

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

Reunión 14

Django: Admin - 1

Les damos la bienvenida

Vamos a comenzar a grabar la clase

Reunión 25

Django: Integración de contenidos

- Desarrollo de proyecto integrador grupal con los puntos vistos hasta el momento

Reunión 26


Django: Admin - 1

- Que es el DjangoAdmin
- Configuración del DjangoAdmin
- Creación del superusuario
- Integrando modelos al DjangoAdmin

¿Qué es el Django Admin

Una de las partes más poderosas de Django es la interfaz de administración automática. Lee los metadatos de los modelos para proporcionar una interfaz rápida y centrada en el modelo donde los usuarios de confianza pueden administrar el contenido de su sitio. La herramienta se encuentra enfocada en la administración interna del sistema por parte de un administrador pero **no está pensada para el usuario final. Su intención no es armar un front-end completo de administración.**

Configurable



The image shows a screenshot of the Django admin login interface. It features a dark green header with the word "django" in white. Below the header, there are two input fields: "Username:" with the value "admin" and "Password:". A dark green "Log in" button is positioned to the right of the password field. The entire form is centered on a light gray background.

Configuración inicial necesaria (por defecto con startproject)

INSTALLED_APPS



`django.contrib.admin`



`django.contrib.auth`
`django.contrib.contenttypes`
`django.contrib.messages`
`django.contrib.sessions`

TEMPLATES



```
'BACKEND': 'django.template.backends.django.DjangoTemplates'  
'OPTIONS': {  
    'context_processors': [  
        'django.template.context_processors.request',  
        'django.contrib.auth.context_processors.auth',  
        'django.contrib.messages.context_processors.messages',  
    ],  
},
```

Configuración inicial necesaria (por defecto con startproject)

MIDDLEWARE



```
django.contrib.auth.middleware.AuthenticationMiddleware  
django.contrib.messages.middleware.MessageMiddleware
```

urls.py



```
from django.contrib import admin  
  
urlpatterns = [  
    path('admin/', admin.site.urls),  
    ...  
]
```

Configuración inicial

En la carpeta del proyecto

is_staff en True

```
>> python manage.py createsuperuser  
Username (leave blank to use 'alejandro'): admin  
Email address: alejandro.hunt@bue.edu.ar  
Password:  
Password (again):  
Superuser created successfully.
```

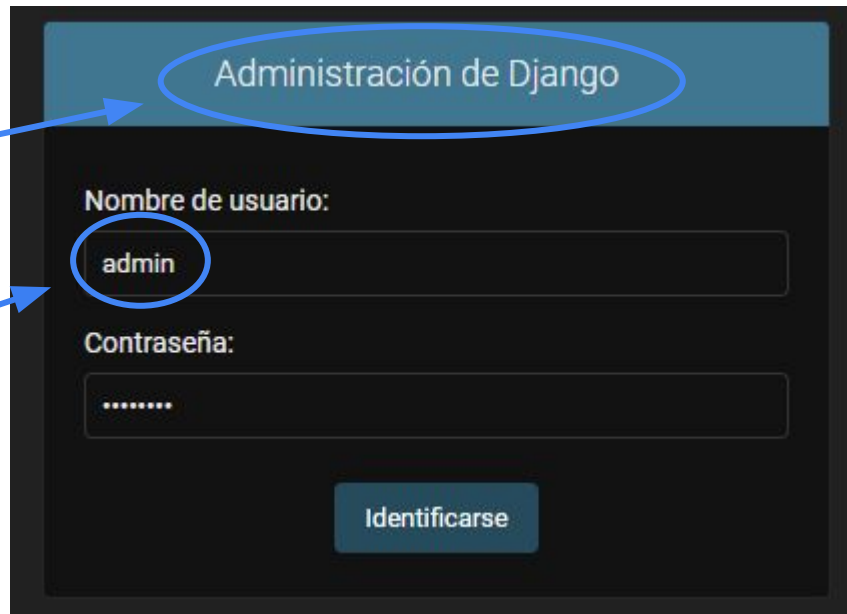
Si tuve perdida de memoria

```
>> python manage.py changepassword admin  
Changing password for user 'admin'  
Password:  
Password (again):  
Password changed successfully for user 'admin'
```


