWithEmail** que hereda de **UserCreationForm**, además se valida que el email no se encuentre ya asignado a algún usuario con el método `clean_email` (`clean_campo`)

```

forms.py x
DAW0822 > Practicas > solutions > django > webrestaurante > registration > forms.py
1  from django import forms
2  from django.contrib.auth.models import User
3  from django.contrib.auth.forms import UserCreationForm
4
5
6  class UserCreationFormWithEmail(UserCreationForm):
7      email = forms.EmailField(required=True, help_text="Requerido, 254 caracteres como máximo y debe ser válido")
8
9      class Meta:
10         model = User
11         fields = ("username", "email", "password1", "password2")
12
13         #Validación que el email sea válido clean_campo
14         def clean_email(self):
15             email = self.cleaned_data.get("email")
16             if User.objects.filter(email=email).exists():
17                 raise forms.ValidationError("El email ya está registrado")
18             return email
19
20

```

2. En el archivo views.py, asignar el campo `form_class` a **UserCreationFormWithEmail** y agregar en los campos el email.

```

1 from django.views.generic.edit import CreateView
2 from django.contrib.auth.forms import UserCreationForm
3 from django import forms
4 from django.urls import reverse_lazy
5 from django.shortcuts import render
6 from .forms import UserCreationFormWithEmail
7
8 class SignUpView(CreateView):
9     # form_class = UserCreationForm
10    form_class = UserCreationFormWithEmail
11    template_name = 'registration/signup.html'
12
13    def get_success_url(self):
14        return reverse_lazy('login') + '?register'
15
16    def get_form(self, form_class=None):
17        form = super(SignUpView, self).get_form()
18        #Modificar en tiempo real
19        form.fields['email'].widget = forms.EmailInput(attrs={'class':'form-control mb-2', 'placeholder':'Dirección email'})
20        form.fields['username'].widget = forms.TextInput(attrs={'class':'form-control mb-2', 'placeholder':'Nombre de usuario'})
21        form.fields['password1'].widget = forms.PasswordInput(attrs={'class':'form-control mb-2', 'placeholder':'Contraseña'})
22        form.fields['password2'].widget = forms.PasswordInput(attrs={'class':'form-control mb-2', 'placeholder':'Repite la contraseña'})
23        return form

```

## Restaurar contraseña

Para restaurar la contraseña, se puede configurar en modo de pruebas y se manejan en ficheros, haciendo una simulación del envío de correos)

Configuración de servidor de email de pruebas

En el archivo settings.py, se agrega el siguiente código

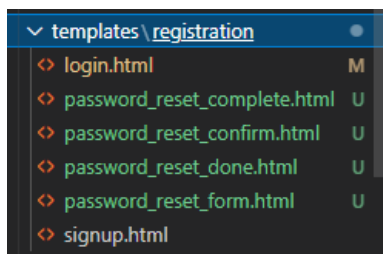
```

157 if DEBUG:
158     EMAIL_BACKEND = 'dango.core.mail.backends.filebased.EmailBackend'
159     EMAIL_FILE_PATH = os.path.join(BASE_DIR, "sent_emails")
160 else:
161     #Aquí va la configuración del email de producción
162     pass
163

```

Páginas relacionadas al password

Django ofrece ya toda la funcionalidad, por lo que solamente es necesario crear los htmls con los siguientes nombres



En el archivo password\_reset\_form.html

```

<form action="" method="post">
    {% csrf_token %}
    <p>Escribe tu correo electrónico</p>
    {% if form.email.errors %} {{ form.email.errors }} {% endif %}
    <p>{{ form.email }}</p>
    <input type="submit" class="btn btn-default btn-lg" value="Reset password" />
</form>

```

```

password_reset_form.html U X
DAW0822 > Practicas > solutions > django > webrestaurant > registration > templates > registration > password_reset_form.html
1  {% extends "core/base.html" %}
2  {% block content %}
3  {% load static %}
4  <div class="container">
5      <div class="about-heading-content mbtm">
6          <div class="row">
7              <div class="col-xl-9 col-lg-10 mx-auto">
8                  <div class="bg-faded rounded p-5 forced">
9                      <h2 class="section-heading mb-4">
10                         Cambio de password
11                     </h2>
12                     <div class="section-content">
13                         <form action="" method="post">
14                             {% csrf_token %}
15                             <p>Escribe tu correo electrónico</p>
16                             {% if form.email.errors %} {{ form.email.errors }} {% endif %}
17                             <p>{{ form.email }}</p>
18                             <input type="submit" class="btn btn-default btn-lg" value="Reset password" />
19                         </form>
20                     </div>
21                 </div>
22             </div>
23         </div>
24     </div>
25 </div>
26 {% endblock %}

```

En el archivo password\_reset\_confirm.html

```

{% if validlink %}
    <form action="" method="post">
        <div style="display:none">
            <input type="hidden" value="{{ csrf_token }}" name="csrfmiddlewaretoken">
        </div>
        <table>
            <tr>
                <td>{{ form.new_password1.errors }}
                    <label for="id_new_password1">Nuevo password:</label></td>
                <td>{{ form.new_password1 }}</td>
            </tr>
            <tr>
                <td>{{ form.new_password2.errors }}
                    <label for="id_new_password2">Confirma password:</label></td>
                <td>{{ form.new_password2 }}</td>
            </tr>
            <tr>
                <td></td>
                <td><input type="submit" value="Cambiar mi password" /></td>
            </tr>
        </table>
    </form>
{% else %}
    <h1>Password reset falló</h1>
    <p>El link es inválido, posiblemente porque ya ha sido usado. Por favor solicita nuevamente el cambio de password.</p>

```

```
{% endif %}
```

En el archivo password\_reset\_done.html

```
<h2 class="section-heading mb-4">
    Solicitud de cambio de password
</h2>
<div class="section-content">
    <p>Hemos enviado un correo para cambiar el password. </p>
</div>
```

En el archivo password\_reset\_complete.html

```
<h2 class="section-heading mb-4">
    Cambio de password
</h2>
<div class="section-content">
    <h1>El password ha sido cambiado</h1>
    <p><a href="{% url 'login' %}">¡Acceder nuevamente?</a></p>
</div>
```