DESARROLLO CON PYTHON

Contenido de formación


Programación


Pyhton


Paradigma POO


Bases de datos SQL


Tkinter

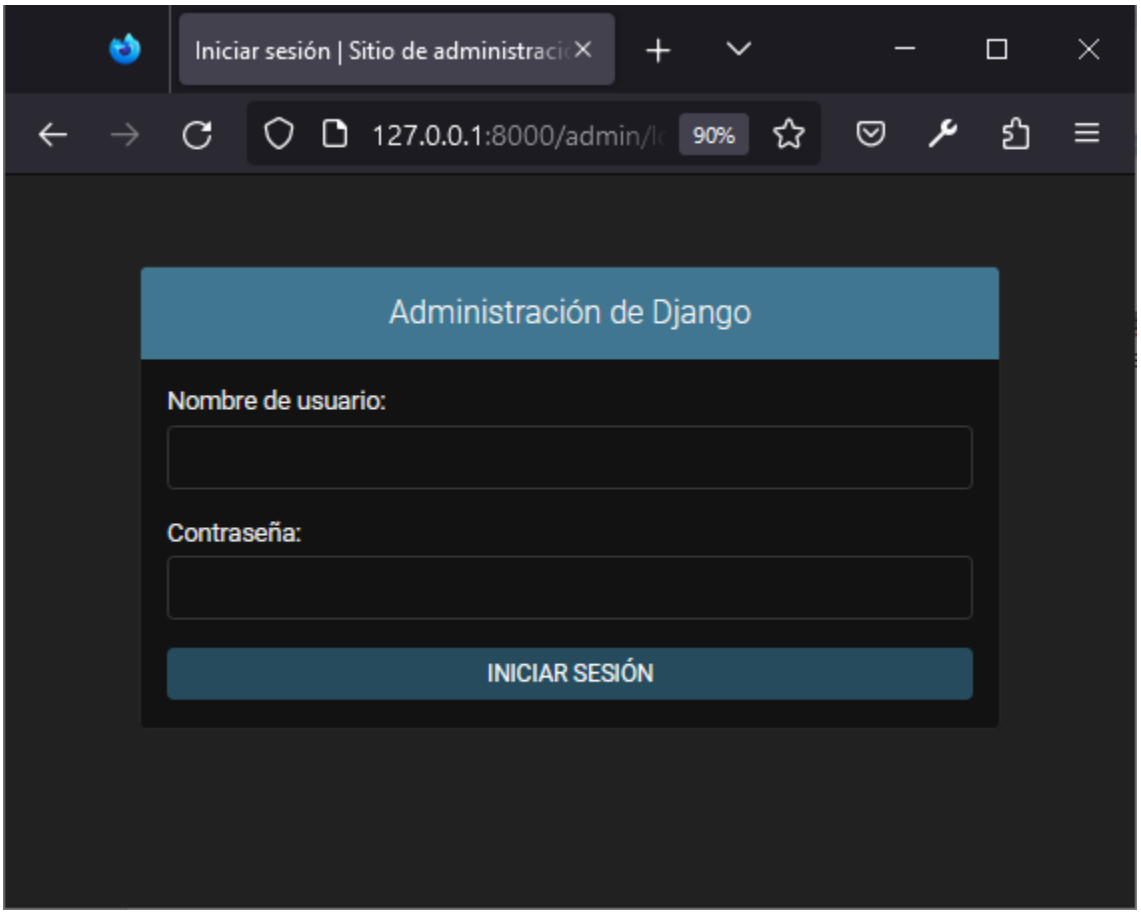

Django


Flask

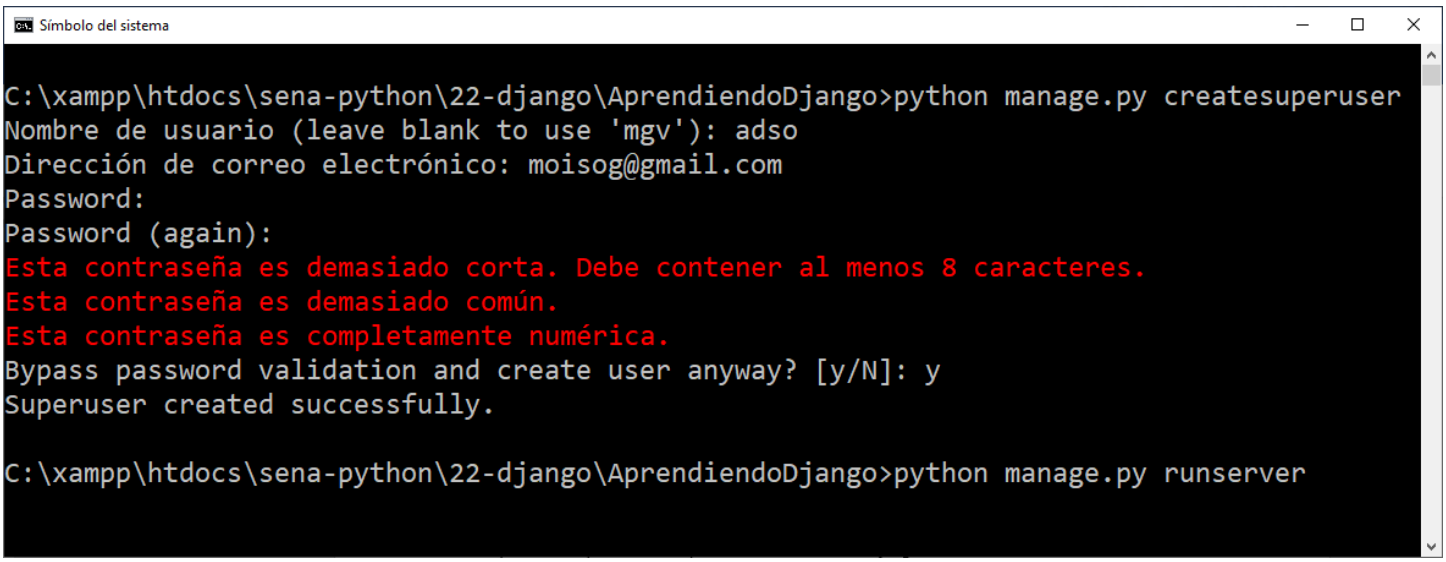
1. Primeros Instalación de SW
2. Variables y tipos de datos
3. Operadores (aritméticos, asignación)
4. Entrada y salida de datos
5. Estructuras de control
   5.1 Condicionales (operadores lógicos, operadores de comparación)
   5.2 Bucles (estructuras interactivas for, while)
6. Bloque de ejercicios de aplicación de conceptos
7. Funciones (parámetros, return, invocación, lambda)
   7.1 Variables locales y globales, funciones y métodos predefinidos
8. Lista y tuplas (creación, índices, recorrer y mostrar listas, listas multidimensionales)
9. Diccionarios y sets
10. Bloque de ejercicios aplicación de conceptos
11. Módulos y paquetes (creación y funcionalidad)
12. Sistemas de archivos y directorios
13. Manejo de errores (captura de excepciones, errores personalizados)
14. Programación orientada a objetos
15. Bases de datos SQLite
16. Bases de datos MySQL
17. Proyecto con Python
18. Interfaces graficas con Tkinter (aplicación de escritorio Python – Tkinter)
19. Desarrollo Web con Django
20. Interfaces graficas con Flask (aplicación de escritorio Python – Flask)

