Esta clase va a ser

- grabada

Clase 19. PYTHON

Playground intermedio Parte I

CODERHOUSE

Temario

18	19	20
Portfolio Parte II	**Playground intermedio Parte I**	**Playground intermedio Parte II**
✓ Mejoras en las plantillas ✓ Modelo ✓ ¿Cómo crear una app?	✓ Profundizando en MVT ✓ Visitas ✓ URLs	✓ Herencia de Templetes ✓ Panel de administración

CODERHOUSE

Objetivos de la clase

- **Profundizar** conceptos de MTV.
- **Ejecutar** cambios en nuestro Git.
- **Gestionar** las versiones en GitHub.
- **Crear** URLs avanzadas.

Repositorio Github

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

✓ Repositorio Python

CODERHOUSE

Profundizando
MVT

¿De dónde partimos?

CODERHOUSE

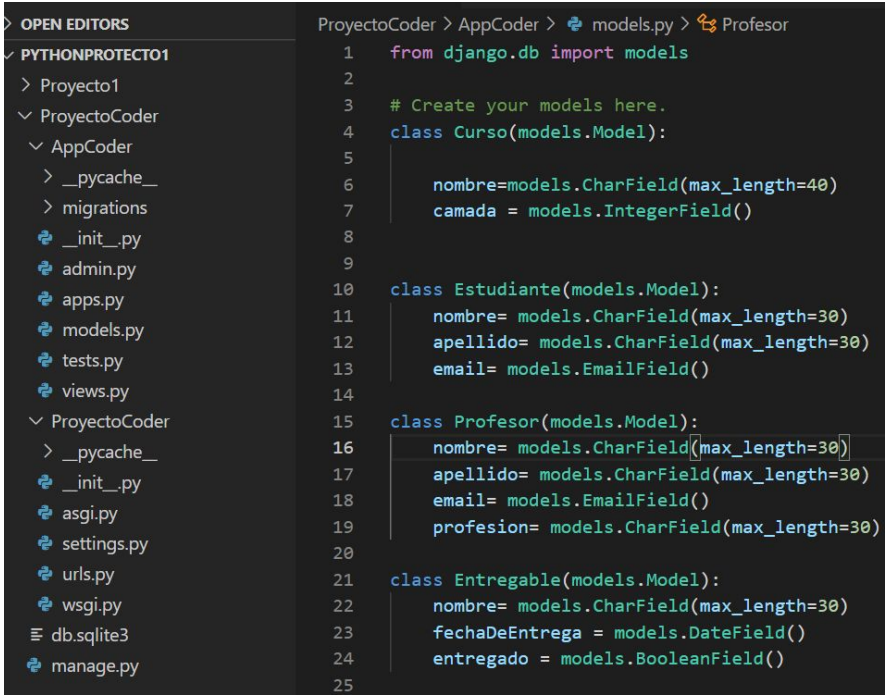
¿De dónde partimos?

Vamos a crear un Proyecto y su respectiva App pero profundizando más en cada capa del patrón MTV. Partiremos de donde dejamos, con la **AppCoder**, con un modelo simple pero bastante rico:

- ✓ **Estudiantes** (nombre, apellido, email)
- ✓ **Profesor** (nombre, apellido, email, profesión)
- ✓ **Entregable** (nombre, fechaDeEntrega, entregado)
- ✓ **Curso** (nombre, comisión)

¿De dónde partimos?

Recordemos que por ahora tenemos algo así 👉



CODERHOUSE

Controlar versiones

CODERHOUSE

¿De dónde partimos?

Django nos da ganas de seguir indagando. Pero, llega un momento que uno empieza a tener miedo de agregar cosas y “romper” todo el código fuente.

CODERHOUSE

¿De dónde partimos?

Es por eso que cuando uno llega a un **punto seguro** quiere guardar esos cambios y **poder volver a este punto si algo sale mal.**
Además, si estamos orgullosos de nuestro avance y queremos compartirlo para que alguien lo siga o lo vea, también es útil tenerlo disponible.

