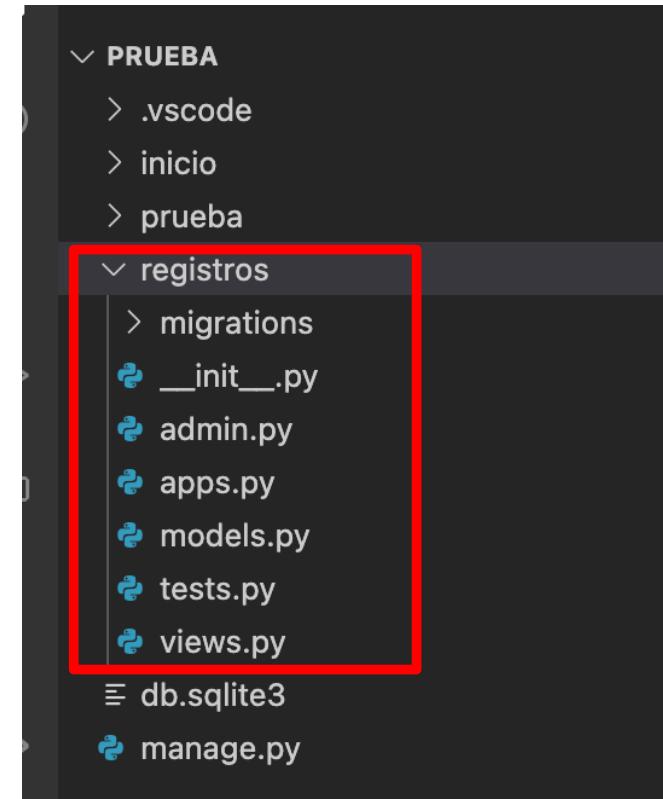

ADMIN DJANGO

PRIMERA PARTE



INTERACCIÓN CON BASE DE DATOS

- Crearemos una nueva APP. → registros
 - `python manage.py startapp registros`



MODELS.PY → APP REGISTROS

```
models.py ●  
registros > models.py  
1  from django.db import models  
2  
3  # Create your models here.
```

MODELOS

- Un Django model es una tabla de la base de datos, los atributos de ese modelo se convierten en las columnas de esa tabla esos atributos reciben el nombre de django fields los cuales manejan automáticamente las conversiones de tipos de datos para la base de datos que estemos usando.
- Una de las grandes características de Django es su ORM (Object–relational mapping), gracias a el no tenemos que escribir ninguna consulta de base de datos, e incluso se recomienda NO escribir una al usar Django.

MODELOS

- ORM convierte los Django models y todas las operaciones que realiza con las consultas de base de datos correspondientes. Esto significa que toda la manipulación se hará ahora con los objetos de Python creados a partir de ese modelo, y todo el material de la base de datos subyacente será cuidado por el ORM de Django.
- El ORM de Django es compatible con todas las bases de datos principales como Postgres , MySQL , sqlite3 y otras. Esto significa que si deseamos cambiar de una base de datos a otra, podemos hacerlo sin cambiar una sola línea de la lógica de su aplicación, solo cambiamos la cadena de la base de datos en el archivo de configuración `settings.py`.

MODELS.PY → APP REGISTROS

```
from django.db import models

class Alumnos(models.Model): #Define la estructura de nuestra tabla
    matricula = models.CharField(max_length=12) #Texto corto
    nombre = models.TextField() #Texto largo
    carrera = models.TextField()
    turno = models.CharField(max_length=10)
    created = models.DateTimeField(auto_now_add=True) #Fecha y tiempo
    updated = models.DateTimeField(auto_now=True)
```

SETTINGS.PY

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'inicio',
    'registros',
]
```

APLICAMOS EL CAMBIO AL MODELO

- `python manage.py makemigrations registros`

