



DESARROLLO CON PYTHON

Contenido de formación



Programación



Python



Paradigma POO



Bases de datos SQL



Tkinter



Django



Flask

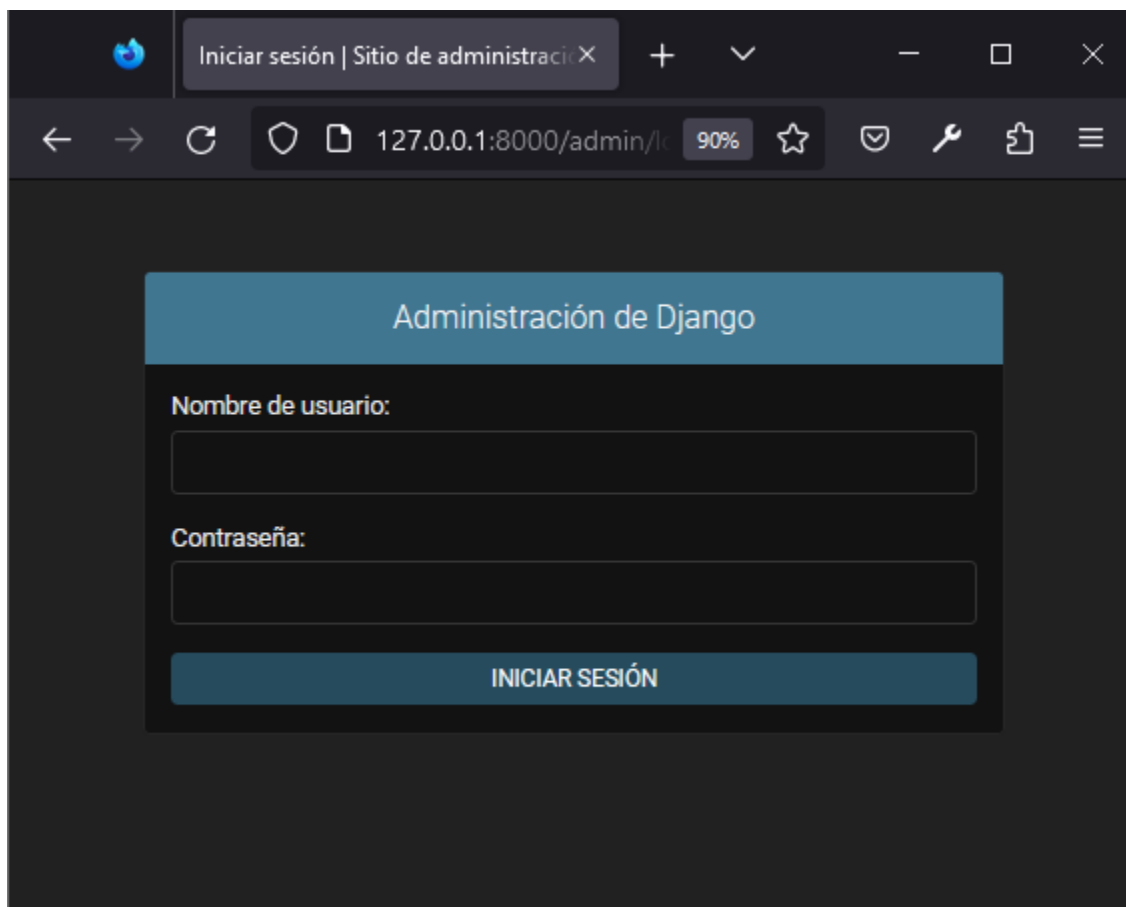
1. Primeros Instalación de SW
2. Variables y tipos de datos
3. Operadores (aritméticos, asignación)
4. Entrada y salida de datos
5. Estructuras de control
 - 5.1 Condicionales (operadores lógicos, operadores de comparación)
 - 5.2 Bucles (estructuras interactivas for, while)
6. Bloque de ejercicios de 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.