Panel de administrador de Django

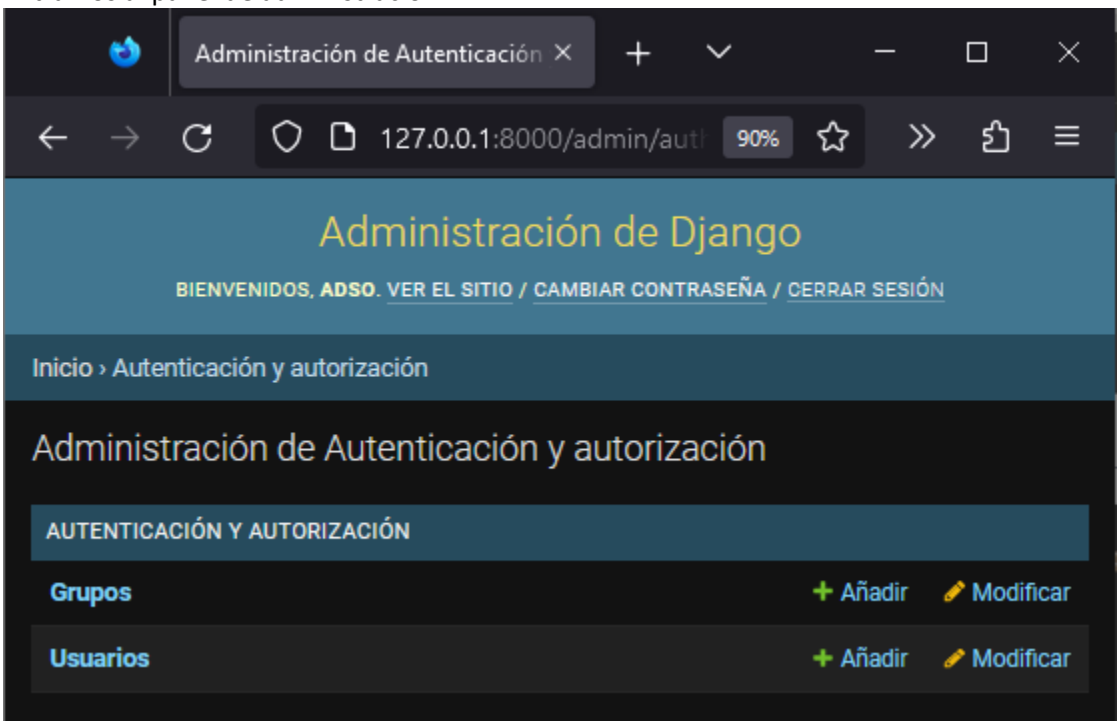
Una de las mejores características de django es que cuenta con el django admin panel, un panel de administración listo para usarse, con funciones básicas como crear, leer, editar y eliminar modelos, usuarios, grupos y permisos. Todo listo con solo montar tu aplicación.



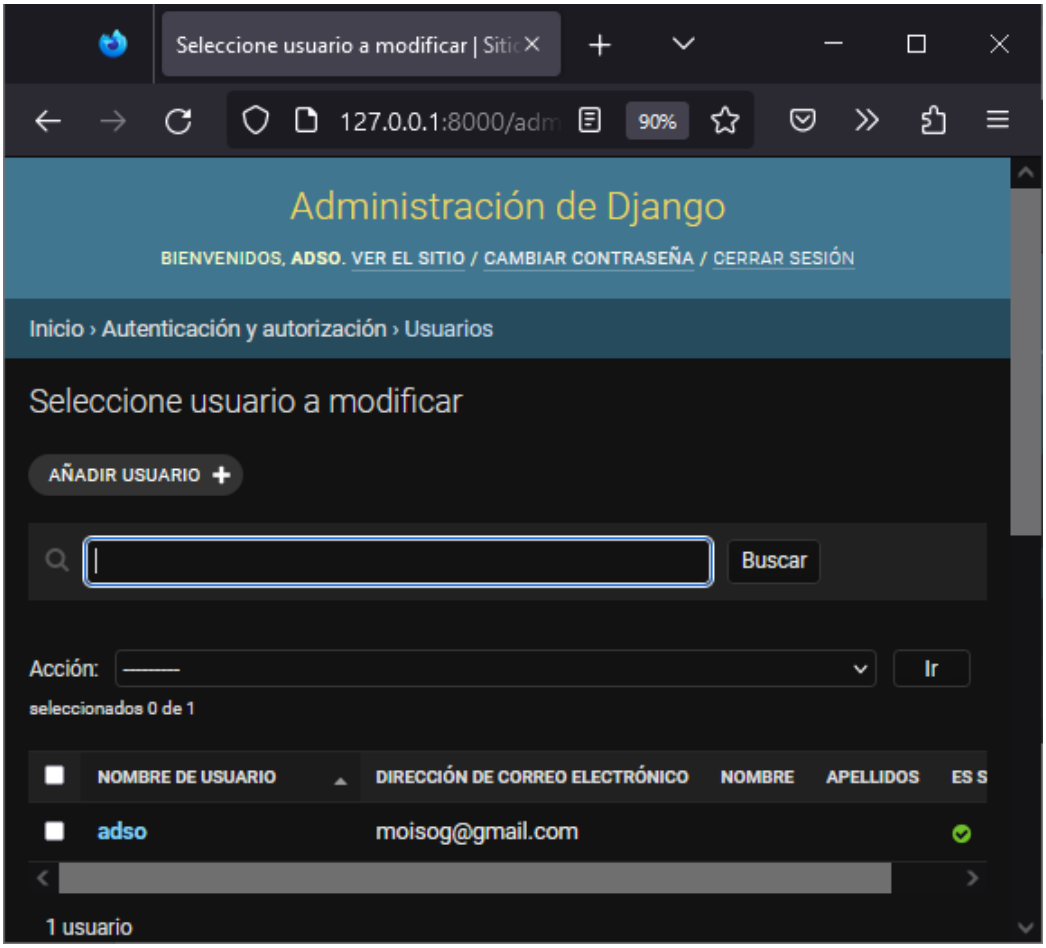
Creación de super usuario para entrar al panel de administración Django



