

A graphic on the left side of the slide. It features a 3D effect with four overlapping rectangular blocks in purple, orange, yellow, and blue. The text 'Agencia de Aprendizaje a lo largo de la vida' is written across these blocks in white. An orange arrow points to the right from the orange block.

Agencia de
Aprendizaje
a lo largo
de la vida

DJANGO

Clase 27

Django: Admin - 2

Les damos la bienvenida

Vamos a comenzar a grabar la clase

Clase 27**Django: Admin - 2**

- Modelos muchos a muchos
- Personalizando DjangoAdmin

Clase 28**Django: Autenticación - 1**

- Autenticación vs Autorización
- Configuración autenticación
- Usuarios y Grupos
- Vistas y templates de autenticación

Modelos muchos a muchos

```
class Estudiante(Persona):  
    legajo = models.CharField(max_length=100, verbose_name='Legajo')  
  
    def __str__(self):  
        return f"{self.legajo} - {self.apellido}, {self.nombre}"
```

```
class Comision(models.Model):  
    nombre = models.CharField(max_length=100, verbose_name='Nombre')  
    ...  
    estudiantes = models.ManyToManyField(Estudiante)
```

La clase en la que se decide poner la relación, será la encargada de cargar la misma desde el admin

En ejemplo pig, ver tema traducciones

Modelos muchos a muchos

```
class Estudiante(Persona):  
    legajo = models.CharField(max_length=100, verbose_name='Legajo')  
    def __str__(self):  
        return f"{self.legajo} - {self.apellido}, {self.nombre}"
```

Agregar estudiante

Nombre:	<input type="text" value="D"/>
Apellido:	<input type="text"/>
Legajo:	<input type="text"/>

Modelos muchos a muchos

Agregar comision

Nombre:

Descripcion:

Fecha de inicio: Hoy

Nota: Ud. se encuentra en una zona horaria que está 3 horas atrasada respecto a la del servidor.

Portada: Selecionar archivo Sin archivos seleccionados

Curso:

Docente:

Estudiantes:

123 - Jant, Alejandro
456 - Pablo, Juan
789 - Alberto, Carlos

Mantenga presionada "Control" ("Command" en una Mac) para seleccionar más de uno.

```
class Comision(models.Model):  
    nombre = models.CharField(max_length=100, verbose_name='Nombre')  
    ...  
    estudiantes = models.ManyToManyField(Estudiante)
```

Se puede sobrescribir funcionalidad con el método **formfield_for_manytomany** de **ModelAdmin** y crear otros manejos con la clase **TabularInline** (VER EJEMPLO PIG DE PERSONALIZACIÓN)

Modelos muchos a muchos

```
class Comision(models.Model):  
    nombre = models.CharField(max_length=100, verbose_name='Nombre')  
    ...  
    estudiantes = models.ManyToManyField(Estudiante, through='Inscripcion')
```

Por defecto no aparece mas en comisión, se deben cargar las inscripciones una a una

Agregar comision

Nombre:

Descripcion:

Fecha de inicio: Hoy I

Nota: Ud. se encuentra en una zona horaria que está 3 horas atrasada respecto a la del servidor.

Portada: Seleccionar archivo Sin archivos seleccionados

Curso:

Docente:

Agregar inscripcion

Fecha de creación: Hoy I

Nota: Ud. se encuentra en una zona horaria que está 3 horas atrasada respecto a la del servidor.

Estudiante:

Comision:

Estado: Inscripto

Personalizando el sitio por defecto del DjangoAdmin

```
from django.contrib import admin
from .models import Estudiante, Docente, Curso, Comision, Inscripcion
from .forms import DocenteForm

class EstudianteAdmin(admin.ModelAdmin):
    list_display = ('legajo', 'apellido', 'nombre')
    list_editable = ('apellido', 'nombre')
    list_display_links = ('legajo',)
    search_fields = ['apellido']

admin.site.register(Estudiante, EstudianteAdmin)
```

Sobre escribimos atributos del ModelAdmin

Personalizando el sitio por defecto del DjangoAdmin

```
@admin.register(Comision)
class ComisionAdmin(admin.ModelAdmin):
    list_display = ('nombre', 'fecha_inicio', )

    def formfield_for_manytomany(self, db_field, request, **kwargs):
        if db_field.name == "estudiantes":
            kwargs["queryset"] = Estudiante.objects.filter(legajo__startswith="2").order_by("apellido")
        return super().formfield_for_manytomany(db_field, request, **kwargs)
```

Sobre escribimos métodos del ModelAdmin

Personalizando nuestro DjangoAdmin

```
from django.contrib import admin
from cac.models import Estudiante, Docente, Curso, Comision, Incripcion

class CacAdminSite(admin.AdminSite):
    site_header = "Administración de Codo a Codo"
    site_title = "Administración para super usuarios"
    index_title = "Administrador del Sitio"
    empty_value_display = "No hay nada"

class EstudianteAdmin(admin.ModelAdmin):
    list_display = ('legajo', 'apellido', 'nombre')
    list_display_links = ('nombre', 'apellido', )

sitio_admin = CacAdminSite(name='cacadmin')
sitio_admin.register(Estudiante, EstudianteAdmin)
sitio_admin.register(Docente)
sitio_admin.register(Curso)
sitio_admin.register(Comision)
sitio_admin.register(Incripcion)
```

admin.py

Personalizando nuestro DjangoAdmin

```
from django.urls import path
from django.urls.conf import include
from cac.admin import sitio_admin

urlpatterns = [
    path('cac_admin/', sitio_admin.urls),
    path('', include('cac.urls'))
]
```

urls.py

```
INSTALLED_APPS = [
    # 'django.contrib.admin', # Utiliza el autodiscover
    'django.contrib.admin.apps.SimpleAdminConfig', # No utiliza el autodiscover"
```

settings.py

Utilizando esta configuración no aparecen apps que no agreguemos a mano al admin como auth

**No te olvides de completar la
asistencia y consultar dudas**

Recordá:

- Revisar la Cartelera de Novedades.
- Hacer tus consultas en el Foro.

TODO EN EL AULA VIRTUAL