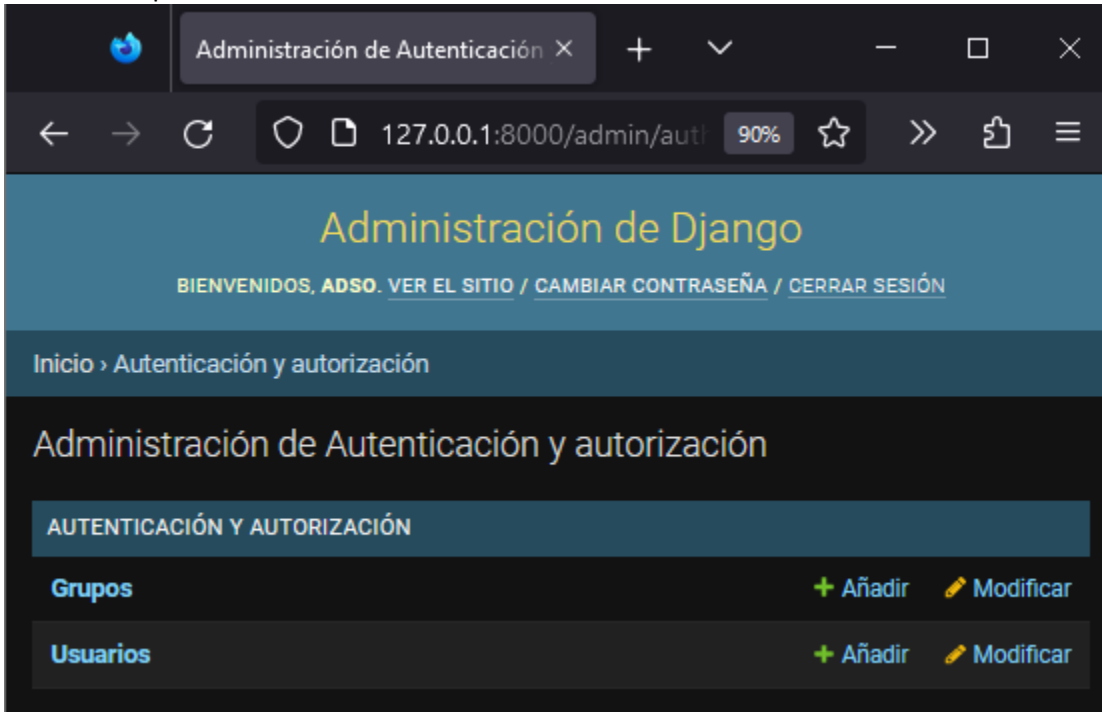
Creación de super usuario para entrar al panel de administración Django

```
Simbolo del sistema
C:\xampp\htdocs\sena-python\22-django\AprendiendoDjango>python manage.py createsuperuser
Nombre de usuario (leave blank to use 'mgv'): adso
Dirección de correo electrónico: moisog@gmail.com
Password:
Password (again):
Esta contraseña es demasiado corta. Debe contener al menos 8 caracteres.
Esta contraseña es demasiado común.
Esta contraseña es completamente numérica.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.

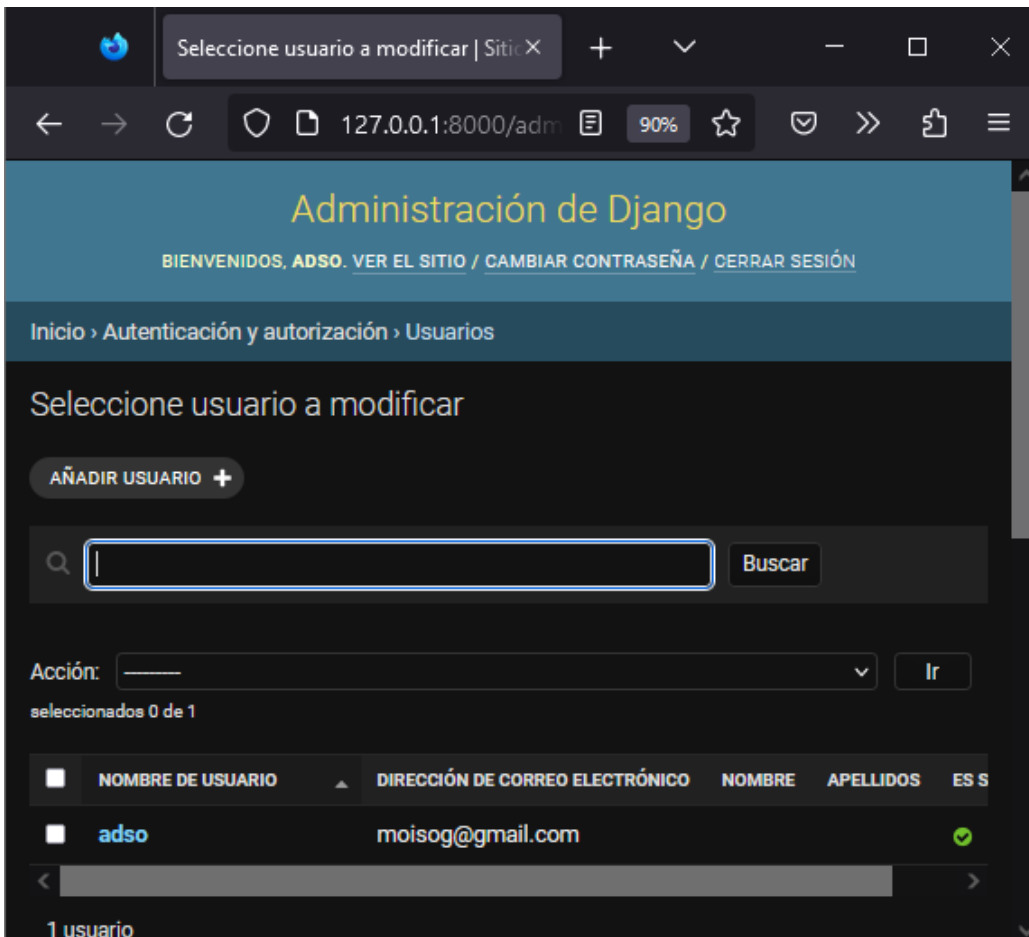
C:\xampp\htdocs\sena-python\22-django\AprendiendoDjango>python manage.py runserver
```