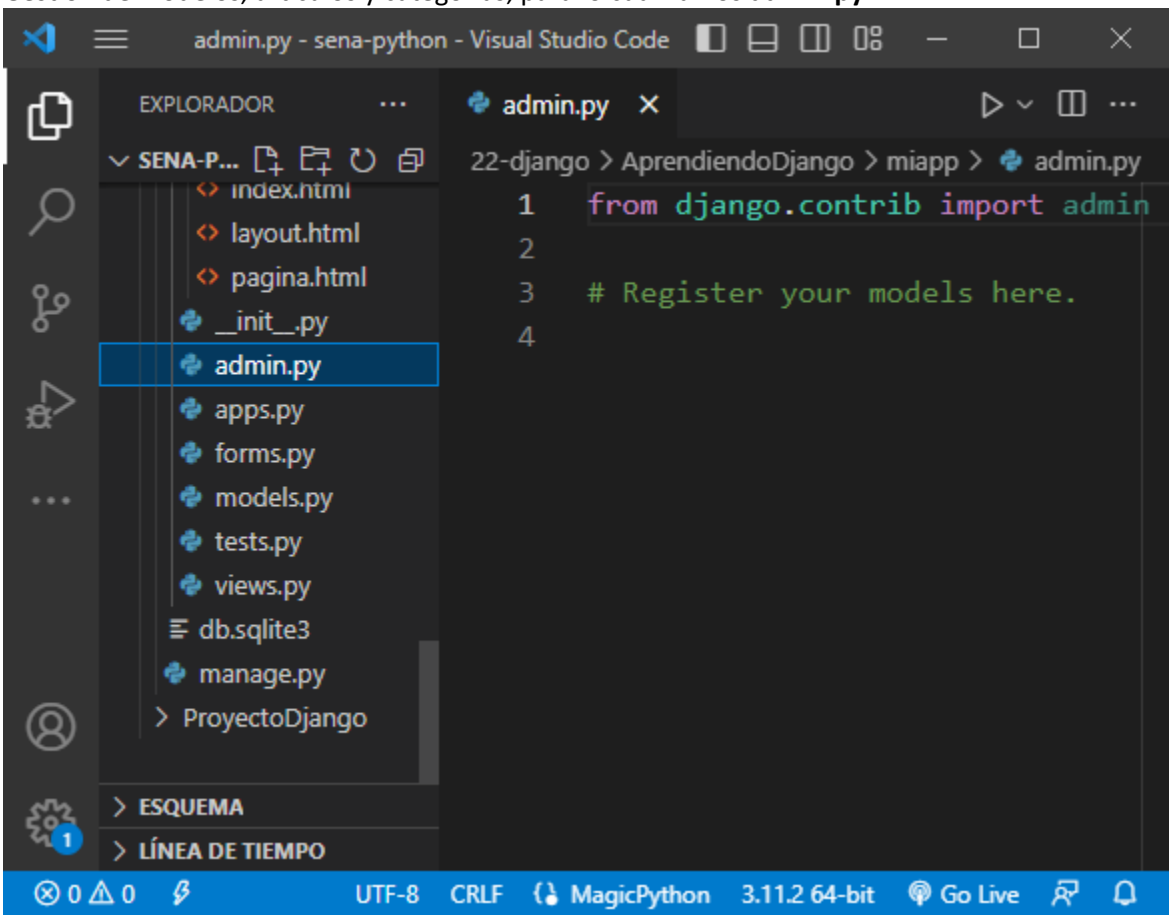
Entramos al panel de administración



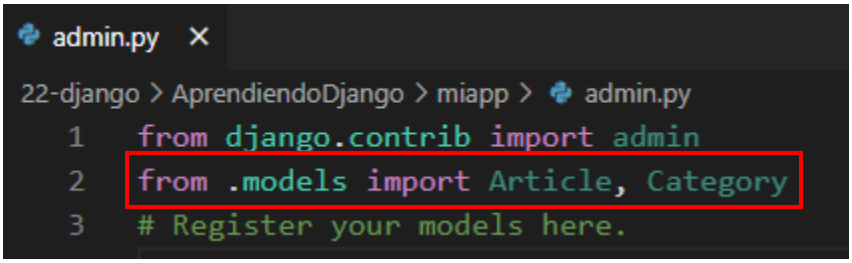
Podemos ver el usuario.. entre otras caracteriscas



Gestión de modelos, artículos y categorías, para lo cual vamos **admin.py**

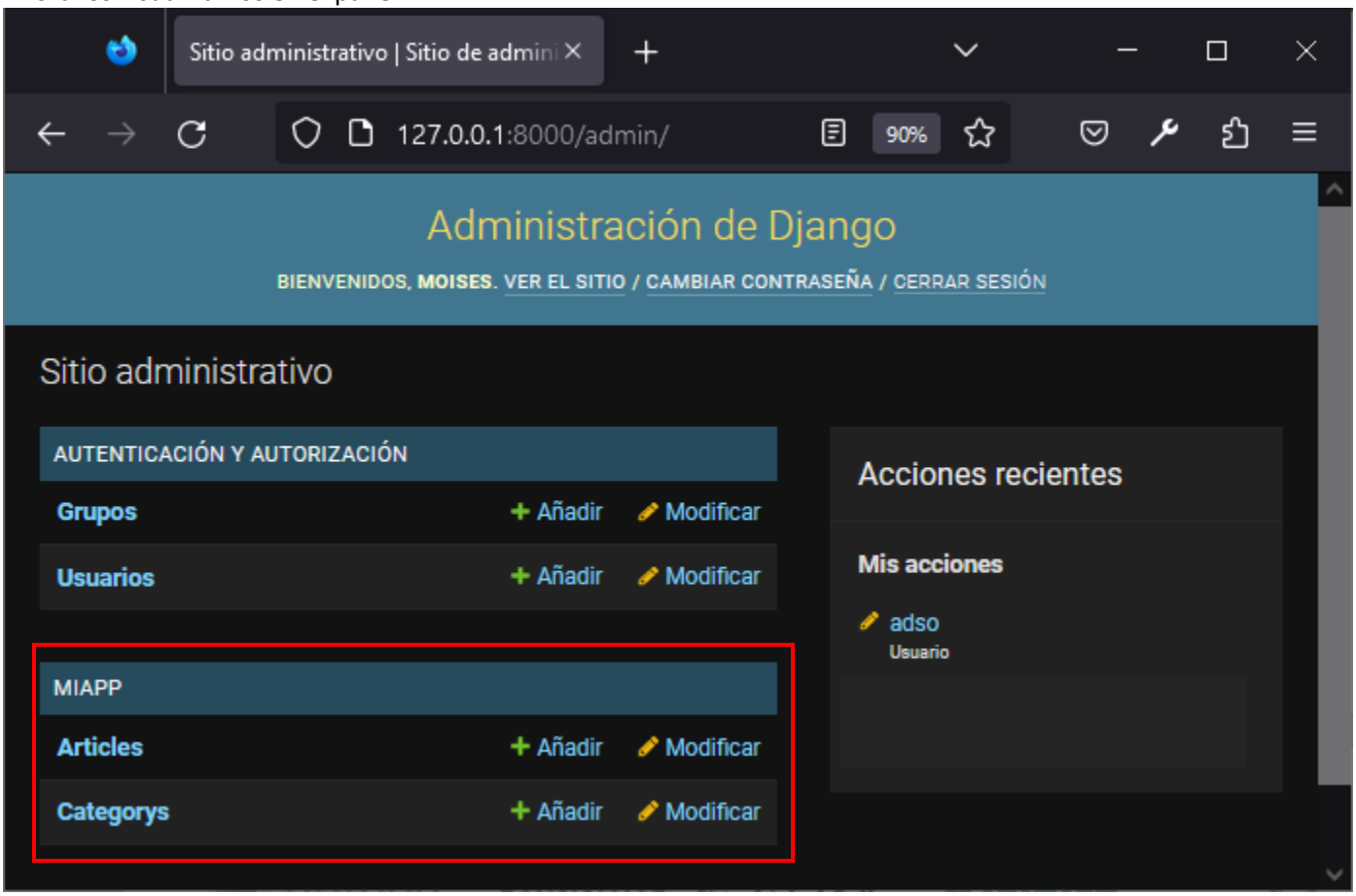


Ahora importamos los modelos from .models import Article, Category
Puedo hacerlo también utilizando el * para decirle que importe todos los modelos from .models import *



Ahora los registramos

Ahora los visualizamos en el panel



Clase Meta en los modelos Django, configurar el panel para que los modelos aparezcan en español

La clase Meta es una clase interna, lo que significa que se define dentro del modelo.

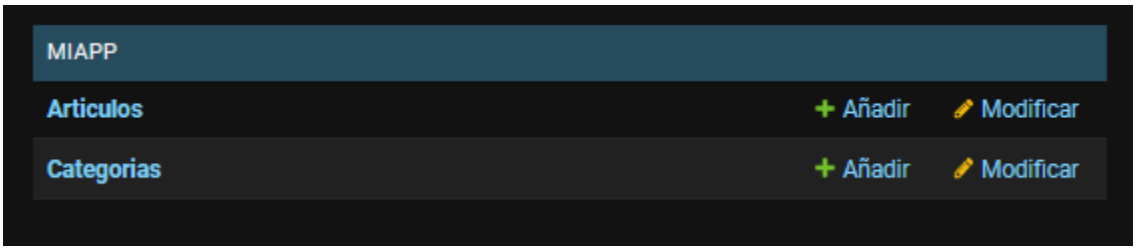
La clase Meta se puede utilizar para definir varias cosas sobre el modelo, como los permisos, el nombre de la base de datos, los nombres en singular y plural, abstracción, ordenación, etc. Agregar clases Meta a los modelos Django es completamente opcional.

