

Esta clase va a ser

- grabada

a

Clase 20. PYTHON

Playground Intermedio Parte II

Objetivos de la clase



Heredar templates



Procesar contextos

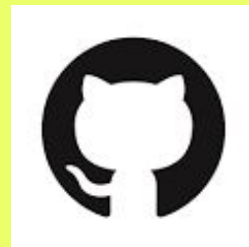


Empezar a administrar

Repositorio Github

Te dejamos el acceso al Repositorio de Github donde encontrarás todo el material complementario y scripts de la clase.

✓ [Repositorio Python](#)



Temario

19

Playground intermedio Parte I

- ✓ Profundizando en MVT
- ✓ Visitas
- ✓ URLs

20

Playground intermedio Parte II

- ✓ [Herencia de Templates](#)
- ✓ [Panel de administración](#)

21

Playground intermedio Parte III

- ✓ Formularios
- ✓ Creación de formularios
- ✓ Búsquedas con Form

Herencia de templates

¿En qué consiste?

Consiste en usar de “base” determinadas cosas que estarán en todos nuestros templates. Suele ser la barra del menú principal, el pie de página, logo, etc. Son cosas que serán idénticas en cada página.

Herencia de templates

Basándonos en donde habíamos llegado antes, podemos definir que los siguientes bloques serán estáticos:

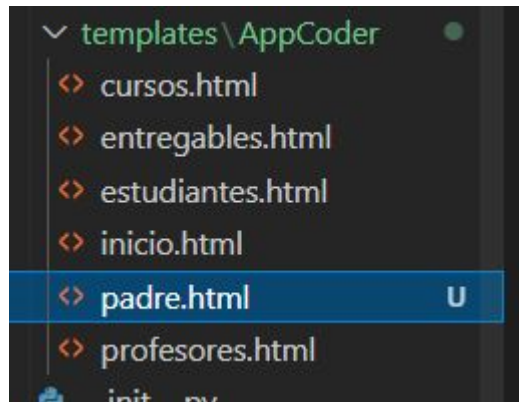


Herencia de templates

Esto se realiza dejando todo lo que no cambiará en un template “padre”, y haciendo que los templates que usan esas partes sean “hijos” que heredan esa estructura. Hacerlo es muy simple:

1

Crearemos un template “padre”, lo llamaremos **padre.html**, y ahí ponemos todo lo que teníamos en inicio.html.



Herencia de templates

Hasta el momento, en la plantilla padre.html solo tenemos una repetición de inicio.html.

2

Dentro de padre.html, creamos un segmento que será el que cambiará según la vista:

```
</div>
</header>

<!-- Esto es lo que va a cambiar -->

{% block contenidoQueCambia %}

{% endblock %}

<!-- Footer -->
<footer class="footer bg-light">
```

Herencia de templates

Es decir, todo será fijo, menos lo que esté dentro de este bloque. Este será el que se genera según la vista.

Notamos que aparece:

```
{% block contenidoQueCambia %}
```

```
.....
```

```
{% endblock %}
```

Herencia de templates

3

Vamos al template que no es base, es decir, a uno de sus hijos y borramos todo. Lo único que debemos dejar es:

```
{% extends "AppCoder/padre.html" %}  
{% load static %}  
{% block contenidoQueCambia %}  
...  
{% endblock %}
```