Entramos al panel de administración

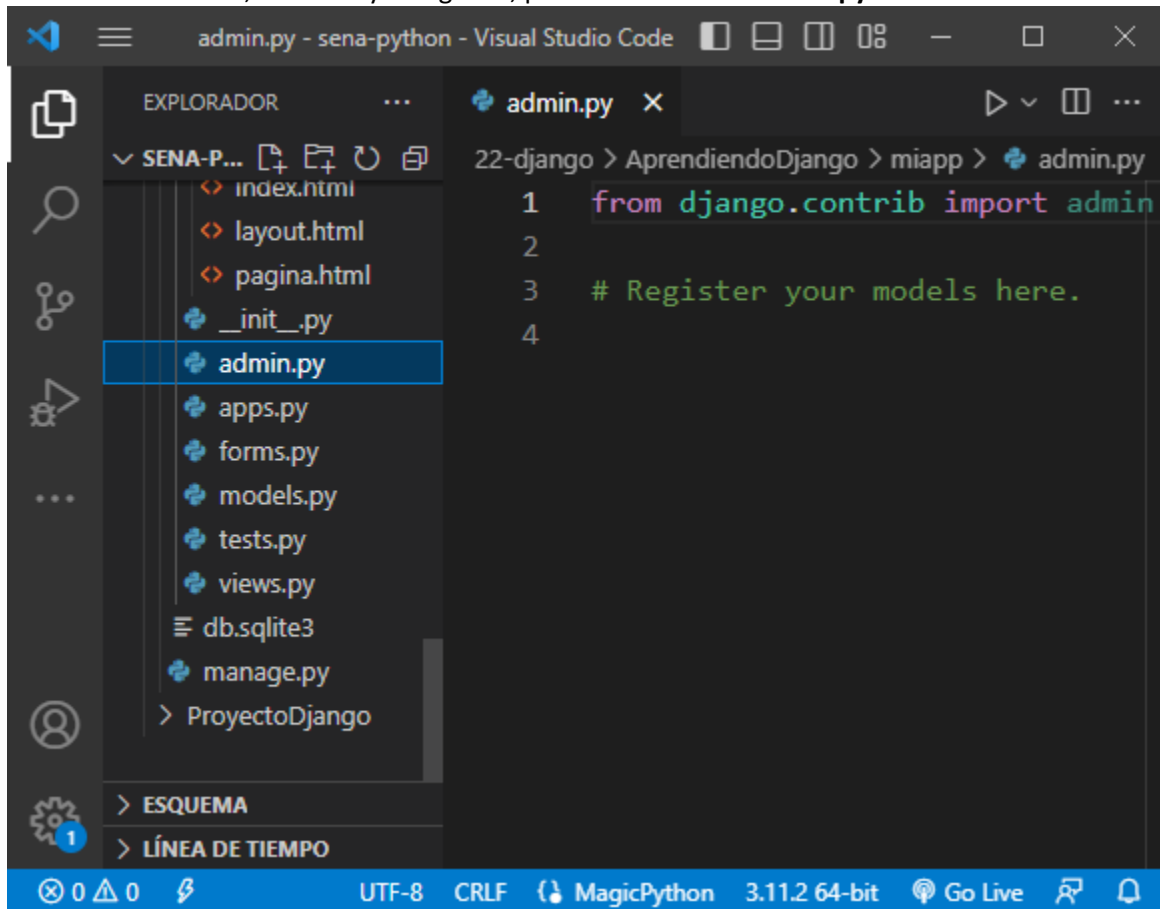


Podemos ver el usuario.. entre otras características



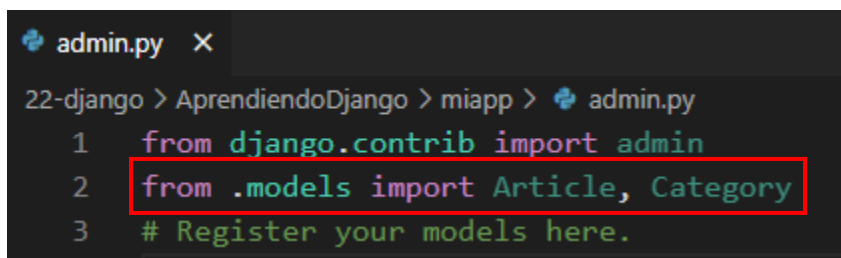


Gestión de modelos, artículos y categorías, para lo cual vamos **admin.py**

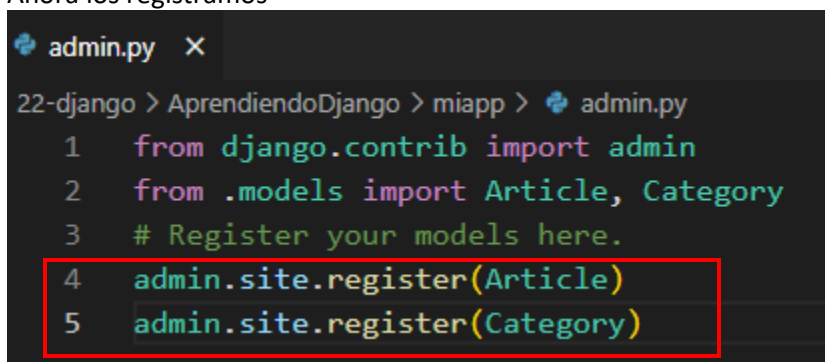


Ahora importamos los modelos **from .models import Article, Category**

Puedo hacerlo también utilizando el * para decirle que importe todos los modelos **from .models import ***