Esta clase también viene con muchas opciones que puede configurar. Las siguientes son algunas de las meta-opciones de uso común; puedes explorar todas las opciones meta en:
https://docs.djangoproject.com/en/3.0/ref/models/options/

models.py

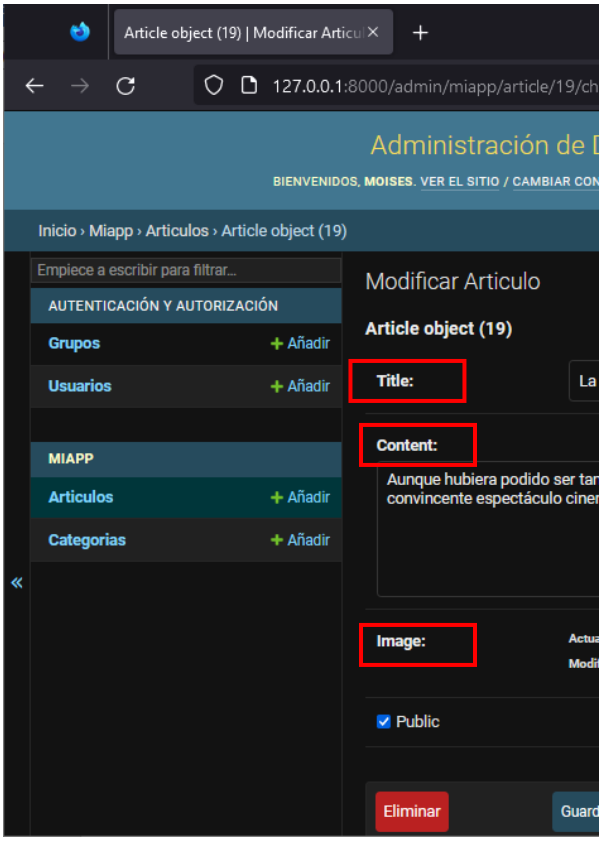
22-django > AprendiendoDjango > miapp > models.py > Category > Meta

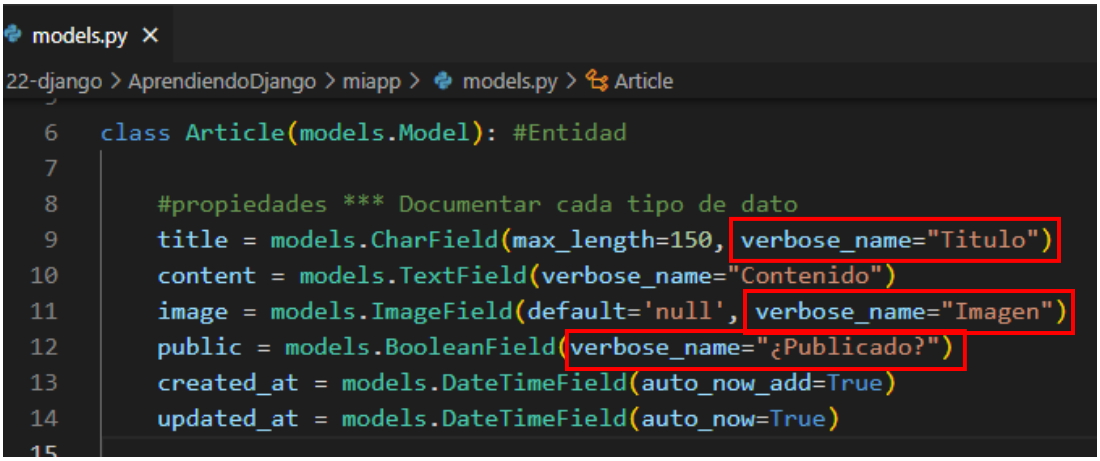
```python
class Article(models.Model): #Entidad

    #propiedades *** Documentar cada tipo de dato
    title = models.CharField(max_length=150)
    content = models.TextField()
    image = models.ImageField(default='null')
    public = models.BooleanField()
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)

    class Meta:
        #poner nombre en singular
        verbose_name = "Articulo"
        verbose_name_plural = "Articulos"
class Category(models.Model): #Entidad
    name = models.CharField(max_length=110)
    description = models.CharField(max_length=250)
    created_at = models.DateField()

    class Meta:
        #poner nombre en singular
        verbose_name = "Categoria"
        verbose_name_plural = "Categorias"
```



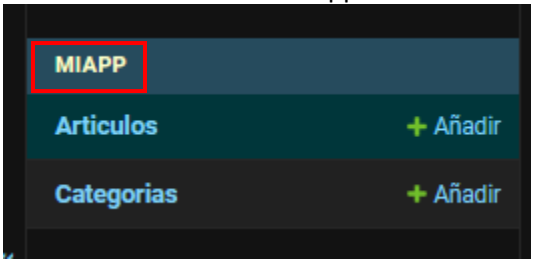
Manipular propiedades de los modelos

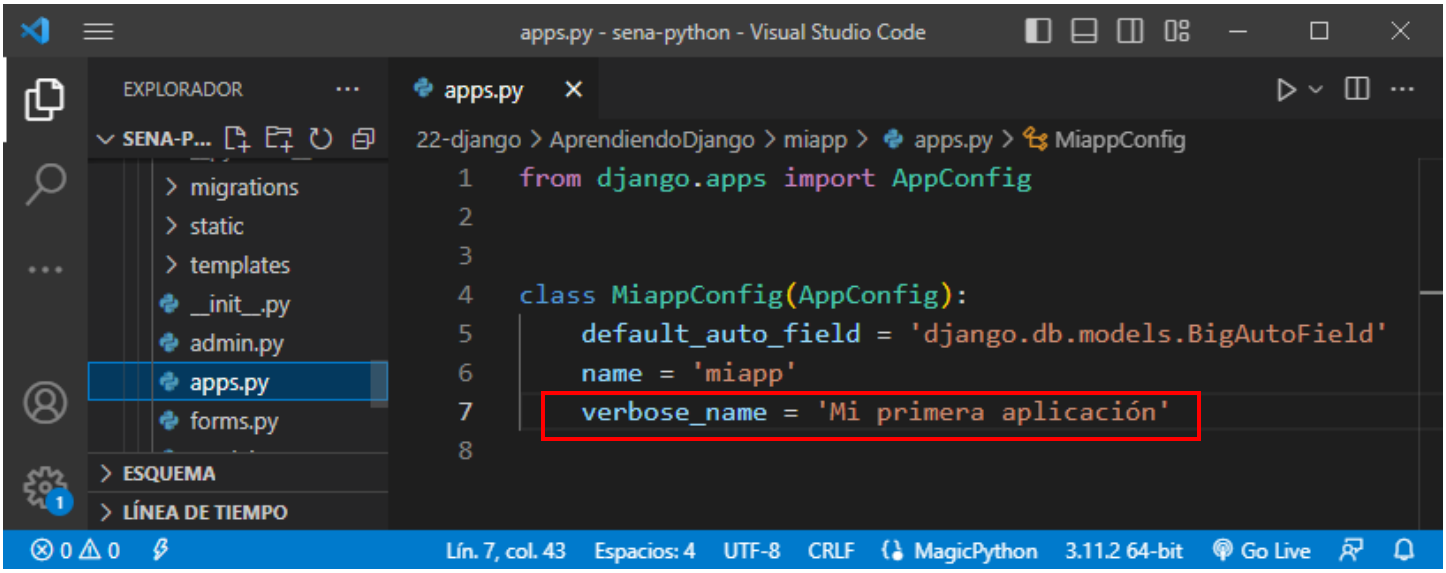
Traducir los parámetros de los campos a español