```
Migrations for 'registros':  
  registros/migrations/0001_initial.py  
    - Create model Alumnos
```

APLICAMOS EL CAMBIO A LA BASE DE DATOS

- `python manage.py migrate registros`

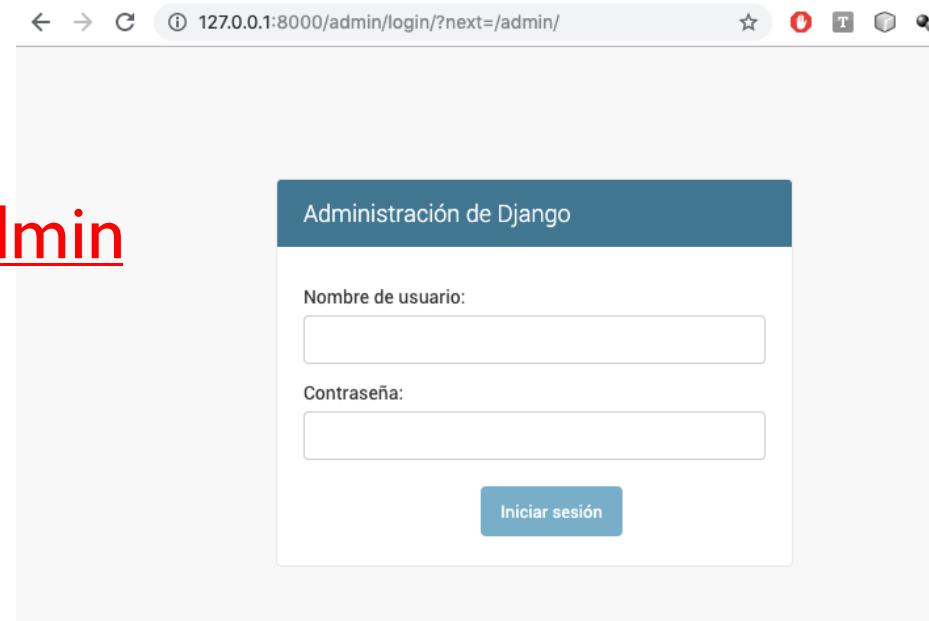
```
Operations to perform:
  Apply all migrations: registros
Running migrations:
  Applying registros.0001_initial... OK
elena@MacBook-Pro-de-Elena prueba %
```

Cualquier cambio al modelo de la tabla, deberá implicar la ejecución de los dos comandos (`makemigrations` y `migrate`)

UTILIZAR EL PANEL ADMINISTRADOR

Ejecutamos el servidor y accedemos al administrador colocando en la ruta /admin

<http://127.0.0.1:8000/admin>



URLS

```
20  urlpatterns = [
21      path('admin/', admin.site.urls),
22      path('',views.principal,name="Principal"),
23      path('contacto/',views.contacto,name="Contacto"),
24      path('formulario/',views.formulario,name="Formulario"),
25      path('ejemplo/',views.ejemplo,name="Ejemplo"),
26  ]
27
```

El panel de administración se encuentra definido por default

CREANDO UN USUARIO

- Creamos un usuario administrador
 - **python manage.py createsuperuser**

```
^C(django2) Air-de-Elena:novenoAHerencia elena$ python manage.py createsuperuser
Username (leave blank to use 'elena'): elena
Email address: elena@utm.com
Password:
Password (again):
This password is too short. It must contain at least 8 characters.
This password is too common.
This password is too common.
This password is entirely numeric.
```

Superuser created successfully.

ACCEDEMOS AL PANEL DE ADMINISTRACIÓN



The screenshot shows the Django Admin interface at the URL `127.0.0.1:8000/admin/`. The title bar says "Administración de Django". The top right shows a welcome message for "ELENA" and links to "VER SITIO", "CAMBIAR CONTRASEÑA", and "CERRAR SESIÓN". The main content area is titled "AUTENTICACION Y AUTORIZACION" and contains two models: "Grupos" and "Usuarios". Each model has "Agregar" and "Cambio" buttons. To the right, a sidebar titled "Acciones recientes" shows "Mis acciones" with the message "Ninguno disponible".

En el último ejemplo pudiste ingresar con tu usuario al panel de administración ahora procederemos a cambiar algunos elementos.

REGISTROS/ADMIN.PY

```
registros > admin.py
1  from django.contrib import admin
2
3  # Register your models here.
4  |
```

REGISTROS/ADMIN.PY

