

EXERCISES QUE TRABAJAREMOS EN EL CUE

- EXERCISE 1: ACCIONES ESPECIALES PARA EL ADMINISTRADOR DE DJANGO.
- EXERCISE 2: AJUSTANDO LA PRESENTACIÓN DE LA CABECERA, TÍTULO Y DESCRIPCIÓN DE ADMINISTRADOR.
- EXERCISE 3: APLICANDO FILTROS.
- EXERCISE 4: GESTIÓN DE CAMPOS DINÁMICOS.
- EXERCISE 5: GESTIÓN DE USUARIOS.

EXERCISE 1: ACCIONES ESPECIALES PARA EL ADMINISTRADOR DE DJANGO.

El objetivo del presente ejercicio es plantear una guía paso a paso del panel de administración que contiene funciones básicas, y activar acciones especiales, gestión de filtros, campos dinámicos, usuarios y permisos en el panel de administración del Framework Django.

Requisitos previos:

- Tener conocimiento de un terminal o línea de comando, e instalación de paquetes de software en el sistema operativo tanto de Windows 10 como de Linux (en este caso, Linux Ubuntu que es donde se desarrollará la práctica).
- En caso de Windows, se debe tener instalado el entorno virtual explicado en el Exercises del CUE03, y la configuración del proyecto en el CUE04.
- Tener previamente instalado la versión de Python 3, y el entorno virtual con (virtualenvwrapper).
- Hacer uso de la herramienta Visual Studio Code.

ACCIONES ESPECIALES PARA EL ADMINISTRADOR DE DJANGO

En el administrador de Django, todos los modelos tienen disponible la acción de eliminar, la cual permite seleccionar varias filas de la base de datos y borrarlas.

Aparte de esta acción de eliminar, también podemos crear nuestras propias acciones que agreguen funcionalidad a los elementos en el administrador en la manera en la que deseemos. Por ejemplo:

deseamos modificar ciertos elementos, como establecer un determinado valor a los objetos del tipo Boards con un color determinado.

Para ello debemos crear una función que recibe el ModelAdmin, con los siguientes argumentos: `request` y `queryset`. El `queryset` contendrá todos los objetos que seleccionaremos en la interfaz. Se puede agregar un mensaje que se muestre cuando se ejecuta la acción.

BOARDS/ADMIN.PY

```
1 from django.contrib import admin
2 from django.contrib import messages
3 from .models import BoardsModel
4 class BoardsAdmin(admin.ModelAdmin):
5     readonly_fields = ('creado', 'modificado')
6     list_display = ('clasificacion', 'titulo', 'valor')
7     search_fields = ('titulo', 'descripcion')
8     ordering = ('valor',)
9     def actualizar_valor_a_8(modeladmin, request, queryset):
10         queryset.update(valor=8.0)
11         messages.success(request, "Valor actualizado a 8")
12     admin.site.add_action(actualizar_valor_a_8, "Colocar Valor a 8")
13 admin.site.register(BoardsModel, BoardsAdmin)
```

Al crear el método, se agrega al admin por medio de su método `add_action()`, pasando el método que creamos y el nombre que queremos que aparezca en pantalla al listar el método de esa acción, como primer argumento, y en el segundo respectivamente.

Select tablero to change

Action: 1 of 2 selected

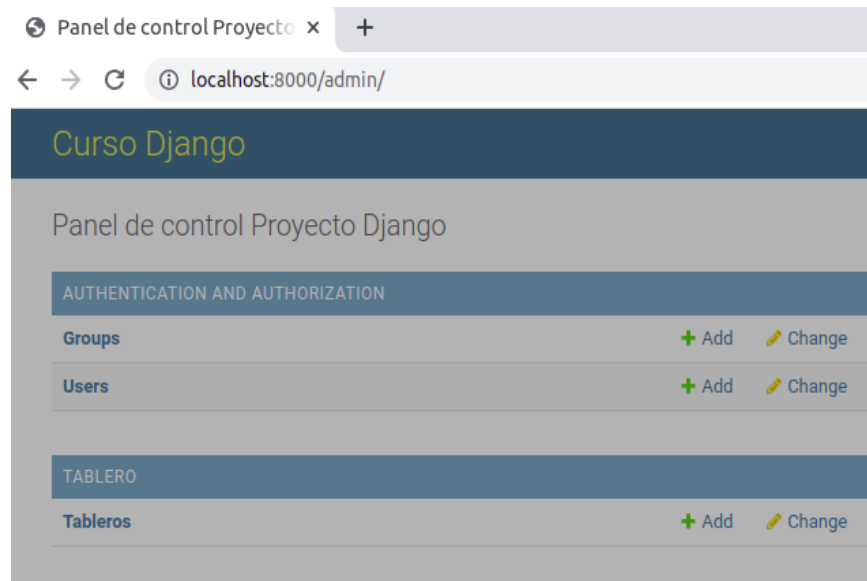
<input type="checkbox"/>	TABLERO
<input checked="" type="checkbox"/>	Descripción 2
<input type="checkbox"/>	Descripción