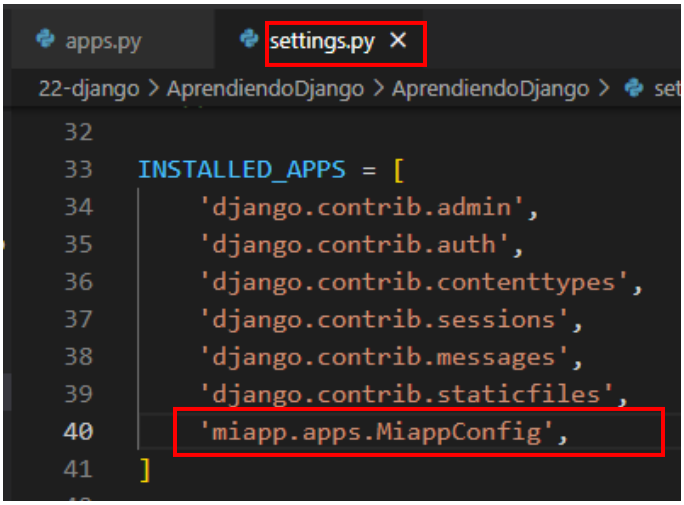


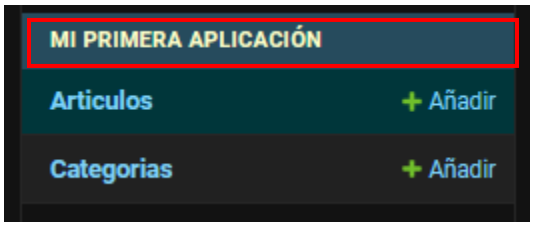


Cambiar el nombre de las apps

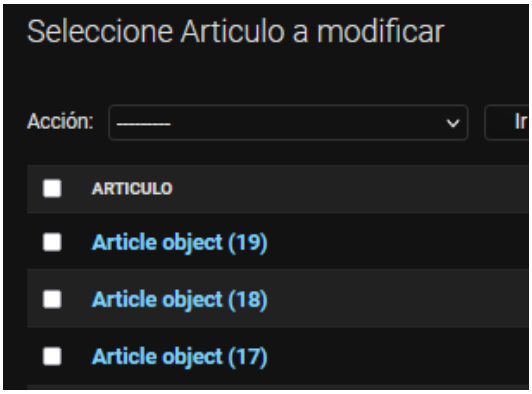


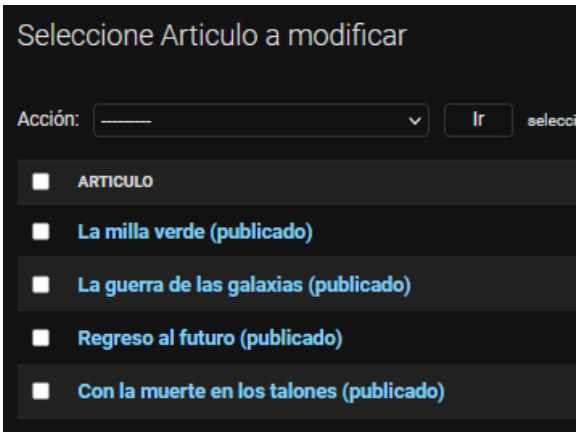


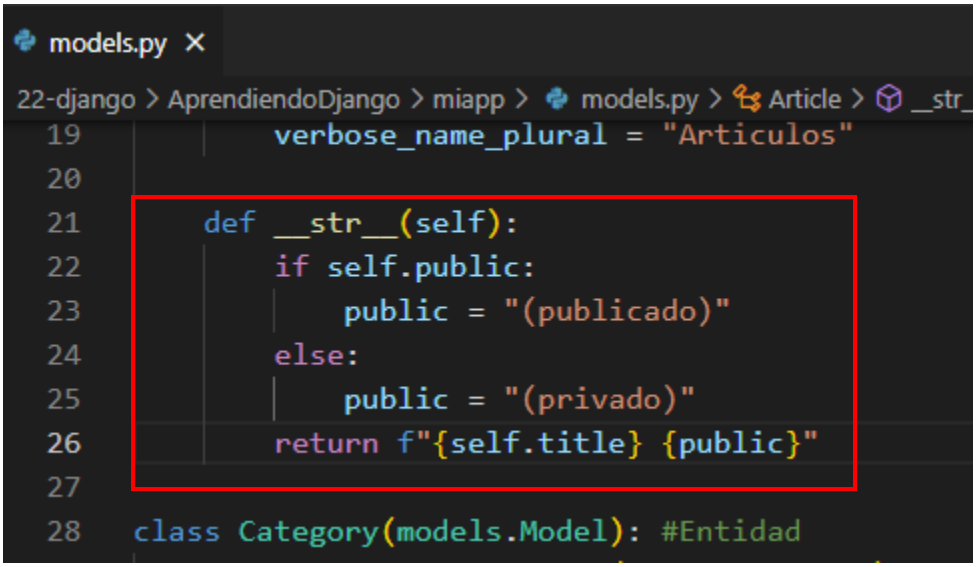




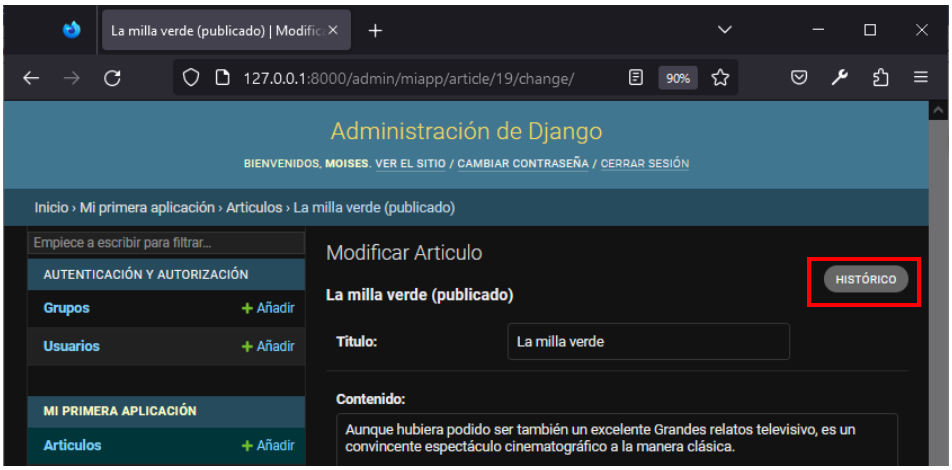
Método mágico para imprimir objetos (modelos) así esta vamos a cambiarlo:
para acceder a mas métodos mágicos https://rszalski.github.io/magicmethods/