Configuración inicial

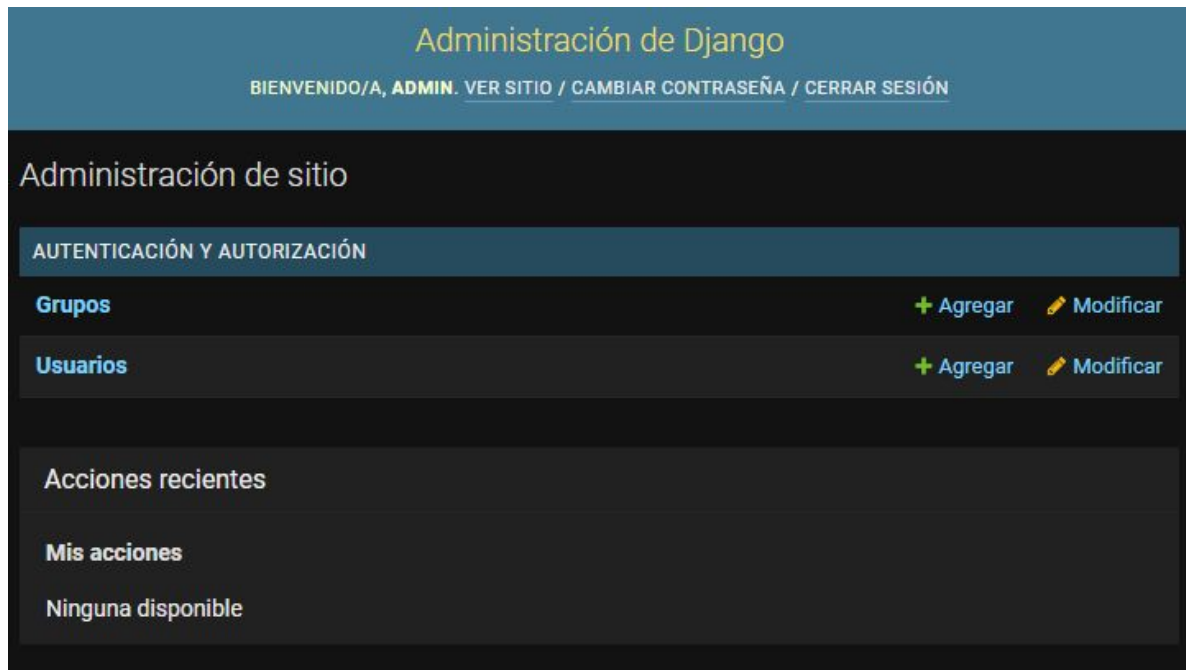
```
LANGUAGE_CODE = 'es-ar'
```

```
is_staff en True
```