2 tableros

EXERCISE 2: AJUSTANDO LA PRESENTACIÓN DE LA CABECERA, TÍTULO Y DESCRIPCIÓN DE ADMINISTRADOR.

BOARDS/ADMIN.PY

```
1 from django.contrib import admin
2 # Register your models here.
3 from .models import BoardsModel
4
5 admin.site.site_header = 'Curso Django'
6 admin.site.index_title = 'Panel de control Proyecto Django'
7 admin.site.site_title = 'Administrador Django'
8
9 class BoardsAdmin(admin.ModelAdmin):
10     readonly_fields = ('creado', 'modificado')
11     list_display = ('titulo', 'valor')
12     search_fields = ('titulo', 'descripcion')
13     ordering = ('valor',)
14
15 admin.site.register(BoardsModel, BoardsAdmin)
```



EXERCISE 3: APLICANDO FILTROS.

BOARDS/ADMIN.PY

```
1 from django.contrib import admin
2 # Register your models here.
3 from .models import BoardsModel
4
5 admin.site.site_header = 'Curso Django'
6 admin.site.index_title = 'Panel de control Proyecto Django'
7 admin.site.site_title = 'Administrador Django'
8
9 class BoardsAdmin(admin.ModelAdmin):
10     readonly_fields = ('creado', 'modificado')
11     list_display = ('titulo', 'valor')
12     search_fields = ('titulo', 'descripcion')
13     ordering = ('valor',)
14     list_filter = ('creado', 'valor')
15
16 admin.site.register(BoardsModel, BoardsAdmin)
```

EXERCISE 4: GESTIÓN DE CAMPOS DINÁMICOS.

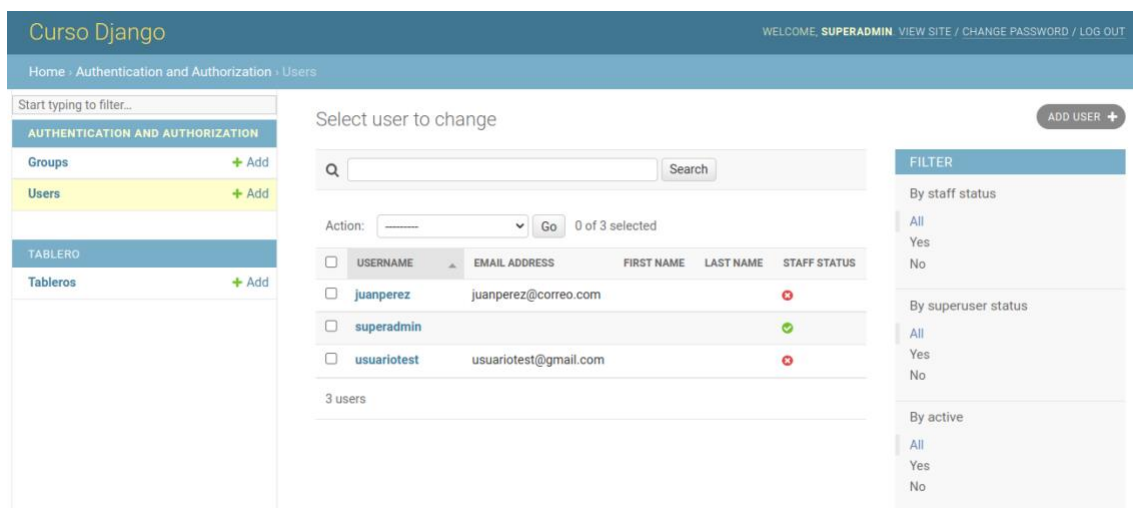
Para crear campos dinámicos personalizados, que no forman parte del modelo, y que se generan de manera dinámica de acuerdo con la información, se deben clasificar los tableros según su valor. En primer lugar, agregar el nombre del campo a `list_display` (para indicarle al admin que lo muestre), y crear un método con el mismo nombre, el cual recibe el objeto individual y debe retornar lo que queremos que se muestre en pantalla.

