


Open Opened 3 days ago by  **Rubén Montero**

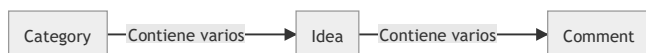
Modelos y sitio de administración

Resumen

- Crearemos clases en `models.py` para representar *Categorías*, *Ideas* y *Comentarios*
- Hablaremos del sitio de administración Django
- Permitiremos editar las *Categorías* de la base de datos con un superusuario Django

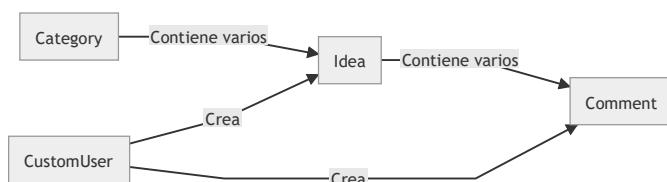
Descripción

Anteriormente presentamos cuáles serían nuestros modelos en lo relevante a las aportaciones de los usuarios:



- Las **Category** son prefijadas, gestionadas por el administrador
- Los usuarios crean **Idea**, que pertenecen a una **Category**
- Los usuarios crean **Comment**, que pertenecen a una **Idea**

Por lo tanto, en realidad:



models.py

Añadamos los modelos restantes:

```

class Category(models.Model):
    title = models.CharField(max_length=60, unique=True)

class Idea(models.Model):
    title = models.CharField(max_length=200)
    description = models.CharField(max_length=2000)
    user = models.ForeignKey(CustomUser, on_delete=models.CASCADE)
    category = models.ForeignKey(Category, on_delete=models.CASCADE)

class Comment(models.Model):
    content = models.CharField(max_length=1400)
    user = models.ForeignKey(CustomUser, on_delete=models.CASCADE)
    idea = models.ForeignKey(Idea, on_delete=models.CASCADE)
  
```

Migraciones

...y migremos la base de datos:

```

python manage.py makemigrations
python manage.py migrate
  
```

Ahora, hemos dicho varias veces que:

Las **Category** son prefijadas, gestionadas por el administrador

Nosotros **sabemos** manipular la base de datos mediante la línea de comandos `sqlite3.exe`.

¡Django nos ofrece algo mucho más estimulante!

Panel de administración

¿Recuerdas esta enigmática línea que se crea en `urls.py` por defecto?

```
path('admin/', admin.site.urls),
```

Pues tiene **su significado**.

Si visitas <http://localhost:8000/admin>, verás el *panel de administración* de Django:



¿Y cómo hago login?

Necesitamos **crear un usuario administrador**.

¿Cómo?

Desde la carpeta `IdeAPI/` donde se aloja `manage.py`, lanzamos el comando:

```
python manage.py createsuperuser
```

Nos pedirá:

- Nombre de usuario
- Email
- Contraseña

¡...y añadirá al usuario a la base de datos!

Ya podemos introducir dichas credenciales en <http://localhost:8000/admin>

Me he *logueado* en el panel

Excelente.

Verás una sección que te permite gestionar *Groups* y *Users*. Estos son modelos predeterminados de Django. Pero aún no podemos gestionar nuestros **propios modelos**.

`admin.py`

Abre este archivo en `idearest04app/`. Habrá algo como:

```
from django.contrib import admin

# Register your models here.
```

Registrar un modelo para su administración

Sólo necesitamos lo siguiente:

```
from django.contrib import admin

from .models import Category

admin.site.register(Category)
```

Refresca la página <http://localhost:8000/admin> en tu navegador

Vaya hay un error...

```
OperationalError at /admin/idearest04app/category/

no such table: idearest04app_category
```

Si ves esto, te olvidaste de hacer `python manage.py makemigrations` y `python manage.py migrate`. Tras hacerlo...

¡Funciona! Ya lo veo

¡Excelente!

Prueba a **añadir** estas cuatro `Category` :

- **Title:** Sociedad
- **Title:** Educación
- **Title:** Arte
- **Title:** Invenciones

Estás **añadiendo filas** a la base de datos, sin necesidad de `sqlite3.exe`. ¿No es genial?

Pero en la pantalla de resumen...

...se ve:

```
Category object (4)
Category object (3)
Category object (2)
Category object (1)
```

¡No te preocupes! Simplemente, Django **no** sabe cómo *representar* nuestras `Category` .

Pero podemos conseguirlo *sobreescribiendo* el método `__str__` en la clase, que es análogo al `toString()` de Java.

En `models.py` :

```
class Category(models.Model):
    title = models.CharField(max_length=60, unique=True)
    +
    + def __str__(self):
    +     return self.title
```

¿Qué tal se ve ahora <http://localhost:8000/admin/idearest04app/category/>?

¡Excelente!

Pues, **¡enhorabuena!** Has visitado y trabajado por primera vez en tu panel de administración Django.

¡Tarea terminada!

Por último

Verifica que tu código pasa el *test* asociado a la tarea.

Haz `commit` y `push` para subir los cambios al repositorio.



[Rubén Montero @ruben.montero](#) changed milestone to [%Sprint 5](#) 3 days ago