Herencia de templates

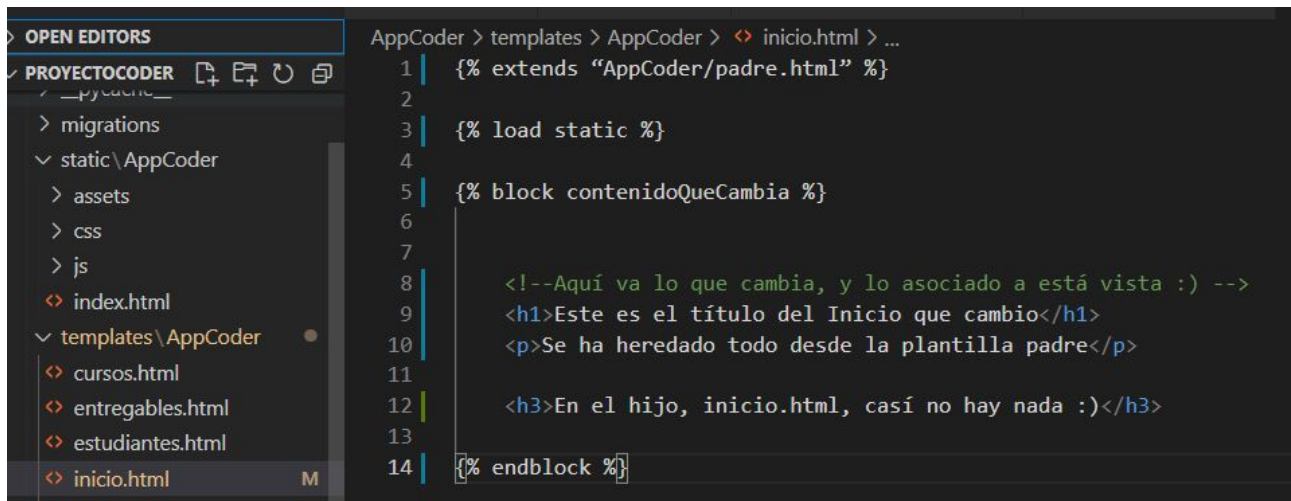
4

Por último, nos resta colocar en los "...", justamente el contenido que será dinámico:

```
<!--Aquí va lo que cambia, y lo asociado a está vista :) -->  
    <h1>Este es el título del Inicio que cambio</h1>  
    <p>Se ha heredado todo desde la plantilla padre</p>  
    <h3>En el hijo, inicio.html, casi no hay nada :)</h3>
```

Herencia de templates

Así nos quedaría:



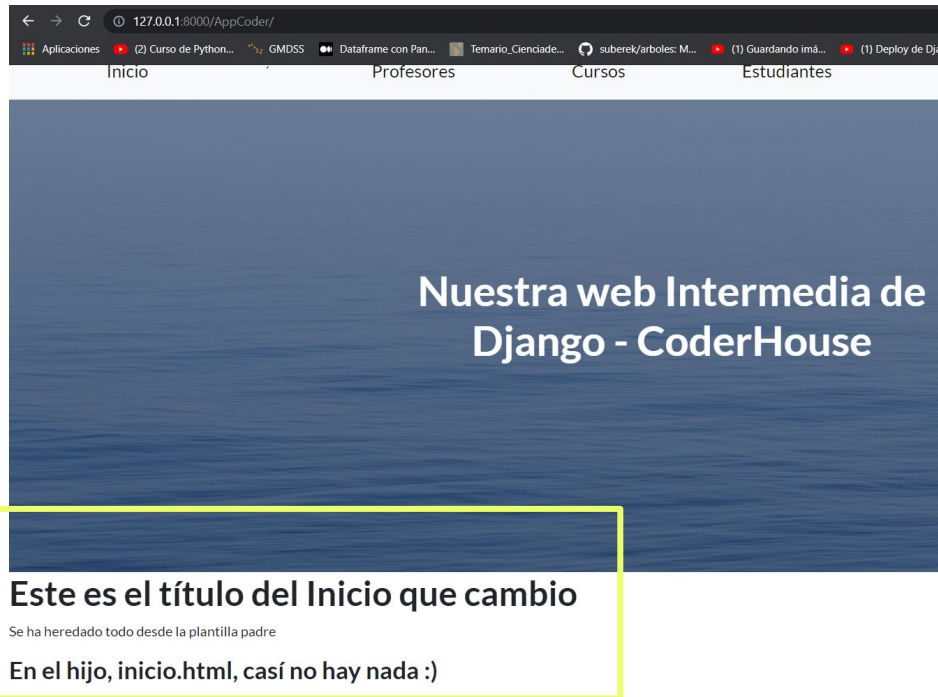
```
AppCoder > templates > AppCoder > inicio.html > ...
1 | {% extends "AppCoder/padre.html" %}
2 |
3 | {% load static %}
4 |
5 | {% block contenidoQueCambia %}
6 |
7 |
8 |     <!--Aquí va lo que cambia, y lo asociado a está vista :) -->
9 |     <h1>Este es el título del Inicio que cambio</h1>
10 |     <p>Se ha heredado todo desde la plantilla padre</p>
11 |
12 |     <h3>En el hijo, inicio.html, casi no hay nada :)</h3>
13 |
14 | {% endblock %}
```

Herencia de templates

Y vemos que el **inicio.html** tiene el aspecto del padre, pero que **agregó información pertinente a dicha plantilla:**

Importante: En el caso de que no lo visualices, recuerda ejecutar nuevamente el siguiente comando:

`python manage.py runserver`



**Navegar entre
nuestros templates**

Navegar entre nuestros templates

¡Excelente! Nuestro inicio se ve muy bien, tiene muy pocas líneas y agrega información a nuestro template padre, pero...