Veamos la correcta forma de trabajarlo. 👉

Paso a Paso con Git

CODERHOUSE

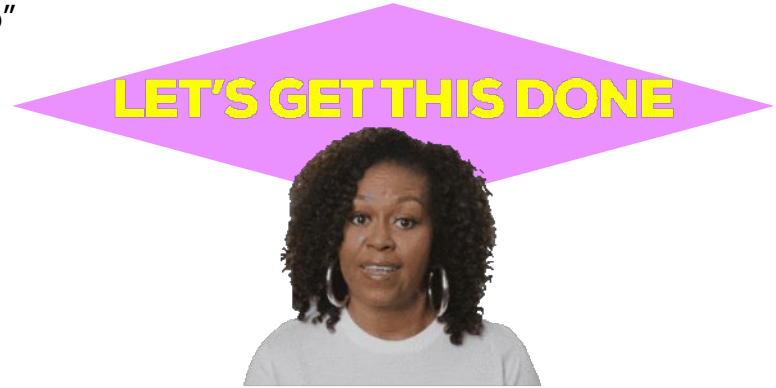
¿Cómo Controlar versiones compartiendo con Github?

1. Entrar a www.github.com e ingresar con tu usuario y pass
2. Crear un nuevo repositorio (Mismo nombre del proyecto en lo posible)
3. Copiar el acceso a nuestro Repositorio Online
4. Clonar el repositorio
5. Ingresar al repositorio
6. Poner nuestro proyecto en el repositorio

👉

¿Cómo Controlar versiones compartiendo con Github?

7. Ver que está todo listo con un git status
8. Cambiar de estado con git add.
9. Guardar la versión con git commit -m “Comentario”
10. Enviar a la web con git push origin master
11. Checkear que se subió a la web

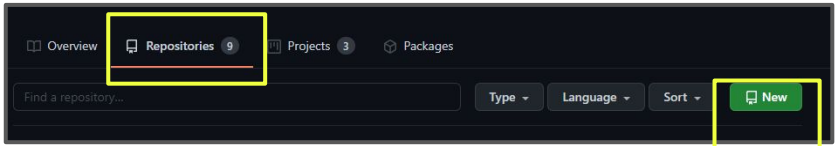


CODERHOUSE

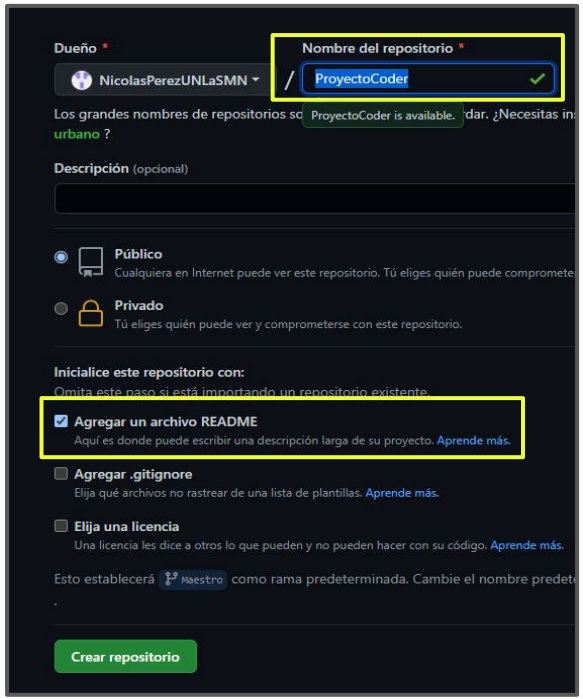
Ejemplo en vivo

Veamos el paso a paso de cómo controlar versiones compartiendo con Github.

Controlar versiones



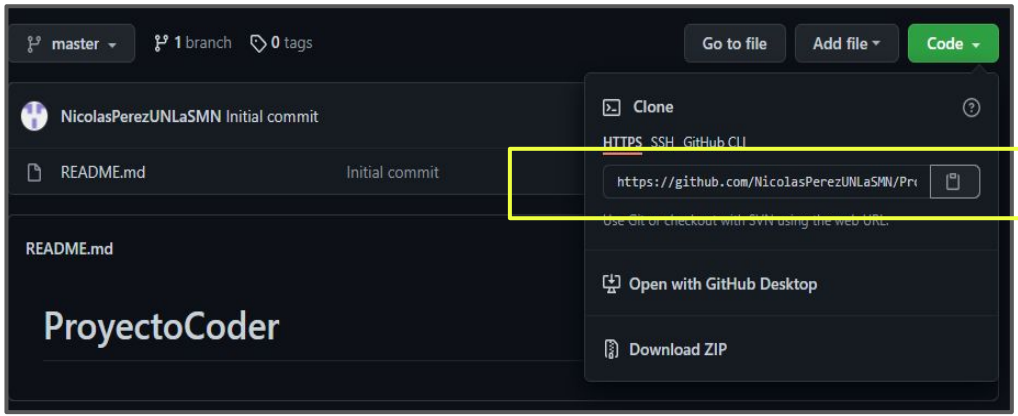
1. Entrar a www.github.com e ingresar con tu usuario y pass.