```
from django.contrib import admin
from .models import Alumnos
```

```
# Register your models here.
admin.site.register(Alumnos)
```

Alumno es el nombre de nuestro clase
en el archivo models.py de la app registro

- Actualiza el navegador, deberás observar que se muestra el módulo de registros con el acceso al modelo de alumnos.
- Observa como se agrega una “s” al final del nombre del modelo, en tu proyecto, el administrador agrega la s por default.

Django administration

Site administration

AUTHENTICATION AND AUTHORIZATION

| Groups | + Add | Change |
|--------|-----------------------|------------------------|
| Users | + Add | Change |

REGISTROS

| Alumnos | + Add | Change |
|---------|-----------------------|------------------------|
|---------|-----------------------|------------------------|

CAMBIANDO EL TITULO DE LA APP EN EL PANEL DE ADMINISTRADOR

registros/apps.py

REGISTROS

Alumnos

```
from django.apps import AppConfig
```

copiamos

```
class RegistrosConfig(AppConfig):  
    name = 'registros'
```

```
    verbose_name='Módulos'
```

Agregamos

SETTINGS.PY

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'inicio',
    'registros.apps.RegistrosConfig',
]
```

MÓDULOS

Alumnos

REGISTROS/MODELS.PY

- Cambiaremos el título del modelo y daremos un ordenamiento a los registros

```
from django.db import models

class Alumnos(models.Model): #Define la estructura de nuestra tabla
    matricula = models.CharField(max_length=12) #Texto corto
    nombre = models.TextField() #Texto largo
    carrera = models.TextField()
    turno = models.CharField(max_length=10)
    created = models.DateTimeField(auto_now_add=True) #Fecha y tiempo
    updated = models.DateTimeField(auto_now=True) #Fecha y tiempo

    class Meta:
        verbose_name = "Alumno"
        verbose_name_plural = "Alumnos"
        ordering = ["-created"]
        #el menos indica que se ordenara del más reciente al más viejo
```

MÓDULOS
Alumnos

AGREGAMOS UN ALUMNO

AUTENTICACION Y AUTORIZACION

Grupos [+ Agregar](#)

Usuarios [+ Agregar](#)

MÓDULOS

Alumnos [+ Agregar](#)

Agregar alumno

Matricula:

Nombre:

Carrera:

Turno:

[Guardar y agregar otro](#) [Guardar y continuar editando](#) [GUARDAR](#)

OBSERVARÁS QUE SE MUESTRA COMO OBJETO

Seleccione el alumno para cambiar

Acción: Ir 0 de 1 seleccionado

| | |
|--------------------------|----------------------|
| <input type="checkbox"/> | ALUMNO |
| <input type="checkbox"/> | Objeto de alumno (1) |
| 1 alumno | |

REGISTROS/MODELS.PY

- Cambiaremos la apariencia de la lista de registros para mostrar el nombre del alumno

```
class Meta:  
    verbose_name = "Alumno"  
    verbose_name_plural = "Alumnos"  
    ordering = ["-creado"]  
    #el menos indica que se ordenara del más reciente al mas viejo  
  
def __str__(self):  
    return self.nombre  
    #Indica que se mostrára el nombre como valor en la tabla
```

Seleccione el alumno para cambiar

Acción: 0 de 1 seleccionado

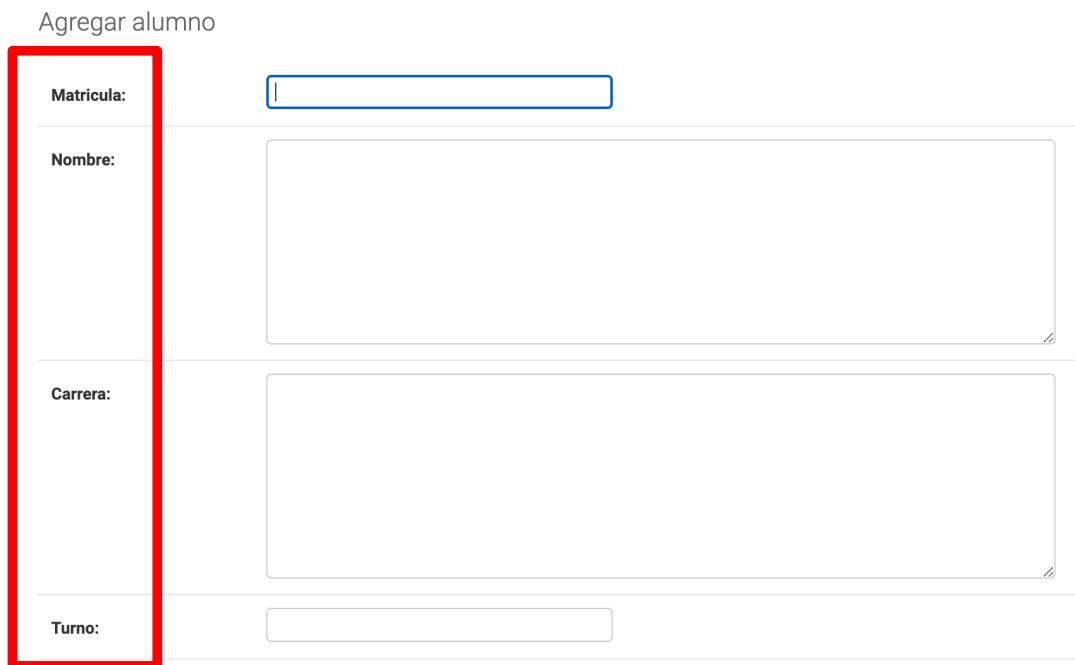
| |
|---------------------------------|
| <input type="checkbox"/> ALUMNO |
| <input type="checkbox"/> Juan |