¿Cómo haríamos para ir desde este hijo a otros? ¿Cómo hacemos para que la barra principal funcione?

Navegar entre nuestros templates

En el archivo padre.html reemplazamos el contenido de la etiqueta **nav**, como se muestra a continuación:

```
<nav class="navbar navbar-light bg-light static-top">
  <div class="container">
    <a class="navbar-brand" href="{% url 'Inicio' %}">Inicio</a>
    <a class="navbar-brand" href="{% url 'Profesores' %}">Profesores</a>
    <a class="navbar-brand" href="{% url 'Cursos' %}">Cursos</a>
    <a class="navbar-brand" href="{% url 'Estudiantes' %}">Estudiantes</a>
    <a class="navbar-brand" href="{% url 'Entregables' %}">Entregables</a>
    <a class="btn btn-primary" href="#NADAAUN">INICIAR</a>
  </div>
</nav>
```

Navegar entre nuestros templates

Así debería, quedarnos:

```
</head>
<body>
  <!-- Navigation-->
  <nav class="navbar navbar-light bg-light static-top">
    <div class="container">
      <a class="navbar-brand" href="{% url 'Inicio' %}">Inicio</a>
      <a class="navbar-brand" href="{% url 'Profesores' %}">Profesores</a>
      <a class="navbar-brand" href="{% url 'Cursos' %}">Cursos</a>
      <a class="navbar-brand" href="{% url 'Estudiantes' %}">Estudiantes</a>
      <a class="navbar-brand" href="{% url 'Entregables' %}">Entregables</a>
      <a class="btn btn-primary" href="#NADAAUN">INICIAR</a>
    </div>
  </nav>
```

Navegar entre nuestros templates

Luego hay que modificar todos los HTML y quitar todo el código que no nos sirva. A continuación, mostraremos como realizar este paso para el archivo: cursos.html, pero recuerda que deberás repetir esta operación con el resto de plantillas como ser:

- ✓ entregables.html
- ✓ estudiantes.html
- ✓ profesores.html.

Navegar entre nuestros templates

Script:

```
{% extends "AppCoder/padre.html" %}
```

```
{% load static %}
```

```
{% block contenidoQueCambia %}
```

```
    <!--Aquí va lo que cambia, y lo asociado a esta vista  
:) -->
```

```
    <h1>Cursos</h1>
```

```
    <p>Acá va la info de los cursos</p>
```

```
{% endblock %}
```

Navegar entre nuestros templates

cursos.html:

```
<> padre.html U  urls.py U  admin.py M  settings.py M  <> cursos.html U X
AppCoder > templates > AppCoder > <> cursos.html > ...
1  {% extends "AppCoder/padre.html" %}
2
3  {% load static %}
4
5  {% block contenidoQueCambia %}
6
7
8      <!--Aquí va lo que cambia, y lo asociado a está vista :) -->
9      <h1>Cursos</h1>
10     <p>Acá va la info de los cursos</p>
11
12
13
14 {% endblock %}
15
```

Navegar entre nuestros templates

Por ultimo, modificamos el archivo: urls.py

```
from django.urls import path
```