2. Crear un nuevo repositorio (Mismo nombre del proyecto en lo posible).

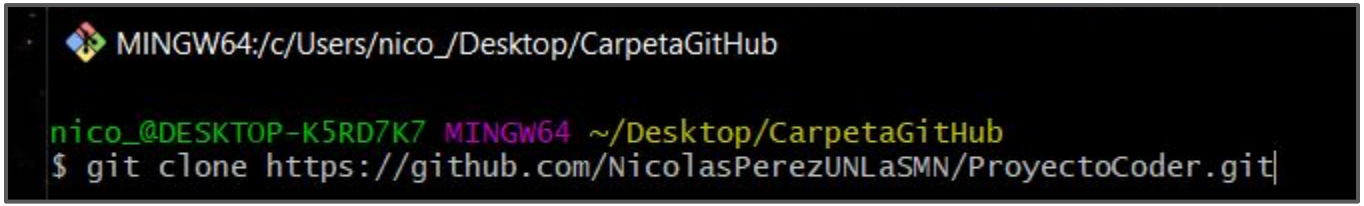
CODERHOUSE

Controlar versiones



3. Copiar el acceso a nuestro Repositorio Online

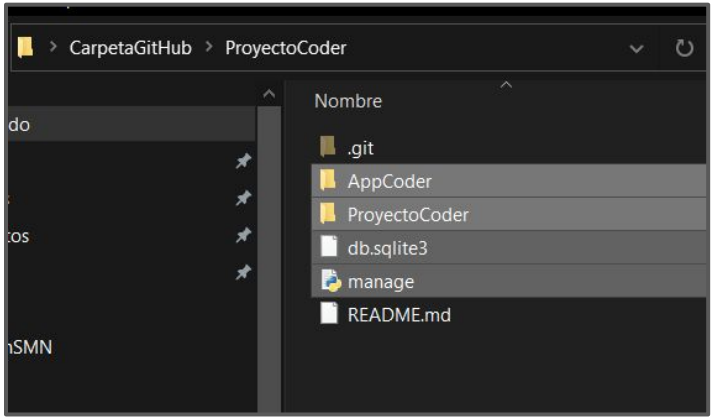
4. Clonar el repositorio



CODERHOUSE

Controlar versiones

```
nico_@DESKTOP-K5RD7K7 MINGW64 ~/Desktop/CarpetaGitHub
$ cd ProyectoCoder

nico_@DESKTOP-K5RD7K7 MINGW64 ~/Desktop/CarpetaGitHub/ProyectoCoder (master)
$
```



5. Ingresar al repositorio

6. Poner nuestro proyecto en el repositorio

CODERHOUSE

Controlar versiones

```
nico_@DESKTOP-K5RD7K7 MINGW64 ~/Desktop/CarpetaGitHub/ProyectoCoder (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        AppCoder/
        ProyectoCoder/
        db.sqlite3
        manage.py
```

7. Ver que está todo listo con un git status.

```
nico_@DESKTOP-K5RD7K7 MINGW64 ~/Desktop/CarpetaGitHub/ProyectoCoder (master)
e$ git add .
```