BOARDS/ADMIN.PY

```
1 class BoardsAdmin(admin.ModelAdmin):
2     readonly_fields = ('creado', 'modificado')
3     list_display = ('clasificacion', 'titulo', 'valor')
4     search_fields = ('titulo', 'descripcion')
5     ordering = ('valor',)
6     list_filter = ('creado', 'valor')
7
8     def clasificacion(self, obj):
9         return "Alto" if obj.valor >= 5 else "Bajo"
10
11 admin.site.register(BoardsModel, BoardsAdmin)
```

EXERCISE 5: GESTIÓN DE USUARIOS.




En Django podemos gestionar la Autenticación y Autorización, para ello vamos a <http://localhost:8000/admin/auth/user/> y observamos:



The screenshot shows the Django Admin interface for managing users. The top navigation bar includes the site title "Curso Django", the user "WELCOME, SUPERADMIN", and links for "VIEW SITE", "CHANGE PASSWORD", and "LOG OUT". The breadcrumb trail indicates the current location: "Home > Authentication and Authorization > Users".

On the left sidebar, the "AUTHENTICATION AND AUTHORIZATION" section is expanded, showing "Groups" and "Users" with "Add" links. The "Users" link is highlighted. Below this, the "TABLERO" section shows "Tableros" with an "Add" link.

The main content area is titled "Select user to change" and features a search bar. Below the search bar, there is an "Action:" dropdown menu and a "Go" button, with a note "0 of 3 selected". A table lists the users:

<input type="checkbox"/>	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
<input type="checkbox"/>	juanperez	juanperez@correo.com			
<input type="checkbox"/>	superadmin				
<input type="checkbox"/>	usuariotest	usuariotest@gmail.com			

Below the table, it says "3 users". On the right side, there is a "FILTER" section with three filter categories: "By staff status", "By superuser status", and "By active". Each category has radio buttons for "All", "Yes", and "No".

Analizamos el usuario:

Curso Django

Home > Authentication and Authorization > Users > juanperez

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

Groups + Add

Users + Add

TABLERD

Tableros + Add

WELCOME, SUPERADMIN - VIEW SITE / CHANGE PASSWORD / LOG OUT

Change user

juanperez

Username: juanperez

Required: 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password: algorithm: pbkdf2_sha256 iterations: 320000 salt: lugaoH***** hash: jo00Rp*****
Raw passwords are not stored, so there is no way to see this user's password, but you can change the password using this form.

Personal info

First name:

Last name:

Email address:

Permissions

☒ Active
Designates whether this user should be treated as active. Unselect this instead of deleting accounts.

☐ Staff status
Designates whether the user can log into this admin site.

☐ Superuser status
Designates that this user has all permissions without explicitly assigning them.

Groups:

Available groups ⌵

Choose all ⌵

Remove all ⌵

Chosen groups ⌵ + Add

The groups this user belongs to. A user will get all permissions granted to each of their groups. Hold down "Control", or "Command" on a Mac, to select more than one.

User permissions:

Available user permissions ⌵

Choose all ⌵

Remove all ⌵

Chosen user permissions ⌵

boards | tablero | Can add boards model

boards | tablero | Can delete boards model

boards | tablero | Can view boards model

Specific permissions for this user. Hold down "Control", or "Command" on a Mac, to select more than one.

Important dates

Last login: Date: 2022-08-07 Today 📅
Time: 22:15:39 Now 🕒
Note: You are 4 hours behind server time.

Date joined: Date: 2022-08-07 Today 📅
Time: 21:42:59 Now 🕒
Note: You are 4 hours behind server time.

Delete

Save and add another

Save and continue editing

SAVE

En el perfil se encuentran varias secciones que se pueden editar, las cuales corresponden a:

- Propiedades del usuario, username y password.
- Información personal: como nombres, apellidos y correo.
- Permisos: activado que se encuentra activo en la plataforma. Staff status, con acceso al sitio de administrador de Django. Superuser status, designado con todos los permisos.
- Grupos.
- Permisos: tanto permisos disponibles para asignar al usuario, como los permisos asignados previamente.
- Fechas de ingreso y de el ultimo inicio de sesión.
- Guardar un usuario.
- Borrar un usuario.