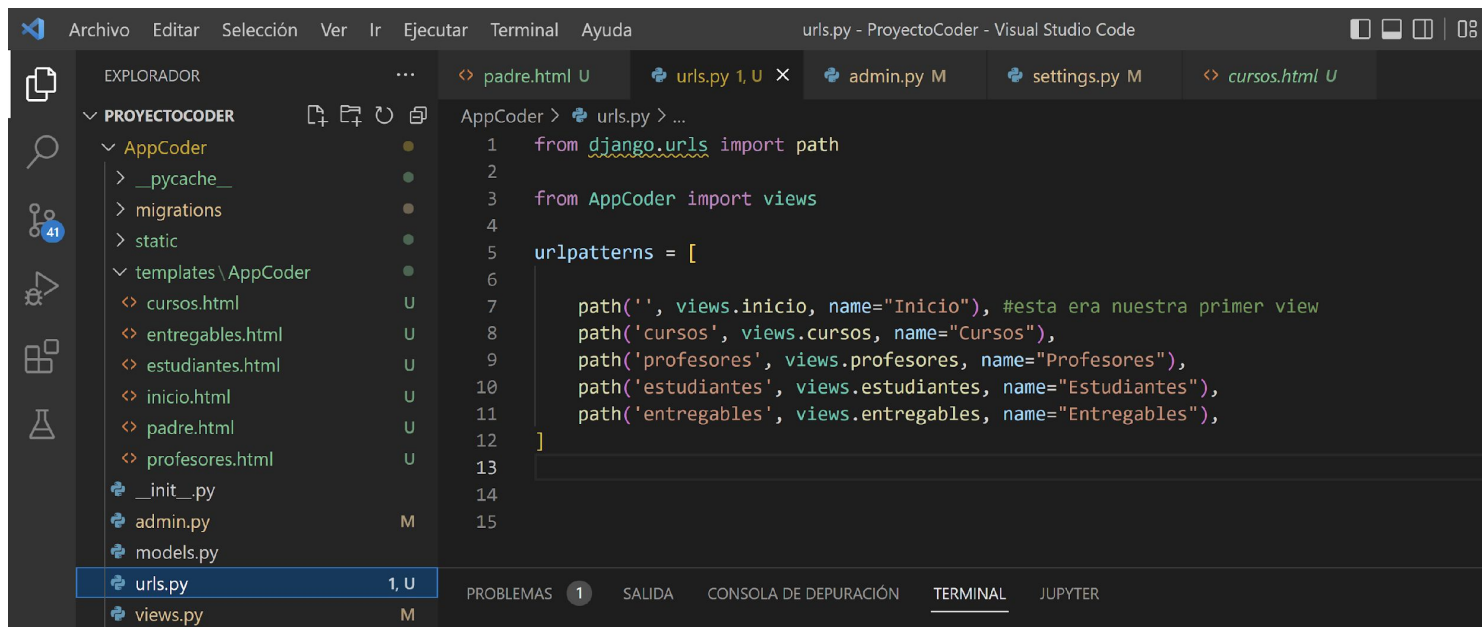
```
from AppCoder import views
```

```
urlpatterns = [
```

```
    path("", views.inicio, name="Inicio"), #este era nuestro primer view
    path('cursos', views.cursos, name="Cursos"),
    path('profesores', views.profesores, name="Profesores"),
    path('estudiantes', views.estudiantes, name="Estudiantes"),
    path('entregables', views.entregables, name="Entregables"),
```

```
]
```

Navegar entre nuestros templates



Navegar entre nuestros templates



¡Listo! 🍳 De esta forma ya tenemos el menú que se hereda por todos los templates y que nos permite movernos de uno a otro.

Es importante ser consistentes con los nombres de los templates, de las urls y de los archivos .html



Herencia y Navegación

Completar la herencia múltiples del proyecto AppCoder

Duración: 20 minutos



ACTIVIDAD EN CLASE

Herencia y Navegación

Terminar de configurar y heredar el resto de los templates de nuestro proyecto AppCoder:

- entregables.html
- estudiantes.html
- profesores.html.

El contenido del HTML, de cada archivo queda a criterio del alumno, seamos creativos 😊



Break

¡10 minutos y volvemos!

Panel de administración

Panel de administración

Por ahora sabemos como guardar campos en una base de datos desde la consola, y hemos visto cómo hacerlo (muy a mano) desde un view. Pero por lo general, la información tiene que ser administrada por la misma web.

Esto suele ser una tarea complicada, pero por fortuna, Django nos simplifica el manejo de la información por medio de un panel de administración ya preparado.