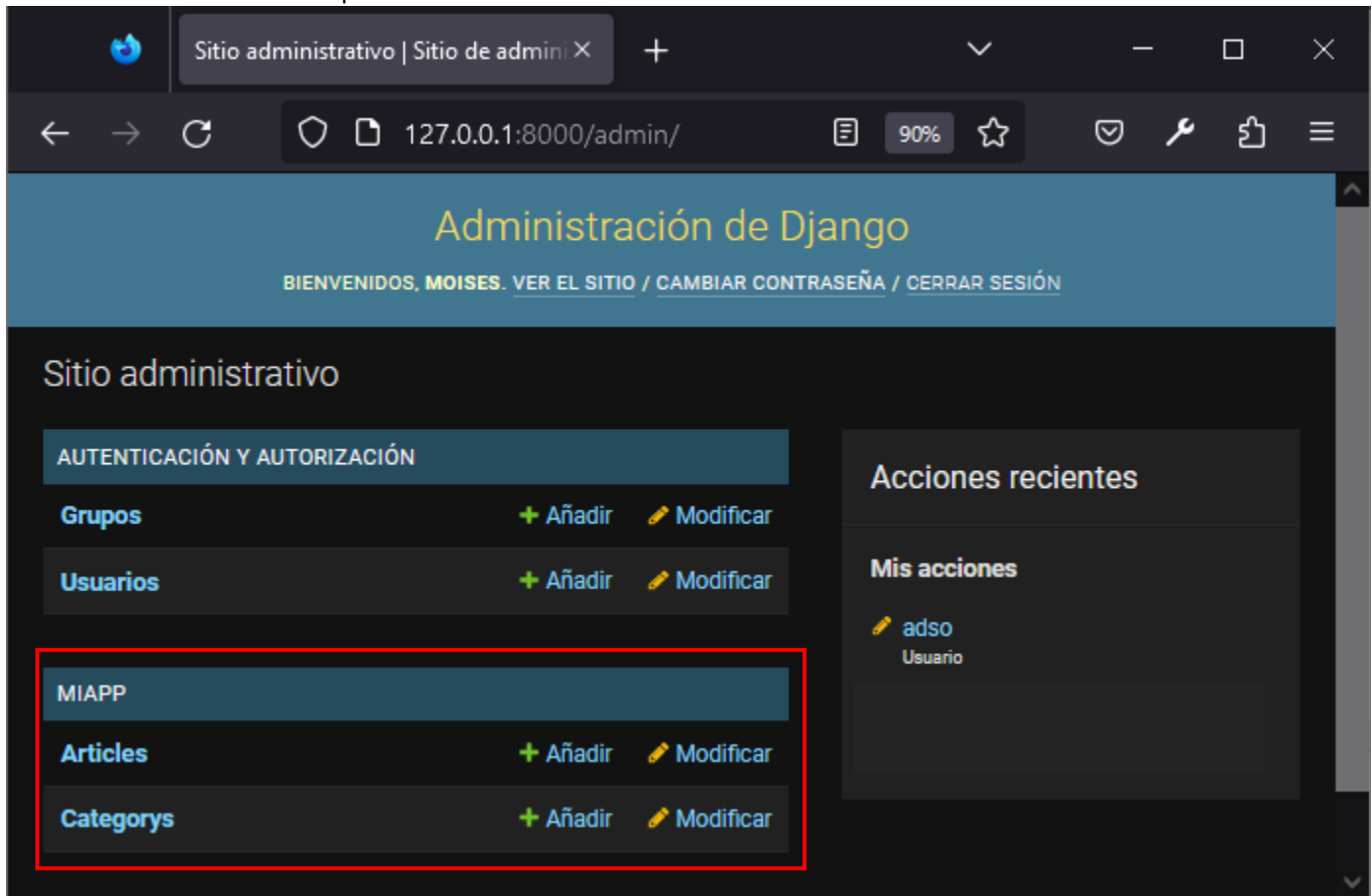


Ahora los registramos





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

6 class Article(models.Model): #Entidad
7
8     #propiedades *** Documentar cada tipo de dato
9     title = models.CharField(max_length=150)
10    content = models.TextField()
11    image = models.ImageField(default='null')
12    public = models.BooleanField()
13    created_at = models.DateTimeField(auto_now_add=True)
14    updated_at = models.DateTimeField(auto_now=True)
15
16    class Meta:
17        #poner nombre en singular
18        verbose_name = "Articulo"
19        verbose_name_plural = "Articulos"
20
21 class Category(models.Model): #Entidad
22    name = models.CharField(max_length=110)
23    description = models.CharField(max_length=250)
24    created_at = models.DateField()
25
26    class Meta:
27        #poner nombre en singular
28        verbose_name = "Categoria"
29        verbose_name_plural = "Categorias"
```

MIAPP		
Articulos	+ Añadir	✎ Modificar
Categorias	+ Añadir	✎ Modificar