Para agregar un usuario, se procede a ingresar el mismo en el icono **ADD USER**:

Add user

First, enter a username and password. Then, you'll be able to edit more user options.

Username:
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password:
Your password can't be too similar to your other personal information.
Your password must contain at least 8 characters.
Your password can't be a commonly used password.
Your password can't be entirely numeric.

Password confirmation:
Enter the same password as before, for verification.

Verificamos si podemos acceder al panel de admin:

Curso Django

Please enter the correct username and password for a staff account. Note that both fields may be case-sensitive.

Username:

Password:

Log in

Procedemos a agregar el permiso al acceso al sitio Web de Admin de Django como super usuario:

Permissions

☒ Active
Designates whether this user should be treated as active. Unselect this instead of deleting accounts.

☒ Staff status
Designates whether the user can log into this admin site.

☐ Superuser status
Designates that this user has all permissions without explicitly assigning them.

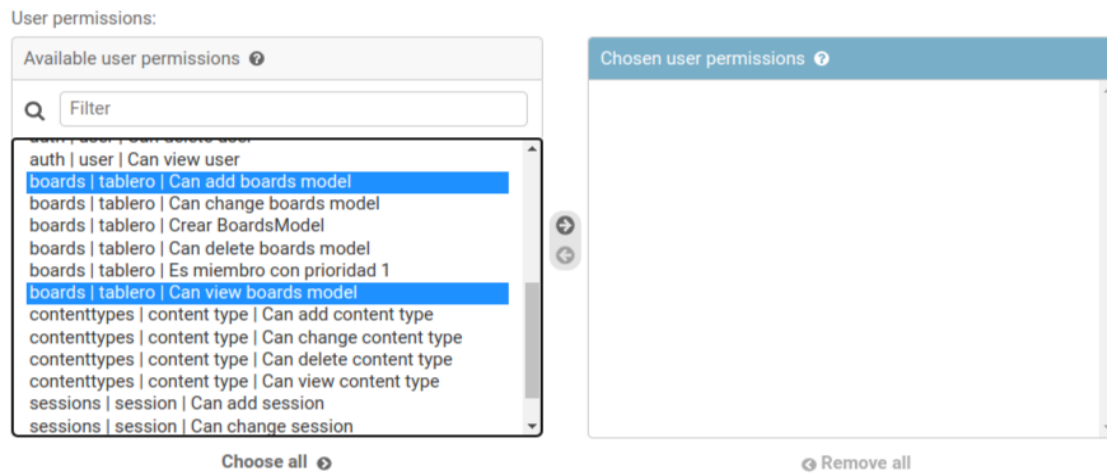
Se vuelve a iniciar sesión con los datos del usuario recientemente creado, observando que no se cuenta con permisos para ver ni editar modelos en el sitio de administración.

Curso Django

Panel de control Proyecto Django

You don't have permission to view or edit anything.

Se debe asignar permisos de añadir y visualizar elementos en el tablero Boards. Para ello se seleccionan los permisos, y se asignan:



Al seleccionarlos y asignarlos, tenemos:

User permissions:

Available user permissions ?

auth | user | Can delete user

auth | user | Can view user

boards | tablero | Can change boards model

boards | tablero | Crear BoardsModel

boards | tablero | Can delete boards model

boards | tablero | Es miembro con prioridad 1

contenttypes | content type | Can add content type

contenttypes | content type | Can change content type

contenttypes | content type | Can delete content type

contenttypes | content type | Can view content type

sessions | session | Can add session

sessions | session | Can change session

sessions | session | Can delete session

sessions | session | Can view session

Choose all ?

Chosen user permissions ?

boards | tablero | Can add boards model

boards | tablero | Can view boards model

Remove all

Guardamos los cambios, e ingresamos nuevamente como el userprueba.