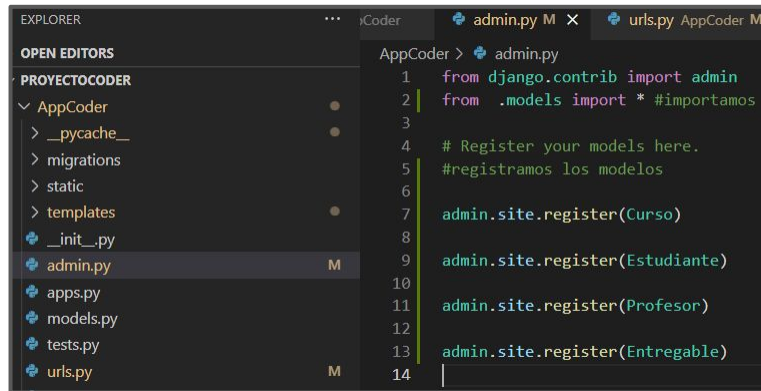
Dar de alta a nuestro panel

¿Cómo dar de alta a nuestro panel de administración?

1

Primero importar los modelos que queremos administrar y registrarlos en el archivo `admin.py` de la app.

Importante: Si el archivo no existe, deberemos crearlo.



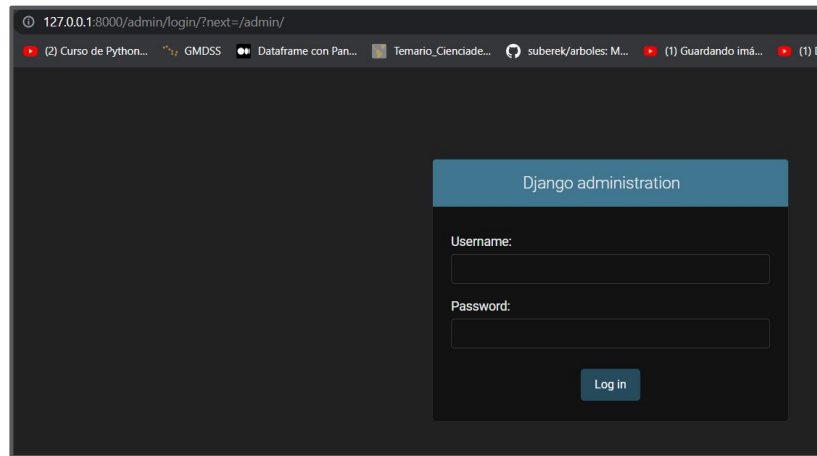
The screenshot shows a code editor with a dark theme. On the left, the 'EXPLORER' sidebar shows a project named 'PROYECTOCODER' with a folder 'AppCoder'. Inside 'AppCoder', several files are listed: '__pycache__', 'migrations', 'static', 'templates', '__init__.py', 'admin.py' (highlighted with a mouse cursor), 'apps.py', 'models.py', 'tests.py', and 'urls.py'. The 'admin.py' file is open in the main editor area. The code in 'admin.py' is as follows:

```
1 from django.contrib import admin
2 from .models import * #importamos
3
4 # Register your models here.
5 #registramos los modelos
6
7 admin.site.register(Curso)
8
9 admin.site.register(Estudiente)
10
11 admin.site.register(Profesor)
12
13 admin.site.register(Entregable)
14
```


Dar de alta a nuestro panel

2

Notar que si entramos a web/admin/
Nos pide un usuario y contraseña.



Dar de alta a nuestro panel

3

Entonces, hay que crear el usuario para administrar, lo que se llama un super usuario:

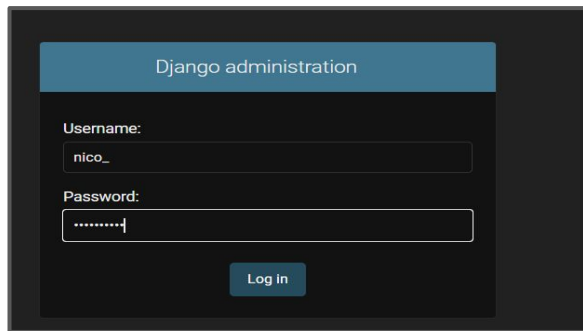
```
python manage.py createsuperuser
```

```
PS C:\Users\nico\Desktop\CarpetaGitHub\ProyectoCoder> python manage.py createsuperuser
Username (leave blank to use 'nico_'):
Email address: nico_perez_velez@hotmail.com
Password:
Password (again):
Superuser created successfully.
```