8. Cambiar de estado con git add.

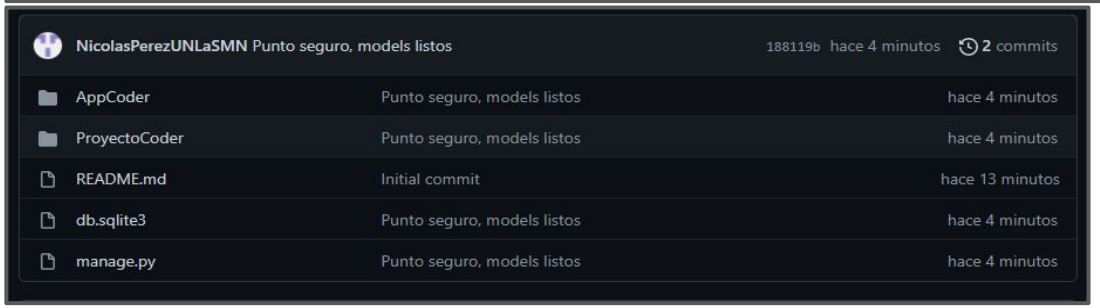
```
nico_@DESKTOP-K5RD7K7 MINGW64 ~/Desktop/CarpetaGitHub/ProyectoCoder (master)
$ git commit -m "Punto seguro, models listos"
[master 188119b] Punto seguro, models listos
 26 files changed, 303 insertions(+)
 create mode 100644 AppCoder/__init__.py
 create mode 100644 AppCoder/__pycache__/__init__.cpython-39.pyc
 create mode 100644 AppCoder/__pycache__/admin.cpython-39.pyc
 create mode 100644 AppCoder/__pycache__/apps.cpython-39.pyc
 create mode 100644 AppCoder/__pycache__/models.cpython-39.pyc
```

9. Guardar la versión con git commit -m “Comentario”

Controlar versiones

```
nico_@DESKTOP-K5RD7K7 MINGW64 ~/Desktop/CarpetaGitHub/ProyectoCoder (master)
$ git push origin master
Enumerating objects: 33, done.
Counting objects: 100% (33/33), done.
Delta compression using up to 4 threads
Compressing objects: 100% (31/31), done.
Writing objects: 100% (32/32), 14.86 KiB | 1.86 MiB/s, done.
Total 32 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/NicolasPerezUNLaSMN/ProyectoCoder.git
   fe14eec..188119b  master -> master
```

10. Enviar a la web con git push origin master.



11. Chequear que se subió a la web

CODERHOUSE

Fin del proceso

¡Listo! Además de tener una versión segura en tu ordenador, a la que podrás volver siempre en caso de que algo salga mal, cuentas con su versión en GitHub para que cualquiera pueda trabajar sobre ella.

De ahora en más, cada vez que realices cambios y llegues a un punto seguro, deberás repetir los pasos del 7 en adelante (aunque 9 y 10 serían solo para cuando quieras que esté disponible online también). 😉

Mi Django a Github

Enviarás una versión de tu proyecto a GitHub.

Duración: **15 minutos**

CODERHOUSE

ACTIVIDAD EN CLASE

Mi Django a Github

Trabajarás sobre un Proyecto Web de Django que ya tengas funcionando y lo subirás a un Repositorio de GitHub, crear dicho repositorio en caso de no tenerlo.

Break

¡10 minutos y volvemos!

Visitas y URLs
(URLs avanzadas)

Visitas

CODERHOUSE

Ejemplo en vivo

Veamos cómo crear visitas

Creando visitas

Estamos tratando de **estructurar nuestro proyecto para incorporar nuevos conceptos.**
Iniciemos **creando algunas vistas** asociadas a nuestro modelo:

```python
def inicio(request):

    return HttpResponse('vista inicio')

def cursos(request):

    return HttpResponse('vista cursos')

def profesores(request):

    return HttpResponse('vista profesores')


def estudiantes(request):

    return HttpResponse('vista estudiantes')


def entregables(request):

    return HttpResponse('vista entregables')
```

URLs

Organicemos nuestras URLs

Un proyecto puede tener muchas App (nosotros tenemos solo una), pero pensando en algo más general, acomodemos un poco las urls, para mejorar la reutilización.

Para pensar

¿Qué pasaría si nuestro archivo urls.py tuviera que dirigir a MUCHAS APPS? ¿Qué alternativa se te ocurre?