Manipular propiedades de los modelos

Traducir los parámetros de los campos a español

```
models.py X
22-django > AprendiendoDjango > miapp > models.py > Article

6 class Article(models.Model): #Entidad
7
8     #propiedades *** Documentar cada tipo de dato
9     title = models.CharField(max_length=150, verbose_name="Titulo")
10    content = models.TextField(verbose_name="Contenido")
11    image = models.ImageField(default='null', verbose_name="Imagen")
12    public = models.BooleanField(verbose_name="¿Publicado?")
13    created_at = models.DateTimeField(auto_now_add=True)
14    updated_at = models.DateTimeField(auto_now=True)
15
```

Cambiar el nombre de las apps

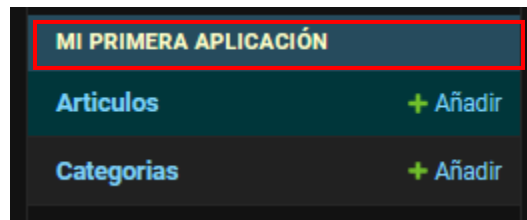




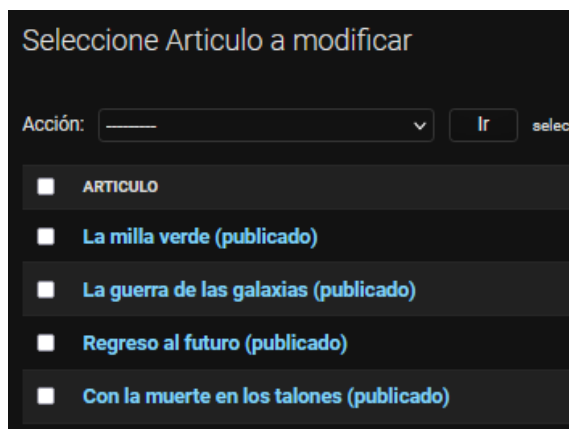
```
from django.apps import AppConfig

class MiappConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'miapp'
    verbose_name = 'Mi primera aplicación'
```

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'miapp.apps.MiappConfig',
]
```