Dar de alta a nuestro panel

4

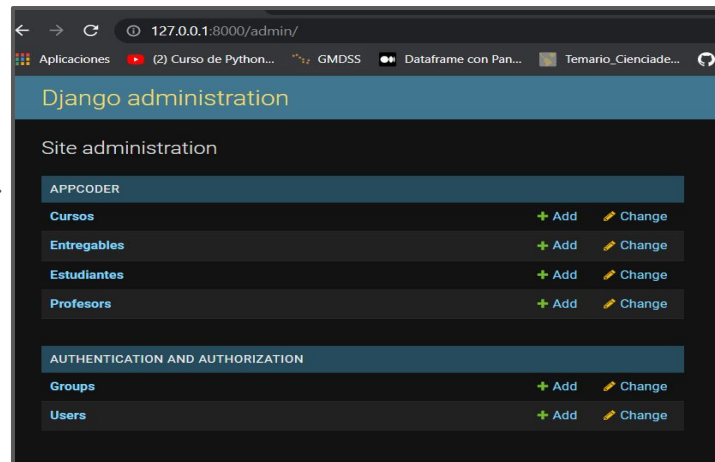
Ingresamos a /admin/ y probamos el super usuario:



Django administration

Username:

Password:



127.0.0.1:8000/admin/

Aplicaciones (2) Curso de Python... GMDSS Dataframe con Pan... Temario_Cienciade...

Django administration

Site administration

APPCODER	
Cursos	+ Add Change
Entregables	+ Add Change
Estudiantes	+ Add Change
Profesors	+ Add Change
AUTHENTICATION AND AUTHORIZATION	
Groups	+ Add Change
Users	+ Add Change

Dar de alta a nuestro panel

5

Si el **Site**, no registra la app, deberemos copiar y reemplazar el código propuesto en el siguiente archivo, en **settings.py**.

Código de registro.txt

Recuerden guardar todos los cambios y lanzar nuevamente el servidor!.

```
python manage.py runserver
```

Dar de alta a nuestro panel

¡Increíble, ya tenemos acceso a nuestro panel de administración y a todas las clases del modelo de nuestra web!





6

Obviamente, seguramente no solo habrá un super usuario, sino también otros usuarios con menos permisos, podemos crear algunos muy simplemente.

Creación de usuarios

Users

 Add  Change

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.