REGISTROS/MODELS.PY

- Es posible cambiar los nombre de los campos del formulario del administrador

Agregar alumno

| | |
|------------|----------------------|
| Matricula: | <input type="text"/> |
| Nombre: | <input type="text"/> |
| Carrera: | <input type="text"/> |
| Turno: | <input type="text"/> |



REGISTROS/MODELS.PY

- Cambiaremos el título de los campos

```
class Alumnos(models.Model):  
    matricula = models.CharField(max_length=12, verbose_name="Mat")
```

Mat:

VISUALIZACIÓN DE CAMPOS AUTOMÁTICOS

- Si acceden al usuario registrado, observarán que los campos automáticos (fechas de creación y modificación) no son visibles.

Cambiar alumno

Juan

HISTORIA

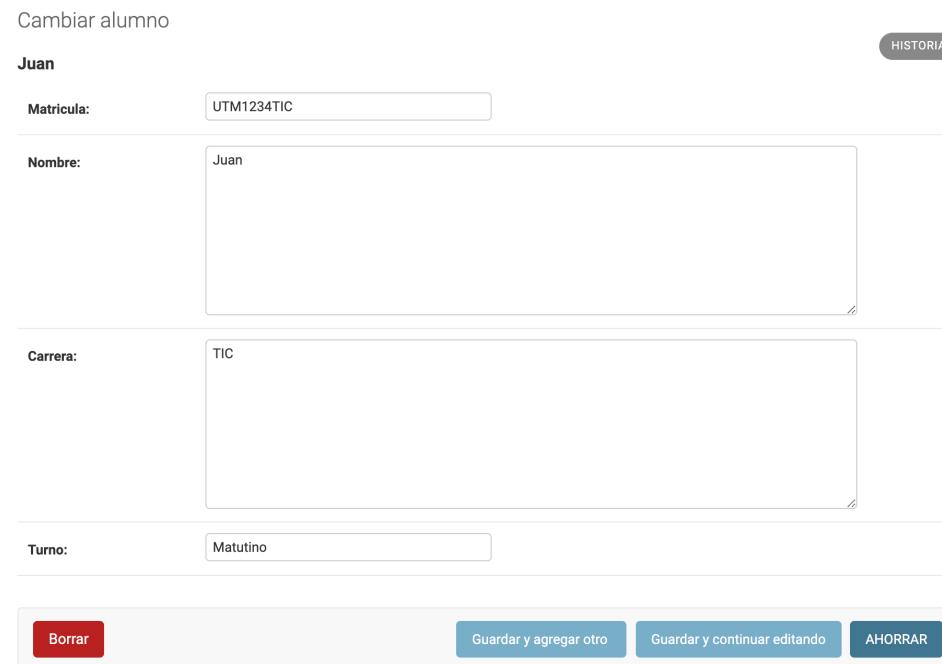
Matrícula: UTM1234TIC

Nombre: Juan

Carrera: TIC

Turno: Matutino

Borrar Guardar y agregar otro Guardar y continuar editando AHORRAR



VISUALIZACIÓN DE CAMPOS AUTOMÁTICOS

■ **registros/admin.py**

```
from django.contrib import admin
from .models import Alumnos
```