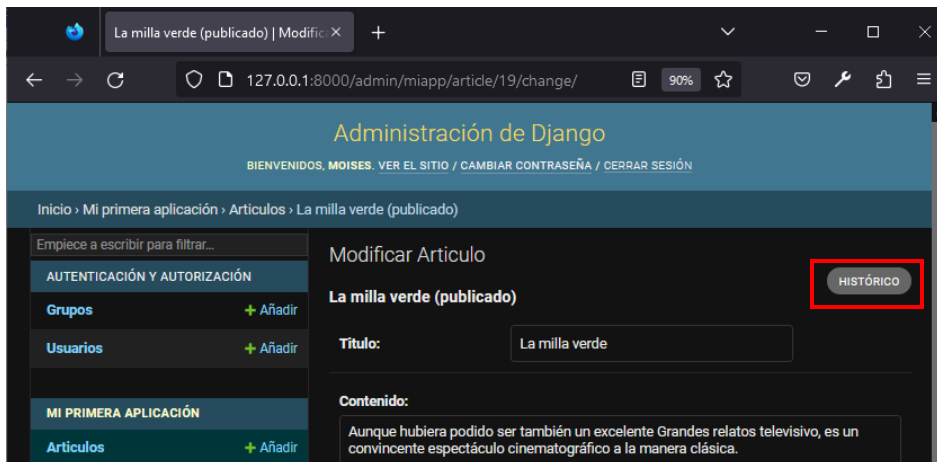
Método mágico para imprimir objetos (modelos) así esta vamos a cambiarlo:
para acceder a mas métodos mágicos <https://rszalski.github.io/magicmethods/>





```
models.py X
22-django > AprendiendoDjango > miapp > models.py > Article > __str__
19     verbose_name_plural = "Articulos"
20
21     def __str__(self):
22         if self.public:
23             public = "(publicado)"
24         else:
25             public = "(privado)"
26         return f"{self.title} {public}"
27
28 class Category(models.Model): #Entidad
```

Mostar campos de solo lectura:



```
admin.py X models.py
22-django > AprendiendoDjango > miapp > admin.py > ...
1  from django.contrib import admin
2  from .models import Article, Category
3  # Register your models here.
4
5  class ArticleAdmin(admin.ModelAdmin):
6      #mostar campos de solo lectura
7      readonly_fields = ('created_at', 'updated_at')
8  admin.site.register(Article, ArticleAdmin)
9  admin.site.register(Category)
```



La milla verde (publicado) | Modificar: X

127.0.0.1:8000/admin/miapp/article/19/change/ 90%

Inicio > Mi primera aplicación > Artículos > La milla verde (publicado)

Empiece a escribir para filtrar...

AUTENTICACIÓN Y AUTORIZACIÓN

Grupos + Añadir

Usuarios + Añadir

MI PRIMERA APLICACIÓN

Artículos + Añadir

Categorías + Añadir

Modificar Artículo

La milla verde (publicado) HISTÓRICO

Título: La milla verde

Contenido:

Aunque hubiera podido ser también un excelente Grandes relatos televisivo, es un convincente espectáculo cinematográfico a la manera clásica.

Imagen: Actualmente: null
Modificar: Examinar... No se ha seleccionado ningún archivo.

☒ ¿Publicado?

Created at: 14 de marzo de 2023 a las 14:04

Updated at: 14 de marzo de 2023 a las 14:04

Cambiar el **verbose_name** a cada campo

Imagen: Actualmente: null
Modificar: Examinar... No se ha seleccionado ningún archivo.

☒ ¿Publicado?

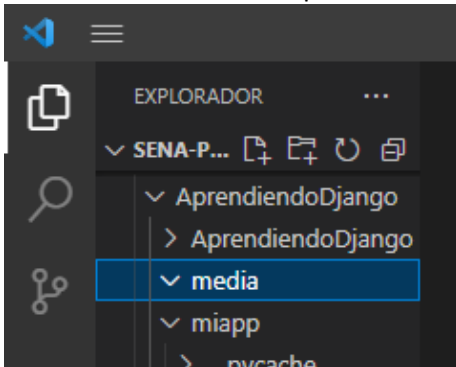
Creado: 14 de marzo de 2023 a las 14:04

Modificado: 14 de marzo de 2023 a las 14:04



Configurar la aplicación para subir imágenes

Primero creamos una carpeta a nivel superior de **AprendiendoDjango** llamada **media**



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

```
settings.py X
22-django > AprendiendoDjango > AprendiendoDjango > settings.py > ...
117 # Static files (CSS, JavaScript, Images)
118 # https://docs.djangoproject.com/en/4.1/howto/static-files/
119
120 STATIC_URL = 'static/'
121
122 #Media files
123 MEDIA_URL = '/media/' #Carpeta donde se guardaran las imagenes
124 MEDIA_ROOT = os.path.join(BASE_DIR, "media")
125
126 # Default primary key field type
127 # https://docs.djangoproject.com/en/4.1/ref/settings/#default-au
128
129 DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```

Importamos **os**

```
settings.py X models.py
22-django > AprendiendoDjango > Aprendiendo
12
13 from pathlib import Path
14 import os
15
```



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

```
settings.py  models.py X
22-django > AprendiendoDjango > miapp > models.py > Article
8      #propiedades *** Documentar cada tipo de dato
9      title = models.CharField(max_length=150, verbose_name="Titulo")
10     content = models.TextField(verbose_name="Contenido")
11     image = models.ImageField(default='null', verbose_name="Imagen", upload_to="articles")
12     public = models.BooleanField(verbose_name="¿Publicado?")
13     created_at = models.DateTimeField(auto_now_add=True, verbose_name="Creado")
14     updated_at = models.DateTimeField(auto_now=True, verbose_name="Modificado")
15
16     class Meta:
```

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

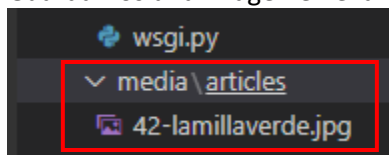
```
Simbolo del sistema
Performing system checks...