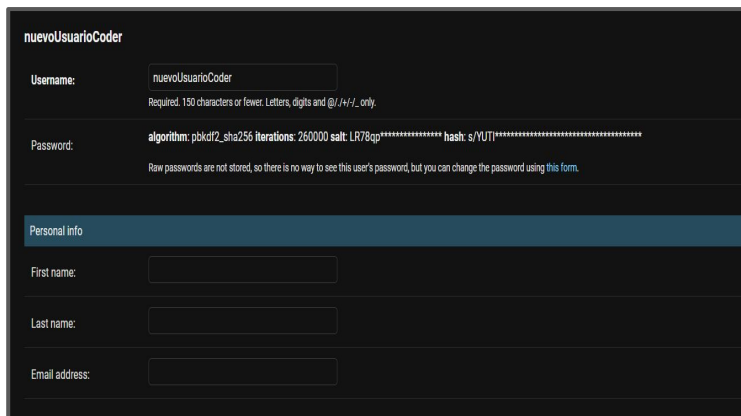
Save and add another

Save and continue editing

SAVE

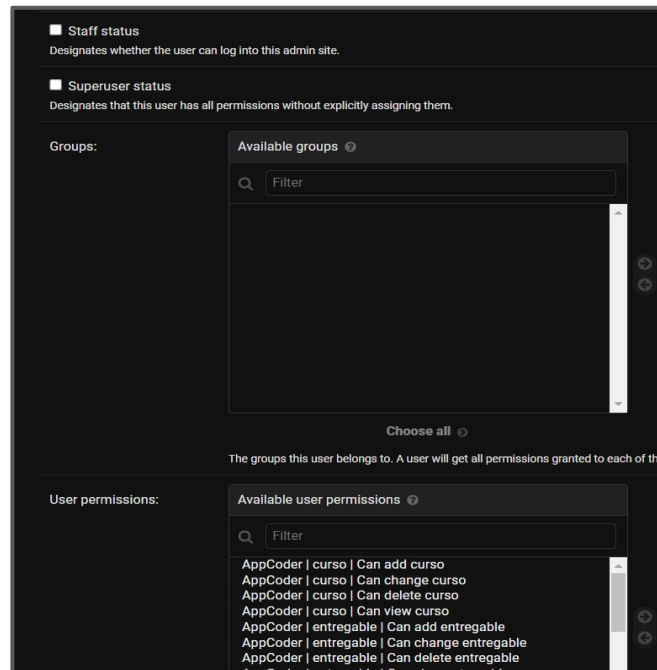
Creación de usuarios

Pueden llenar otros campos no obligatorios:



The screenshot shows a form titled 'nuevoUsuarioCoder'. It has three main sections: 'Username', 'Password', and 'Personal info'. The 'Username' field contains 'nuevoUsuarioCoder' with a note: 'Required, 150 characters or fewer. Letters, digits and @/./+/-/_ only.' The 'Password' field shows technical details: 'algorithm: pbkdf2_sha256 iterations: 260000 salt: LR78qp***** hash: s/YUT*****' and a note: '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.' The 'Personal info' section has fields for 'First name', 'Last name', and 'Email address', all of which are currently empty.

Y establecer permisos que tendrá este usuario (por ahora no nos interesa):



The screenshot shows a configuration interface for a user. It includes several sections: 'Staff status' with a checkbox and description; 'Superuser status' with a checkbox and description; 'Groups' with a search bar and a list of available groups; and 'User permissions' with a search bar and a list of available permissions. The 'Groups' section has a 'Choose all' button. The 'User permissions' section shows a list of permissions including 'AppCoder | curso | Can add curso', 'AppCoder | curso | Can change curso', 'AppCoder | curso | Can delete curso', 'AppCoder | curso | Can view curso', 'AppCoder | entregable | Can add entregable', 'AppCoder | entregable | Can change entregable', and 'AppCoder | entregable | Can delete entregable'.

Creación de usuarios

Le damos **SAVE** y vemos que ya tenemos dos usuarios para usar el panel de administración, solo que el último no tiene permisos para nada:

Action:

Go

 0 of 2 selected

<input type="checkbox"/>	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
<input type="checkbox"/>	nico_	nico_perez_velez@hotmail.com			✓
<input type="checkbox"/>	nuevoUsuarioCoder				✗







Ejemplo en vivo

Ahora veremos cómo agregar instancias de alguna clase desde el admin.

Duración: **10 minutos**

Paso a paso

1


APPCODER		
Cursos	+ Add	 Change
Entregables	+ Add	 Change
Estudiantes	+ Add	 Change
Profesors	+ Add	 Change

Paso a paso

2

Add curso

Nombre:

Camada: 

Paso a paso

3

Select curso to change

Action: 0 of 6 selected

- ☐ CURSO
- ☐ Curso object (6)
- ☐ Curso object (5)
- ☐ Curso object (4)
- ☐ Curso object (3)
- ☐ Curso object (2)
- ☐ Curso object (1)

6 cursors

Paso a paso

4

Curso object (6)

Nombre:	<input type="text" value="Programación Python"/>
Camada:	<input type="text" value="12345"/>

Podemos ver todos los cursos creados, el de arriba es el último en crearse, vemos también que en éste caso, habían otros 6 creados (eso *cambiará*).



Agregar datos a nuestro model

Por medio del panel de administración cargar dos modelos distintos con datos.

Duración: **15 minutos**



ACTIVIDAD EN CLASE

Agregar datos a nuestro model

Descripción de la actividad.

Crear un superusuario para poder ingresar al panel de administración.

Una vez ingresados, agregar datos a dos modelos distintos (botón add).



Tercera pre-entrega

En la clase que viene se presentará la consigna de la primera parte del Proyecto final, que **nuclea temas vistos entre las clases 16 y 21**.

Recuerda que tendrás 7 días para subirla en la plataforma.

¿Preguntas?

Resumen de la clase hoy

- ✓ Heredar contextos entre HTML
- ✓ Navegar entre distintos .html
- ✓ Ingresar al panel de admin
- ✓ Cargar datos desde el panel

Muchas gracias.

#DemocratizandoLaEducación