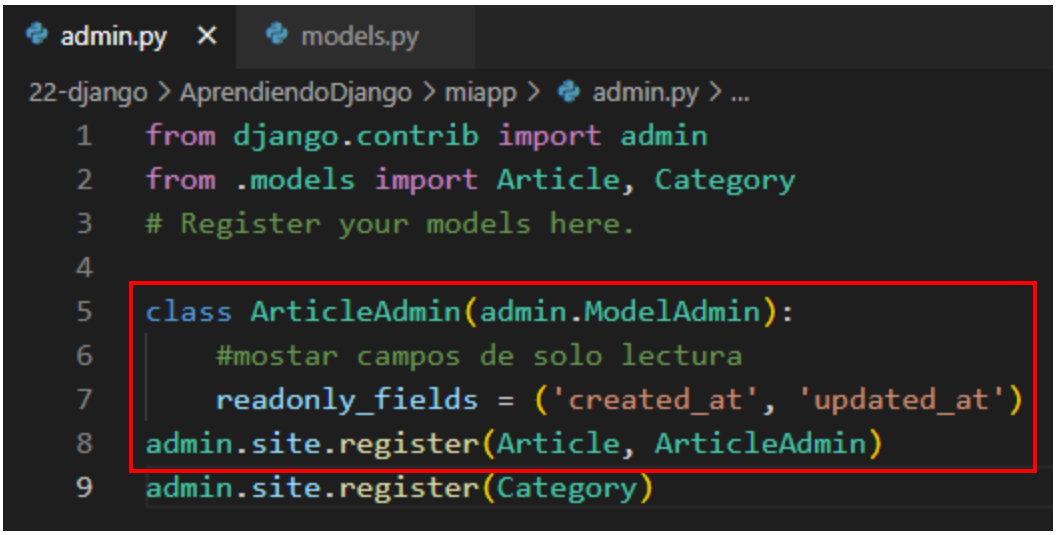


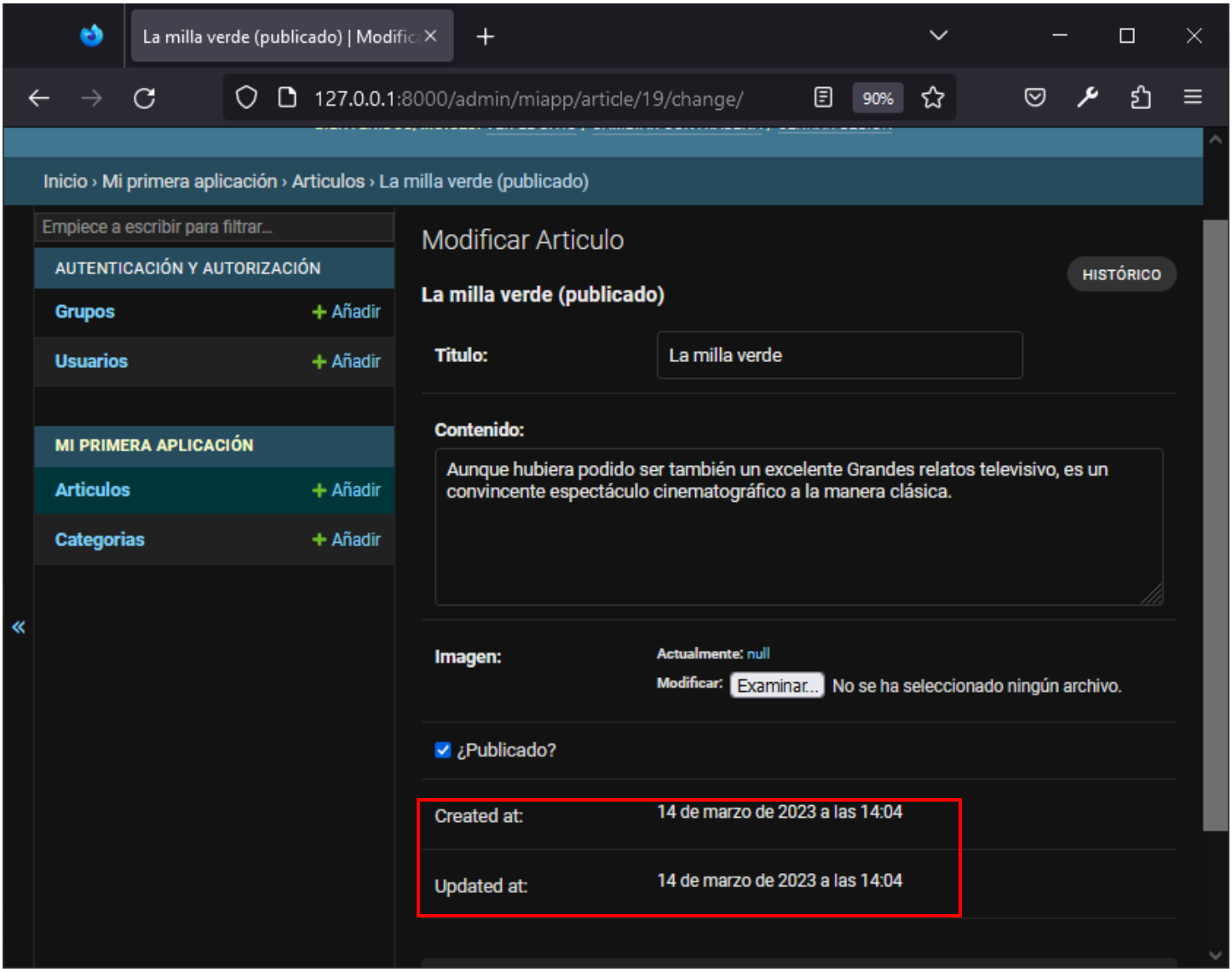




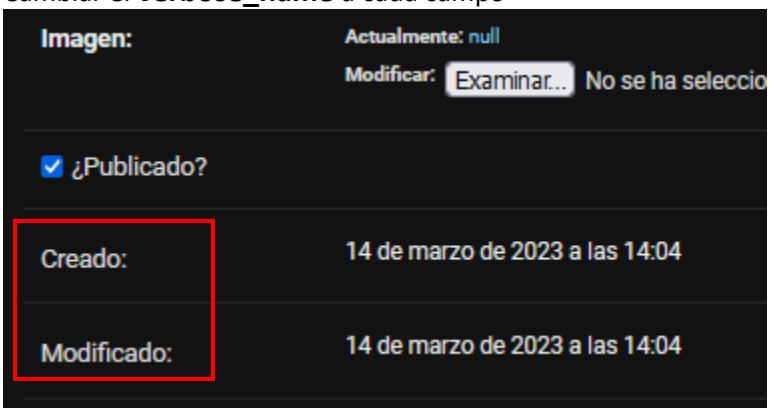
Mostar campos de solo lectura:





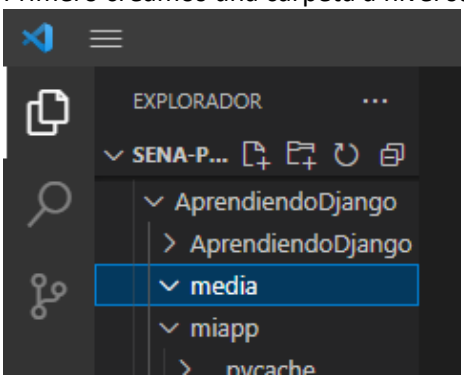


Cambiar el **verbose_name** a cada campo

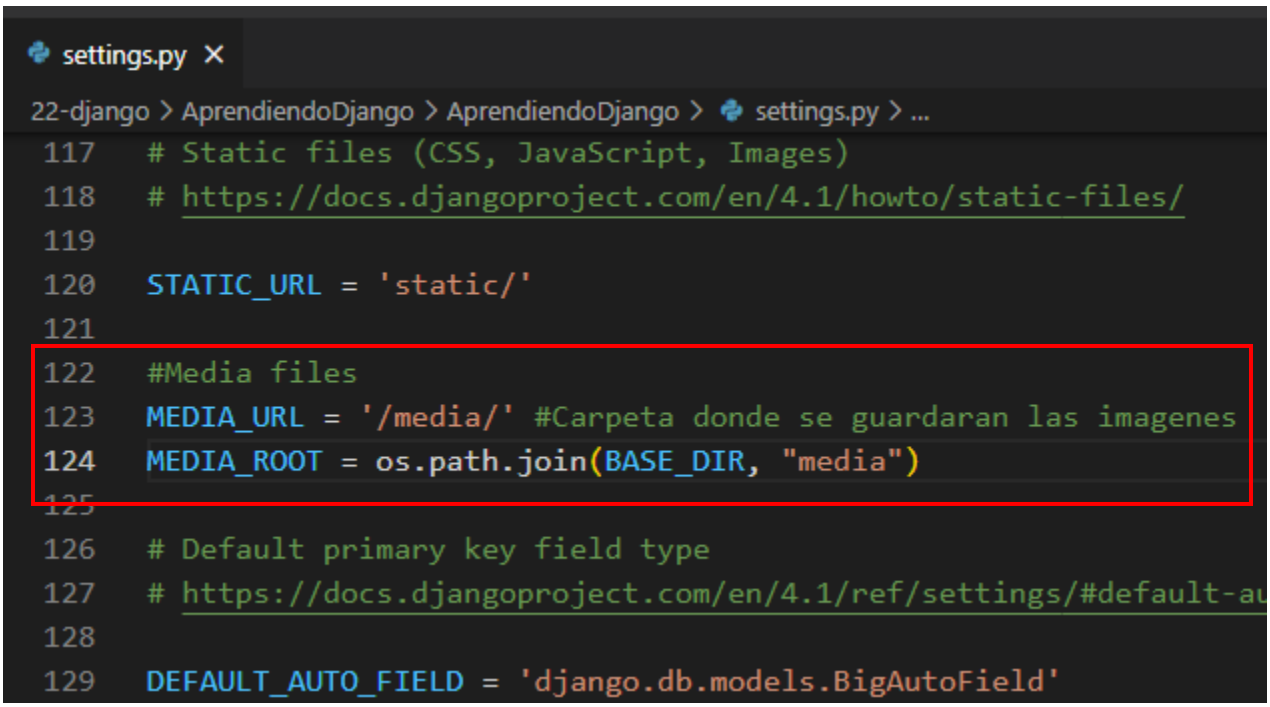


Configurar la aplicación para subir imágenes

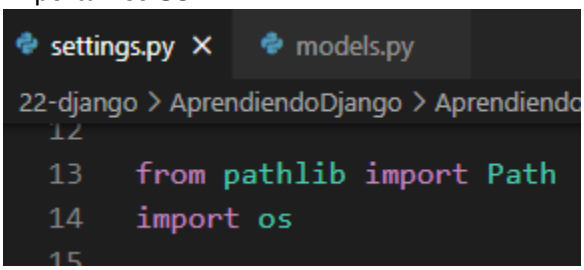
Primero creamos una carpeta a nivel superior de A**prendiendoDjango** llamada **media**



Ahora abrimos la configurara que se encuentra en la carpeta A**prendiendoDjango** en **settings.py**



Importamos **OS**



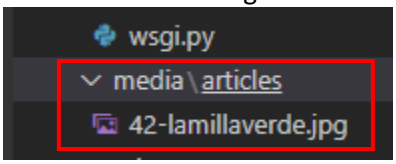
Ahora en **models.py** indicamos una carpeta donde se guardarán las imágenes de cada articulo

```python
#propiedades *** Documentar cada tipo de dato
title = models.CharField(max_length=150, verbose_name="Titulo")
content = models.TextField(verbose_name="Contenido")
image = models.ImageField(default='null', verbose_name="Imagen", upload_to="articles")
public = models.BooleanField(verbose_name="¿Publicado?")
created_at = models.DateTimeField(auto_now_add=True, verbose_name= "Creado")
updated_at = models.DateTimeField(auto_now=True, verbose_name= "Modificado")

class Meta:
```

Ahora abrimos la consola cortamos la ejecución del servidor y instalamos **Pillow** para manejo de imágenes

```
Símbolo del sistema
Performing system checks...

System check identified no issues (0 silenced).
March 14, 2023 - 11:29:40
Django version 4.1.6, using settings 'AprendiendoDjango.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

C:\xampp\htdocs\sena-python\22-django\AprendiendoDjango>pip install Pillow
Requirement already satisfied: Pillow in c:\python311\lib\site-packages (9.4.0)

[notice] A new release of pip available: 22.3.1 -> 23.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Guardamos una imagen en el artículo cualquiera y comprobamos que se guardó la imagen en **media → article**



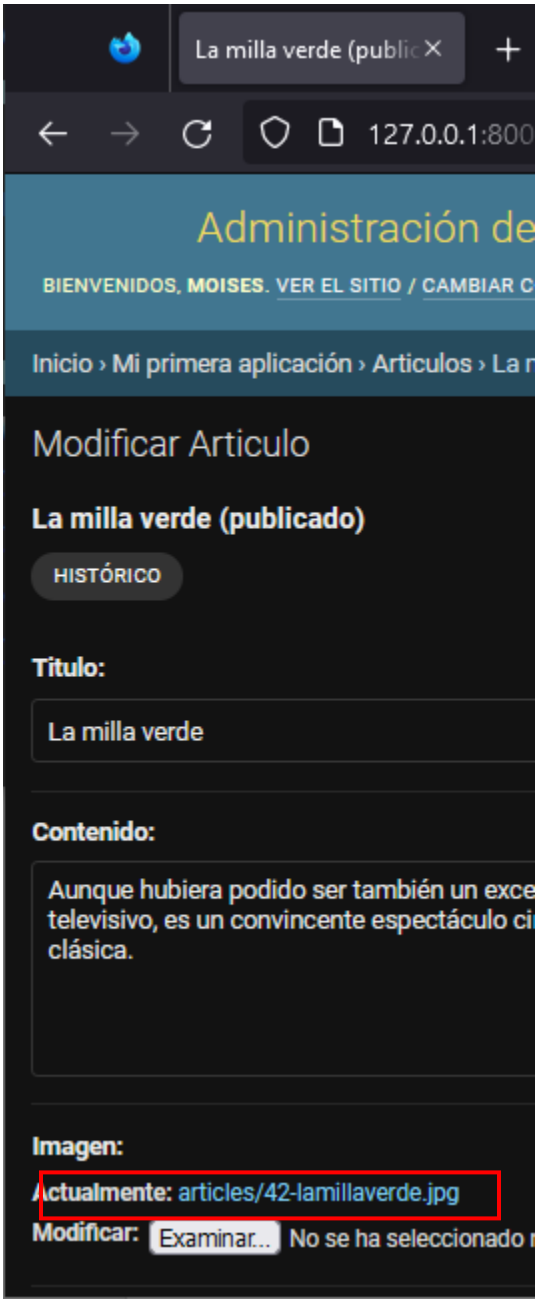
Configurar url para que lea el archivo imagen importamos lo realizado en settings anteriormente con la librería:

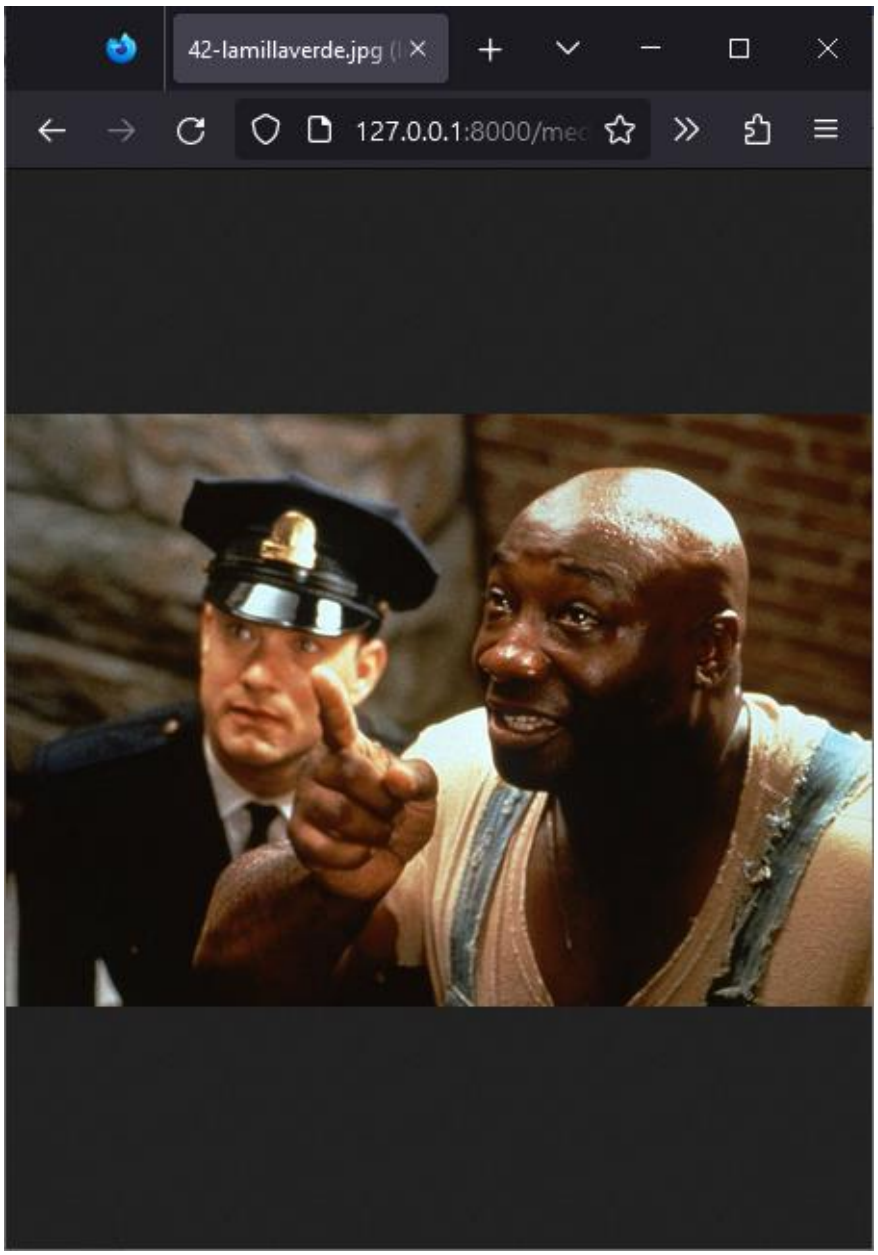
```python
"""
from django.contrib import admin
from django.urls import path
from django.conf import settings