Curso Django

Panel de control Proyecto Django


TABLERO

Tableros

+ Add

View

Ya podemos visualizar y agregar tableros como usuario. Se pueden verificar los accesos creados previamente, es decir, `es_miembro`. También verificamos en <http://localhost:8000/> que nos permite iniciar sesión:

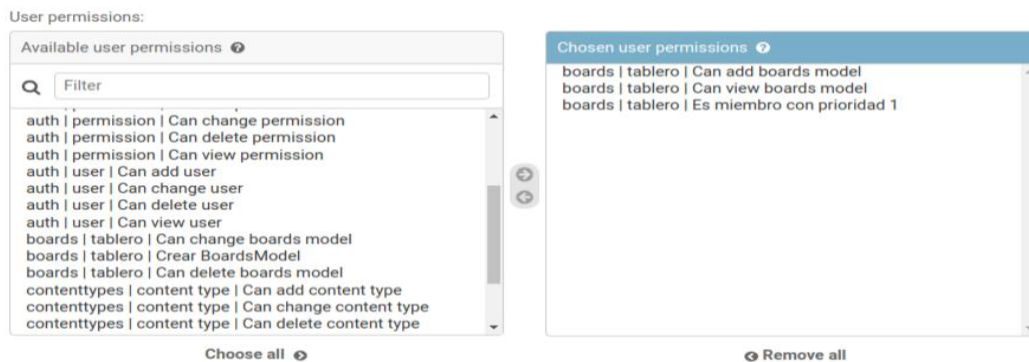


The screenshot shows the Django Admin login interface. At the top, there is a navigation bar with links: 'Navbar', 'Home', 'Link', 'Dropdown', and 'Disabled'. To the right of these links is a search bar with a 'Search' button and a 'Login' link. Below the navigation bar, the main heading is 'Login'. Underneath, there are two input fields: 'Username*' with the value 'superadmin' and 'Password*' with masked characters. A blue 'Login' button is positioned below the password field. At the bottom of the login form, there is a link that says 'Si nio tienes una cuenta, regstrate en [para crear una cuenta.](#)'.

Una vez iniciada, observamos que nos redirige a una pantalla de error:

403 Forbidden

Procedemos a asignar el permiso `es_miembro`.



The screenshot displays the 'User permissions' assignment interface in Django Admin. It is divided into two main panels. The left panel, titled 'Available user permissions', contains a search bar and a list of permissions. The right panel, titled 'Chosen user permissions', shows the permissions that have been assigned to the user. Below the panels are two buttons: 'Choose all' and 'Remove all'.

Available user permissions	Chosen user permissions
auth permission Can change permission	boards tablero Can add boards model
auth permission Can delete permission	boards tablero Can view boards model
auth permission Can view permission	boards tablero Es miembro con prioridad 1
auth user Can add user	
auth user Can change user	
auth user Can delete user	
auth user Can view user	
boards tablero Can change boards model	
boards tablero Crear BoardsModel	
boards tablero Can delete boards model	
contenttypes content type Can add content type	
contenttypes content type Can change content type	
contenttypes content type Can delete content type	

Al ingresar nuevamente como el userprueba, observamos el mensaje, en <http://localhost:8000/>:

Saludos, Hola Mundo

Finalmente, para esta práctica, vamos a ingresar un usuario por el formulario que se creó para ingresar un usuario en <http://localhost:8000/registro>, y que al momento de registrarlo demos acceso al panel de administración, y con los permisos para que pueda agregar y visualizar los objetos del modelo Boards (Tablero) que hemos creado. Editamos el método de registro de la siguiente manera:

BOARDS/VIEWS.PY

```
1 def registro_view(request):
2     if request.method == "POST":
3         form = RegistroUsuarioForm(request.POST)
4         if form.is_valid():
5             user = form.save()
6             content_type = ContentType.objects.get_for_model(BoardsModel)
7
8             es_miembro_1 = Permission.objects.get(
9                 codename='es_miembro_1',
10                 content_type=content_type
11             )
12
13             add_boards = Permission.objects.get(
14                 codename='add_boardsmodel',
15                 content_type=content_type
16             )
17
18             view_boards = Permission.objects.get(
19                 codename='view_boardsmodel',
20                 content_type=content_type
21             )
22
23             # Agregamos el permisos al usuario
24             user.user_permissions.add(es_miembro_1, add_boards,
25 view_boards)
26             user.is_staff = True
27             user.save()
28             login(request, user)
29             messages.success(request, "Registrado Satisfactoriamente.")
30             return HttpResponseRedirect ('/menu')
31             messages.error(request, "Registro invalido. Algunos datos son
32 incorrectos.")
33             form = RegistroUsuarioForm()
```

```
34     return render (request= request,  
35 template_name="registration/registro.html",  
36 context={"register_form":form})
```

Al crear un usuario nuevo, tenemos los accesos al site admin para agregar y visualizar respectivamente los **Boards**.