### Conexión de datos a determinados usuarios

A continuación, tenemos que conectar los datos al usuario que los envió. Sólo tenemos que conectar los datos más altos de la jerarquía a un usuario, y los datos de nivel inferior le seguirán. En Learning Log, los temas son el nivel más alto de datos de la aplicación, y todas las entradas están conectadas a un tema. Mientras cada tema pertenezca a un usuario concreto, podremos rastrear la propiedad de cada entrada en la base de datos.

Modificaremos el modelo Topic añadiendo una relación de clave foránea a un usuario. A continuación, tendremos que migrar la base de datos. Por último, modificaremos algunas de las vistas para que sólo muestren los datos asociados al usuario conectado en ese momento.

#### Modificar el modelo de temas

La modificación de *models.py* son sólo dos líneas:

**models.py**

from django.db import models  
from django.contrib.auth.models import User  
  
class Topic(models.Model):  
 """A topic the user is learning about."""  
 Text = models.CharField(max\_length=200)  
 date\_added = models.DateTimeField(auto\_now\_add=True)  
 owner = models.ForeignKey(User, on\_delete=models.CASCADE)  
  
 def \_\_str\_\_(self):  
 """Return a string representing the topic."""  
 Return self.text  
  
class Entry(models.Model):  
 --snip--

Importamos el modelo User de django.contrib.auth. A continuación, añadimos un campo owner a Topic, que establece una relación de clave foránea con el modelo User. Si se elimina un usuario, se eliminarán también todos los temas asociados a ese usuario.

#### Identificar a los usuarios existentes

Cuando migremos la base de datos, Django modificará la base de datos para que pueda almacenar una conexión entre cada tema y un usuario. Para realizar la migración, Django necesita saber qué usuario asociar a cada tema existente. Lo más sencillo es empezar asignando todos los temas existentes a un usuario, por ejemplo, el superusuario. Pero antes, necesitamos conocer el ID de ese usuario.

Veamos los ID de todos los usuarios creados hasta ahora. Inicia una sesión shell de Django y emite los siguientes comandos:

(ll\_env)learning\_log$ python manage.py shell  
❶ >>> from django.contrib.auth.models import User  
❷ >>> User.objects.all()  
<QuerySet [<User: ll\_admin>, <User: eric>, <User: willie>]>  
❸ >>> for user in User.objects.all():  
... print(user.username, user.id)  
...  
ll\_admin 1  
eric 2  
willie 3  
>>>

Primero importamos el modelo User a la sesión shell ❶. A continuación, miramos todos los usuarios que se han creado hasta el momento ❷. La salida muestra tres usuarios para mi versión del proyecto: ll\_admin, eric, y willie.

A continuación, recorremos la lista de usuarios e imprimimos el nombre de usuario y el ID ❸ de cada usuario . Cuando Django pregunte a qué usuario asociar los temas existentes, utilizaremos uno de estos valores de ID.

#### Migrar la base de datos

Ahora que conocemos los ID, podemos migrar la base de datos. Cuando lo hagamos, Python nos pedirá que conectemos el modelo Topic a un propietario concreto temporalmente o que añadamos un valor por defecto a nuestro archivo *models.py* para indicarle qué hacer. Elige la opción **1**:

❶ (ll\_env)learning\_log$ python manage.py makemigrations learning\_logs  
❷ It is impossible to add a non-nullable field 'owner' to topic without  
specifying a default. This is because...  
❸ Please select a fix:  
 1) Provide a one-off default now (will be set on all existing rows with a  
 null value for this column)  
 2) Quit and manually define a default value in models.py.  
❹ Select an option: 1  
❺ Please enter the default value now, as valid Python  
The datetime and django.utils.timezone modules are available...  
Type 'exit' to exit this prompt  
❻ >>> 1  
Migrations for 'learning\_logs':  
 learning\_logs/migrations/0003\_topic\_owner.py  
- Add field owner to topic  
(ll\_env)learning\_log$

Comenzamos emitiendo el comando makemigrations ❶. En la salida, Django indica que estamos intentando añadir un campo obligatorio (*non-nullable*) a un modelo existente (topic) sin valor por defecto especificado ❷. Django nos da dos opciones: podemos proporcionar un valor por defecto ahora mismo, o podemos abandonar y añadir un valor por defecto en *models.py* ❸. Aquí he elegido la primera opción ❹. A continuación, Django nos pide que introduzcamos el valor por defecto ❺.

Para asociar todos los temas existentes al usuario administrador original, ll\_admin, he introducido el ID de usuario de 1 ❻. Puedes utilizar el ID de cualquier usuario que hayas creado; no tiene por qué ser un superusuario. A continuación, Django migra la base de datos utilizando este valor y genera el archivo de migración *0003\_topic\_owner.py*, que añade el campo owner al modelo Topic.

Ahora podemos ejecutar la migración. Introduce lo siguiente en un entorno virtual activo:

(ll\_env)learning\_log$ python manage.py migrate  
Operations to perform:  
 Apply all migrations: admin, auth, contenttypes, learning\_logs, sessions  
Running migrations:  
❶ Applying learning\_logs.0003\_topic\_owner... OK  
(ll\_env)learning\_log$

Django aplica la nueva migración, y el resultado es OK ❶.

Podemos verificar que la migración ha funcionado como se esperaba en una sesión de shell, de la siguiente forma:

>>> from learning\_logs.models import Topic  
>>> for topic in Topic.objects.all():  
... print(topic, topic.owner)  
...  
Chess ll\_admin  
Rock Climbing ll\_admin  
>>>

Importamos Topic de learning\_logs.models y luego recorremos en bucle todos los temas existentes, imprimiendo cada tema y el usuario al que pertenece. Puedes ver que cada tema pertenece ahora al usuario ll\_admin. (Si obtienes un error al ejecutar este código, intenta salir de la shell e iniciar una nueva shell).

[anterior](c19_15.html)[Subtema 16 de 23: (Ver todo)](c19.html)[siguiente](c19_17.html)