#Importar app con mis vistas
```

urls.py

22-django > AprendiendoDjango > AprendiendoDjango > urls.py > ...

```
46      path('create-full-article/', views.create_full_article, name="create_full")
47  ]
48  # Configurar para cargar imagenes
49  if settings.DEBUG:
50      #Podemos tener accesible la función static que permite cargar de una url a un fichero estatico
51      #que pueda leer el framework
52      from django.conf.urls.static import static
53      urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
54
```





Mostrar imágenes subidas en la web – vamos al template **artículos.html**

articulos.html

22-django > AprendiendoDjango > miapp > templates > articulos.html

```
<ul>
    {% for articulo in articulos %}
        <li>
            {% if articulo.image != 'null' and articulo.image.url | length >= 1 %}
                <img src="{{articulo.image.url}}"/>
            {% endif %}
            <h4>{{articulo.id}} {{articulo.title}}</h4>
```

Maquetar con CSS listado de artículos con imágenes

1. Quitamos la lista de como se muestran los artículos → artículos.html quitar o reemplazar <ul> y

articulos.html

22-django > AprendiendoDjango > miapp > templates > articulos.html

```
10      {% endfor %}
11  {% endif %}
12
13          {% for articulo in articulos %}
14              <article class = "article-item">
15                  {% if articulo.image != 'null' and articulo.image.url | length >= 1 %}
16                      <img src="{{articulo.image.url}}"/>
17                  {% endif %}
18                  <h4><!--{{articulo.id}}--> {{articulo.title}}</h4>
19                  <span>{{articulo.created_at}}</span>
20                  <!--{% if articulo.public %}
21                      <strong>Publicado</strong> ESTO NO LO MOSTRAMOS
22                      {% else %}
23                      <strong>Privado</strong>
24                  {% endif %}-->
25                  <p>
26                      {{articulo.content}}
27                      <br/>
28                      <a href = "{% url 'borrar' id=articulo.id %}">Eliminar</a>
29                  </p>
30              </article>
31          {% endfor %}
32  {% endblock  %}
```

Ahora vamos a views.py para que se muestre solo los artículos publicados y aplicamos un filter

views.py

22-django > AprendiendoDjango > miapp > views.py > articulos

```
147  #Mostrar articulos
148  def articulos(request):#creacion de la vista articulos
149      #Creamos una template para listar varos articulos
150      #1. hacemos la peticion a la base de datos
151      #Creamos la variable articulos y LLamamos al modelo  Article, usamo
152      articulos = Article.objects.filter(public=True).order_by('-id') #Er
153      #dentro de articulos tenemos una lista de objetos un array de objet
154      """
155      #Consultas con condiciones filter para filtrar por un valor especif
```

Comprobar que en la lista de artículos solo se muestren los publicados

Ahora para maquetar las imágenes en articulo.html creamos unos div con clases y aplicamos CSS

```
# styles.css   articulos.html   views.py
22-django > AprendiendoDjango > miapp > templates > articulos.html
{% extends 'layout.html' %}
{% block title %} Listado de Artículos {% endblock %}
{% block content %}
<h1 class="title">Listado de Artículos</h1>
{% if messages %}
    {% for message in messages %}
        <div class="message">
            {{message}}
        </div>
    {% endfor %}
{% endif %}
        {% for articulo in articulos %}
            <article class="article-item">
                {% if articulo.image != 'null' and articulo.image.url|length >= 1 %}
                    <div class="image">
                        <img src="{{articulo.image.url}}"/>
                    </div>
                {% endif %}
                <div class="data">
                    <h2>{{articulo.title}}</h2>
                    <span class="date">{{articulo.created_at}}</span>
                    <p>
                        {{articulo.content}}
                        <a href = "{% url 'borrar' id=articulo.id %}" class="btn btn-delete">Eliminar</a>
                    </p>
                </div>
                <div class="clearfix"></div>
            </article>
        {% endfor %}
{% endblock  %}
```

Vamos a CSS

```
ul, ol {
    margin-left: 30px;
    margin-bottom: 20px;
}
/* para mantener el flujo de pagina */
.clearfix{
    clear: both;
    float: none;
}
body{
    background-color: #f2f2f2;
```

```
    margin-bottom: 10px;
    margin-top: 5px;
}
/*Estilos para maquetar la imagen*/
.article-item{
    margin-bottom: 20px;
    margin-top: 20px;
    padding: 15px;
    border-bottom: 1px solid #eee;
    clear:both;
}
.article-item .image{
    width: 200px;
    height: 200px;
    float: left;
    overflow: hidden;
}
.article-item .image img{
    height: 200px;
}
.article-item .data{
    float: left;
    padding: 15px;
    padding-top: 0px;
}
```

```
.article-item h2{
    display: block;
    font-size: 25px;
    color: #333333;
    margin-bottom: 10px;
}
/*Estilo de fecha */
.article-item span.date{
    display: block;
    font-size: 15px;
    color: gray;
    margin-bottom: 10px;
}
/* Estilos para boton eliminar*/
.btn{
    padding: 10px;
    background: rgb(199, 98, 98);
    color: white;
    border: 1px solid black;
    display: block;
    margin-top: 15px;
    width: 80px;
    text-align: center;
}
/*Estilos para el footer*/
footer{
    width: 1250px;
```

Cambiar nombre del panel de administración de Django

En **admin.py** al final realizamos el cambio

admin.py

22-django > AprendiendoDjango > miapp > admin.py > ...

```python
from django.contrib import admin
from .models import Article, Category
# Register your models here.

class ArticleAdmin(admin.ModelAdmin):
    #mostar campos de solo lectura
    readonly_fields = ('created_at', 'updated_at')
admin.site.register(Article, ArticleAdmin)
admin.site.register(Category)

#Configurar el titulo del panel
admin.site.site_header = "Django ADSO - SENA"
admin.site.site_title = "Django ADSO - SENA"
admin.site.index_title = "Panel de Gestión"
```