### Restringir el acceso con @login\_required

Django facilita la restricción de acceso a determinadas páginas mediante el decorador @login\_required. Recuerda del Capítulo 11 que un *decorator* es una directiva que se coloca justo antes de la definición de una función y que modifica el comportamiento de ésta. Veamos un ejemplo.

#### Restringir el acceso a la página Temas

Cada tema será propiedad de un usuario, por lo que sólo los usuarios registrados podrán solicitar la página de temas. Añade el siguiente código a *learning\_logs/views.py*:

**learning\_logs/views.py**

from django.shortcuts import render, redirect  
from django.contrib.auth.decorators import login\_required  
  
from .models import Topic, Entry  
--snip--  
  
@login\_required  
def topics(request):  
 """Show all topics."""  
 --snip--

Primero importamos la función login\_required(). Aplicamos login\_required() como decorador a la función de vista topics() anteponiendo a login\_required el símbolo @. Como resultado, Python sabe que debe ejecutar el código de login\_required() antes que el de topics().

El código en login\_required() comprueba si un usuario está conectado, y Django ejecuta el código en topics() sólo si lo está. Si el usuario no ha iniciado sesión, se le redirige a la página de inicio de sesión.

Para que esta redirección funcione, tenemos que modificar *settings.py* para que Django sepa dónde encontrar la página de inicio de sesión. Añade lo siguiente al final de *settings.py*:

**settings.py**

--snip--  
# My settings.  
LOGIN\_REDIRECT\_URL = 'learning\_logs:index'  
LOGOUT\_REDIRECT\_URL = 'learning\_logs:index'  
LOGIN\_URL = 'accounts:login'

Ahora, cuando un usuario no autenticado solicite una página protegida por el decorador @login\_required, Django enviará al usuario a la URL definida por LOGIN\_URL en *settings.py*.

Puedes probar esta configuración cerrando la sesión de cualquier cuenta de usuario y accediendo a la página de inicio. Haz clic en el enlace **Temas**, que debería redirigirte a la página de inicio de sesión. A continuación, inicia sesión en cualquiera de tus cuentas y, desde la página de inicio, vuelve a hacer clic en el enlace **Temas**. Deberías poder acceder a la página de temas.

#### Restringir el acceso a través del Registro de Aprendizaje

Django facilita la restricción de acceso a las páginas, pero tienes que decidir qué páginas proteger. Lo mejor es pensar primero qué páginas deben estar libres de restricciones, y luego restringir todas las demás páginas del proyecto. Puedes corregir fácilmente el acceso excesivamente restringido, y es menos peligroso que dejar páginas sensibles sin restringir.

En Learning Log, mantendremos sin restricciones la página de inicio y la página de registro. Restringiremos el acceso a todas las demás páginas.

Aquí tienes *learning\_logs/views.py* con los decoradores @login\_required aplicados a todas las vistas excepto a index():

**learning\_logs/views.py**

--snip--  
@login\_required  
def topics(request):  
 --snip--  
  
@login\_required  
def topic(request, topic\_id):  
 --snip--  
  
@login\_required  
def new\_topic(request):  
 --snip--  
  
@login\_required  
def new\_entry(request, topic\_id):  
 --snip--  
  
@login\_required  
def edit\_entry(request, entry\_id):  
 --snip--

Intenta acceder a cada una de estas páginas mientras estás desconectado; deberías ser redirigido a la página de inicio de sesión. Tampoco podrás hacer clic en enlaces a páginas como new\_topic. Pero si introduces la URL *http://localhost:8000/new\_topic/*, serás redirigido a la página de inicio de sesión. Deberías restringir el acceso a cualquier URL que sea de acceso público y esté relacionada con datos privados del usuario.

[anterior](c19_14.html)[Subtema 15 de 23: (Ver todo)](c19.html)[siguiente](c19_16.html)