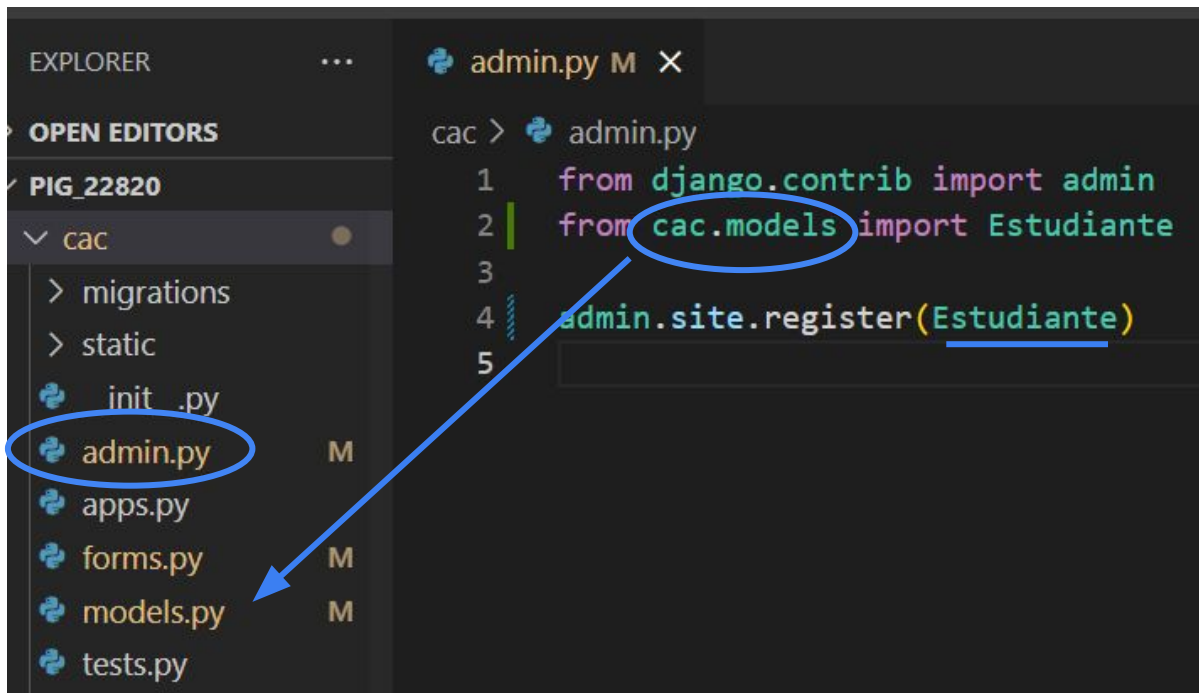


The image shows a screenshot of the Django Admin login interface. The title 'Administración de Django' is circled in blue. Below it, the 'Nombre de usuario:' field contains the text 'admin', which is also circled in blue. The 'Contraseña:' field is shown with masked characters. At the bottom, there is a blue button labeled 'Identificarse'. Two blue arrows point from the text boxes on the left to the 'admin' text in the username field and the 'Identificarse' button.

Y los modelos?



Registro modelos



Registro modelos



Registro modelos

```
class Estudiante(models.Model):  
    nombre = models.CharField(max_length=100, verbose_name='Nombre:')  
    apellido = models.CharField(max_length=150, verbose_name='Apellido:')  
    email = models.EmailField(max_length=150, verbose_name='Email:', null=True, default=None)  
    dni = models.BigIntegerField(verbose_name='DNI:')  
  
    def __str__(self):  
        return f"DNI: {self.dni} - {self.apellido}, {self.nombre}"
```

Administración de Django

Inicio > Cac > Estudiantes

AUTENTICACIÓN Y AUTORIZACIÓN

Grupos + Agregar

Usuarios + Agregar

CAC

Estudiantes + Agregar

Seleccione estudiante a modificar

Acción: Eliminar estudiantes seleccionados/as ▼ Ejecutar 0 de 4 seleccionados/as

- ☐ ESTUDIANTE
- ☒ DNI: 222 - dddddd, asdddd
- ☐ DNI: 2345 - asd, asd
- ☐ DNI: 1234 - Hunt, Ale
- ☐ DNI: 123 - Almada, Thiago

4 estudiantes

Registro modelos

Administración de Django

BIENVENIDO/A, ADMIN. VER SITIO / CAMBIAR CONTRASEÑA

Inicio > Cac > Estudiantes > DNI: 222 - dddddd, asdddd

AUTENTICACIÓN Y AUTORIZACIÓN

Grupos + Agregar

Usuarios + Agregar

CAC

Estudiantes + Agregar

Modificar estudiante

DNI: 222 - dddddd, asdddd HISTORIA

Por favor, corrija el error detallado mas abajo.

Nombre: asdddd

Este campo es obligatorio.

Apellido:

Email: dddd@sd.com

DNI: 222

Eliminar Guardar y agregar otro Guardar y continuar editando GUARDAR

```
class Estudiante(models.Model):
    nombre = models.CharField(max_length=100, verbose_name='Nombre:')
    apellido = models.CharField(max_length=150, verbose_name='Apellido:')
    email = models.EmailField(max_length=150, verbose_name='Email:', null=True, default=None)
    dni = models.BigIntegerField(verbose_name='DNI:')

    def __str__(self):
        return f"DNI: {self.dni} - {self.apellido}, {self.nombre}"
```

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**

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 comisi3n, se deben cargar las inscripciones una a una

Agregar comision

Nombre:

Descripcion:

Fecha de inicio: Hoy !

Nota: Ud. se encuentra en una zona horaria que est1 3 horas atrasada respecto a la del servidor.

Portada: Seleccionar archivo Sin archivos seleccionados

Curso:

Docente:

Agregar inscripcion

Fecha de creaci3n: Hoy !

Nota: Ud. se encuentra en una zona hor

Estudiante:

Comision:

Estado: Inscripto

Personalizando el sitio por defecto del DjangoAdmin

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

Sobrescribimos 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)
```

Sobrescribimos 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