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:

```
urls.py  X
22-django > AprendiendoDjango > AprendiendoDjango > urls.py > ...
15     """
16     from django.contrib import admin
17     from django.urls import path
18     from django.conf import settings
19
20
21     #Importar app con mis vistas
```



```
urls.py
22-django > AprendiendoDjango > AprendiendoDjango > urls.py > ...
45 path('create-article/', views.create_article, name='create'),
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
```

La milla verde (public X) +

127.0.0.1:8000

Administración de

BIENVENIDOS, MOISES. VER EL SITIO / CAMBIAR C

Inicio > Mi primera aplicación > Articulos > La r

Modificar Artículo

La milla verde (publicado)

HISTÓRICO

Titulo:

La milla verde

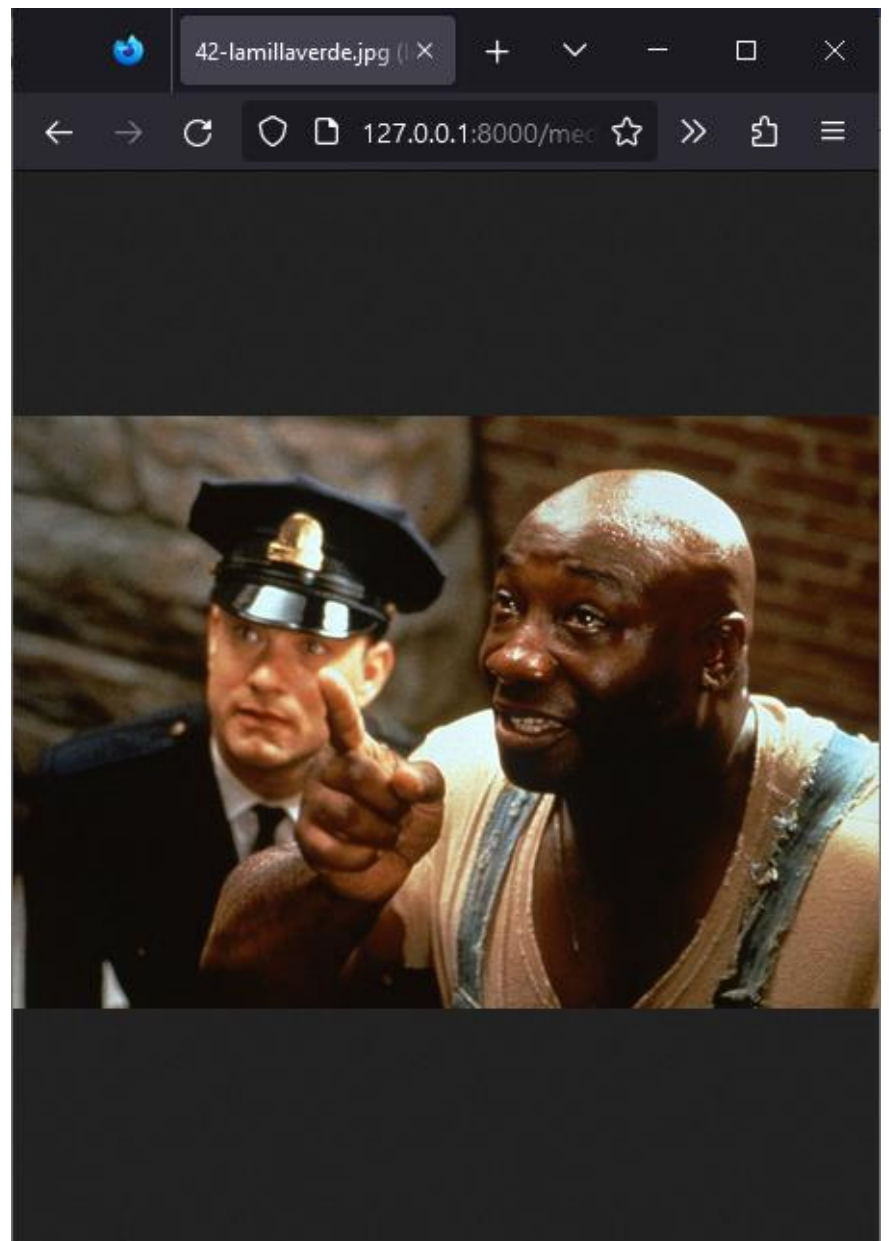
Contenido:

Aunque hubiera podido ser también un exce
televisivo, es un convincente espectáculo ci
clásica.