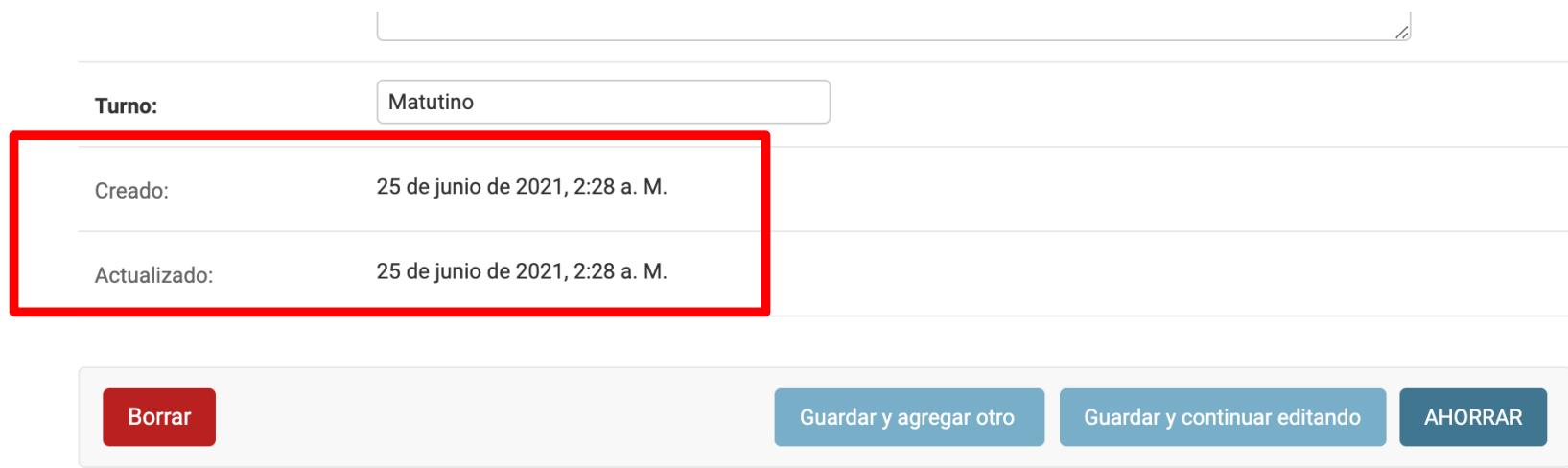
```
# Register your models here.
```

```
class AdministrarModelo(admin.ModelAdmin):
    readonly_fields = ('created', 'updated')
```

```
admin.site.register(Alumnos, AdministrarModelo)
```

VISUALIZACIÓN DE CAMPOS AUTOMÁTICOS

- Actualiza el navegador y deberás observar al final del formulario las fechas de creación y modificación.



The screenshot shows a web form with a red box highlighting the 'Creado:' and 'Actualizado:' fields. The 'Creado:' field contains '25 de junio de 2021, 2:28 a. M.' and the 'Actualizado:' field also contains '25 de junio de 2021, 2:28 a. M.'.

| | |
|--------------|---------------------------------|
| Turno: | Matutino |
| Creado: | 25 de junio de 2021, 2:28 a. M. |
| Actualizado: | 25 de junio de 2021, 2:28 a. M. |

Borrar **Guardar y agregar otro** **Guardar y continuar editando** **AHORRAR**

RECUERDA QUE TU PROYECTO DE PRUEBA DEBE DE ESTÁR AL CORRIENTE HASTA ESTÉ PUNTO YA QUE SE REQUIERE PARA CONTINUAR CON LOS PROCESOS DE EJEMPLO SIGUIENTES.

TAREA I – SEGURIDAD – TIPOS DE DATOS ORM

- **ACTIVIDAD:**
Lee sobre los tipos de datos aplicables en los modelos Django.
 - Documentación
Oficial: <https://docs.djangoproject.com/en/5.0/ref/models/fields/>
- Realiza en tu cuaderno un resumen sobre los tipos de datos principales (texto, números, booleanos, caracteres, fechar, etc.) y sus restricciones (nulos, en blanco, default, auto incrementales, llaves, etc.). Define de manera clara su sintaxis, significado y uso.