


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

# Una tabla de usuarios

## Resumen

- Crearemos una clase en `model.py` que represente los usuarios
- Efectuaremos las migraciones a la base de datos

## Descripción

Como bien sabemos, un *modelo* es una clase que, según el [mapeo objeto-relacional](#) representa una tabla SQL de una base de datos.

Crearemos un nuevo *modelo* que represente a los *usuarios* de nuestro `IdeAPI`.

### CustomUser

CustomUser

### ¿Y por qué no `User`?

Entraría en conflicto con el `User` que ya crea Django por defecto desde la app `django.contrib.admin`. Nosotros crearemos un modelo nuevo **desde cero**. ¡Es la mejor forma de aprender!

## Vamos allá

Si abres `IdeAPI/idearest04app/models.py` verás:

```
from django.db import models

# Create your models here.
```

Nuestra clase `CustomUser` **debe** indicar `models.Model` entre los *paréntesis* `( ( ) )`, que es la sintaxis de Python para indicar [herencia](#).

```
from django.db import models

class CustomUser(models.Model):
    # ...
```

## Tres `VARCHAR`

Especificamos *tres* atributos de tipo `models.CharField`, que es la *clase* Django correspondiente a un `VARCHAR`:

```
from django.db import models

class CustomUser(models.Model):
    email = models.CharField(max_length=280, unique=True)
    username = models.CharField(max_length=300)
    encrypted_password = models.CharField(max_length=120)
```

### ¿`email unique=True`?

¡Buena observación! Sirve para que ese atributo deba ser **único** para cada fila. En otras palabras, no puede haber dos **filas** de la tabla con el mismo `email`. Así evitaremos que se registren dos usuarios con el mismo correo electrónico.

### `username` no parece ser `unique=True`

En efecto. Permitiremos que dos usuarios distintos tengan el mismo `username`.

### ¿Por qué?

Pues... Porque podemos. Es nuestra decisión de diseño.

## ¿Y conlleva alguna limitación?

Sí.

Los usuarios **no** podrán *loguearse* mediante su nombre de usuario, ya que no es único. Deberán hacerlo mediante su **correo electrónico**.

Prosigamos.

## Un momento, ¿ **encrypted\_password** ?

Sí. La **contraseña** del usuario debemos almacenarla en la base de datos... **¡Encriptada!**

En caso de que haya algún *acceso no autorizado* a la base de datos, **no** queremos exponer *contraseñas en texto claro*. Profundizaremos en esto en la tarea siguiente.

## Vale. Prosigamos

**Migra** tu *modelo* a una base de datos real **ejecutando** estos dos comandos desde la carpeta **IdeAPI/** que contiene **manage.py** :

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

¡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