Imagen:

Actualmente: [articles/42-lamillaverde.jpg](#)

Modificar: Examinar... No se ha seleccionado r





Mostrar imágenes subidas en la web – vamos al template `articulos.html`

```
articulos.html X
22-django > AprendiendoDjango > miapp > templates > <articulos.html
12 <ul>
13     {% for articulo in articulos %}
14     <li>
15         {% if articulo.image != 'null' and articulo.image.url | length >= 1 %}
16         
17         {% endif %}
18     <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 y

```
articulos.html X
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                 
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 X
22-django > AprendiendoDjango > miapp > views.py > articulos
140
147 #Mostrar articulos
148 def articulos(request):#creacion de la vista articulos
149     #Creamos una template para listar varios articulos
150     #1. hacemos la peticion a la base de datos
151     #Creamos la variable articulos y Llamamos al modelo Article, usamos
152     articulos = Article.objects.filter(public=True).order_by('-id') #En
153     #dentro de articulos tenemos una lista de objetos un array de objetos
154     """
155     #Consultas con condiciones filter para filtrar por un valor específico
```

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



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

```
# styles.css | <div> articulos.html X | views.py
22-django > AprendiendoDjango > miapp > templates > <div> articulos.html
1  {% extends 'layout.html' %}
2  {% block title %} Listado de Artículos {% endblock %}
3  {% block content %}
4  <h1 class="title">Listado de Artículos</h1>
5  {% if messages %}
6      {% for message in messages %}
7          <div class="message">
8              {{message}}
9          </div>
10     {% endfor %}
11 {% endif %}
12     {% for articulo in articulos %}
13         <article class="article-item">
14             {% if articulo.image != 'null' and articulo.image.url|length >= 1 %}
15                 <div class="image">
16                     
17                 </div>
18             {% endif %}
19             <div class="data">
20                 <h2>{{articulo.title}}</h2>
21                 <span class="date">{{articulo.created_at}}</span>
22                 <p>
23                     {{articulo.content}}
24                     <a href = "{% url 'borrar' id=articulo.id %}" class="btn btn-delete">Eliminar</a>
25                 </p>
26             </div>
27             <div class="clearfix"></div>
28         </article>
29     {% endfor %}
30 {% endblock %}
```

Vamos a CSS

```
12  ul, ol {
13      margin-left: 30px;
14      margin-bottom: 20px;
15  }
16  /* para mantener el flujo de pagina */
17  .clearfix{
18      clear: both;
19      float: none;
20  }
21  body{
22      background-color: #f2f2f2;
```




```
165     margin-bottom: 10px;
166     margin-top: 5px;
167 }
168 /*Estilos para maquetar la imagen*/
169 .article-item{
170     margin-bottom: 20px;
171     margin-top: 20px;
172     padding: 15px;
173     border-bottom: 1px solid #eee;
174     clear: both;
175 }
176 .article-item .image{
177     width: 200px;
178     height: 200px;
179     float: left;
180     overflow: hidden;
181 }
182 .article-item .image img{
183     height: 200px;
184 }
185 .article-item .data{
186     float: left;
187     padding: 15px;
188     padding-top: 0px;
189 }
```

```
190 .article-item h2{
191     display: block;
192     font-size: 25px;
193     color: #333333;
194     margin-bottom: 10px;
195 }
196 /*Estilo de fecha */
197 .article-item span.date{
198     display: block;
199     font-size: 15px;
200     color: gray;
201     margin-bottom: 10px;
202 }
203 /* Estilos para boton eliminar*/
204 .btn{
205     padding: 10px;
206     background: rgb(199, 98, 98);
207     color: white;
208     border: 1px solid black;
209     display: block;
210     margin-top: 15px;
211     width: 80px;
212     text-align: center;
213 }
214 /*Estilos para el footer*/
215 footer{
216     width: 1250px;
217     background-color: #141414;
```





Cambiar nombre del panel de administración de Django

En `admin.py` al final realizamos el cambio

```
admin.py X
22-django > AprendiendoDjango > miapp > admin.py > ...
1  from django.contrib import admin
2  from .models import Article, Category
3  # Register your models here.
4
5  class ArticleAdmin(admin.ModelAdmin):
6      #mostar campos de solo lectura
7      readonly_fields = ('created_at', 'updated_at')
8  admin.site.register(Article, ArticleAdmin)
9  admin.site.register(Category)
10
11  #Configurar el titulo del panel
12  admin.site.site_header = "Django ADSO - SENA"
13  admin.site.site_title = "Django ADSO - SENA"
14  admin.site.index_title = "Panel de Gestión"
```

