

---

---

---

# TRABAJANDO CON LA ADMINISTRACIÓN DE BD

PARTE II



## AGREGAR IMAGEN A NUESTRO USUARIO

- Debemos instalar un módulo externo llamado Pillow para que Django procese las imágenes
- Dentro de tu proyecto instala Pillow con el comando pip:
  - **Mac:** `python3 -m pip install Pillow`
  - **Windows:** `py -m pip install Pillow`

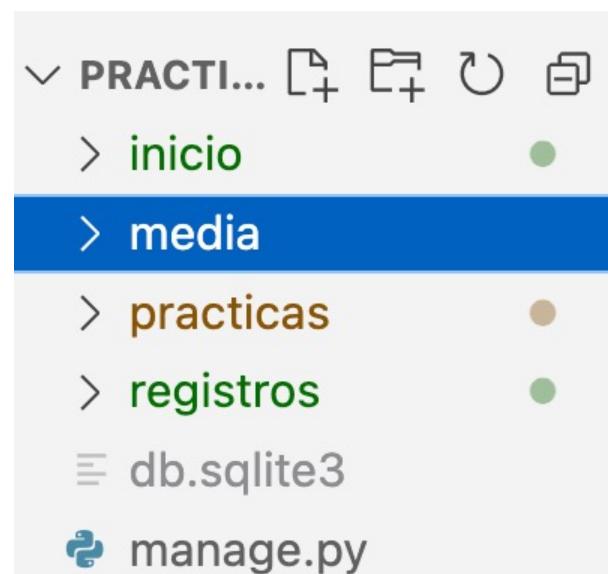
Installing collected packages: Pillow  
Successfully installed Pillow-10.3.0

## PILLOW

- Es una biblioteca de imágenes de Python que agrega capacidades de procesamiento de imágenes a su intérprete de Python.
- Esta biblioteca proporciona un amplio soporte de formato de archivo, una representación interna eficiente y capacidades de procesamiento de imágenes bastante poderosas.
- La biblioteca de imágenes principal está diseñada para un acceso rápido a los datos almacenados en unos pocos formatos de píxeles básicos. Debe proporcionar una base sólida para una herramienta general de procesamiento de imágenes.

## ADMINISTRACIÓN DE ARCHIVOS

- Creamos el directorio **media** en la raiz del proyecto:



## ADMINISTRACIÓN DE ARCHIVOS

- Django almacena archivos localmente, usando la configuración `MEDIA_ROOT` y `MEDIA_URL`.
- `MEDIA_ROOT`: Ruta absoluta del sistema de archivos al directorio que contendrá los archivos cargados por el usuario .
- `MEDIA_URL`: Se encarga de los medios de comunicación sirven de `MEDIA_ROOT`, utilizado para la gestión de archivos almacenados .

## ADMINISTRACIÓN DE ARCHIVOS

### settings.py

- Agregar al final de todo el código lo siguiente:

```
# Archivos fotográficos
```

```
MEDIA_URL = '/media/'  
MEDIA_ROOT = os.path.join(BASE_DIR, "media")
```

## AGREGAMOS EL CAMPO IMAGEN EN EL MODELO DE LA TABLA

```
class Alumnos(models.Model): #Define la estructura de nuestra tabla
    matricula = models.CharField(max_length=12,verbose_name="Matricula") #Texto corto
    nombre = models.TextField() #Texto largo
    carrera = models.TextField()
    turno = models.CharField(max_length=10)
    imagen = models.ImageField(null=True,upload_to="fotos",verbose_name="Fotografía")
    created = models.DateTimeField(auto_now_add=True) #Fecha y tiempo
    updated = models.DateTimeField(auto_now=True)
```

**upload\_to** subira las imágenes a **media/fotos** si la carpeta **fotos** no existe se creará automáticamente.

## APLICANDO MIGRACIONES A LA BASE DE DATOS

- Ejecutar los comandos makemigrations y migrate para aplicar los cambios

```
elena@MacBook-Pro-de-Elena prueba % python3 manage.py makemigrations registros  
Migrations for 'registros':
```

```
    registros/migrations/0002.Alter_alumnos_options_alumnos_imagen_and_more.py  
        - Change Meta options on alumnos  
        - Add field imagen to alumnos  
        - Alter field matricula on alumnos
```

```
elena@MacBook-Pro-de-Elena prueba % python3 manage.py migrate registros
```

```
Operations to perform:
```

```
    Apply all migrations: registros
```

```
Running migrations:
```

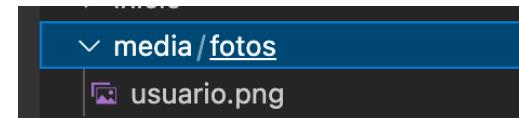
```
    Applying registros.0002.Alter_alumnos_options_alumnos_imagen_and_more... OK
```

```
elena@MacBook-Pro-de-Elena prueba %
```

- Ejecutar el servidos, ingresa al módulo de alumnos y procederemos a cargar una imagen para el alumnos existente. Observaremos que la imagen se guarda en la ruta marcada **media/fotos**

Carrera:	TIC
Turno:	Matutino
Fotografía:	<input type="button" value="Seleccionar archivo"/> No se eligió archivo
Creado:	25 de junio de 2021, 2:28 a. M.
Actualizado:	25 de junio de 2021, 2:28 a. M.
<input type="button" value="Borrar"/> <input type="button" value="Guardar y agregar otro"/> <input type="button" value="Guardar y continuar editando"/>	

Fotografía: Actualmente: fotos / usuario.png  
 Cambiar:  No se eligió archivo



Si intentas acceder a la foto desde el vínculo, se mostrará un error.

## Página no encontrada (404)

Método de solicitud: OBTENER

URL de solicitud: http://127.0.0.1:8000/media/fotos/usuario.png

Usando la URLconf definida en prueba.urls, Django probó estos patrones de URL, en

## VISUALIZAR LA IMAGEN GUARDADA

- Si intentamos abrir la imagen guardada desde la ventana de administración, observaremos un error, esto ocurre porque el servidor de desarrollo no puede cargar estos archivos, sin embargo en producción los servidores como Apache o Nginx se encargarían de el.
- Podemos activarlo si estamos en modo de debug podemos direccionar a la ruta específica registrando lo siguiente en el archivo **urls.py**

```
from django.contrib import admin
from django.urls import path
from inicio import views
from django.conf import settings
#permite acceder a las variables MEDIA_URL y MEDIA_ROOT que
almacenan la ubicación de nuestras imágenes
```

## URLS.PY

**Al final de todo el código**

Fotografía:

Actualmente: fotos / usuario.png

Cambiar:  No se eligió archivo

Ahora puedes ejecutar tu servicios y visualizar la imagen guardada.