Contesta mediante el chat de Zoom

Generar archivo URL. PY en nuestra app

CODERHOUSE

Paso a paso

1. Crear en AppCoder urls.py
2. Le importamos el path: **from django.urls import path**
3. Importamos las vistas **from AppCoder import views**
4. Copiamos y pegamos el urlpatterns que ya teníamos, pero sin el admin
5. Dejamos el admin, SOLO, en el url del Proyecto (tampoco necesita las vistas)
6. Lo más importante, relacionamos el urls.py de la App con el Proyecto:
7. **path('AppCoder/', include('AppCoder.urls'))**

Ejemplo en vivo

Veamos cómo generar URL.PY

URLs

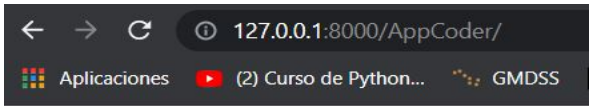
URLs del Proyecto

```python
from django.contrib import admin
from django.urls import path, include
#from AppCoder.views import * #Ya no seria necesario :)


urlpatterns = [
    path('admin/', admin.site.urls),

    path('AppCoder/', include('AppCoder.urls')),

]
```

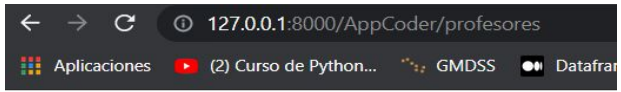
URLs de la App

```python
from django.urls import path

from AppCoder import views

urlpatterns = [

    path('', views.inicio), #esta era nuestra primer view
    path('cursos', views.cursos, name="Cursos"),
    path('profesores', views.profesores),
    path('estudiantes', views.estudiantes),
    path('entregables', views.entregables),
]
```

Todo está bien 🙂



vista inicio



vista profesores

CODER HOUSE

CODERHOUSE

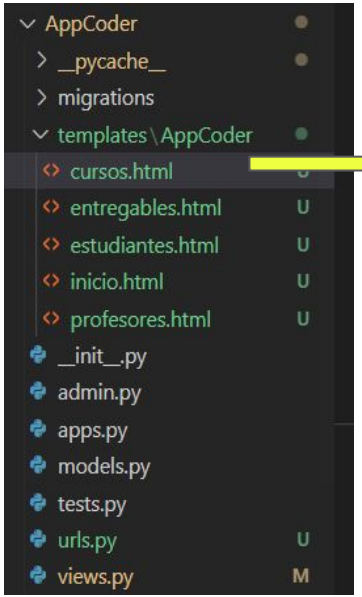
Creando nuestros Templates

CODERHOUSE

Ejemplo en vivo

Veamos cómo pulir nuestro Template.

Puliendo estructura y definiciones



1- Vamos a la App y creamos una carpeta templates, y dentro
de ella una subcarpeta que se llame AppCoder

AppCoder > templates > AppCoder > cursos.html > html

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Cursos</title>
8  </head>
9  <body style="background-color: blue;">
10
11 </body>
12 </html>
```

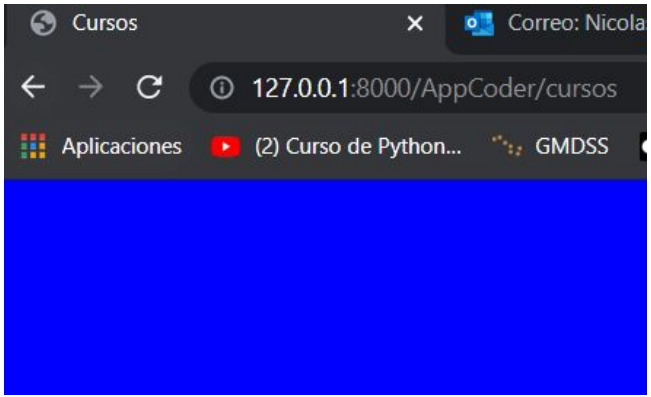
2- Dentro de esta última creamos los html. Uno por vista.

CODERHOUSE

Puliendo estructura y definiciones

```python
def inicio(request):

    return render(request, "AppCoder/inicio.html")

def cursos(request):

    return render(request, "AppCoder/cursos.html")

def profesores(request):

    return render(request, "AppCoder/profesores.html")


def estudiantes(request):

    return render(request, "AppCoder/estudiantes.html")


def entregables(request):

    return render(request, "AppCoder/entregables.html")
```

3- Hacemos que las vistas hagan Render de los templates:



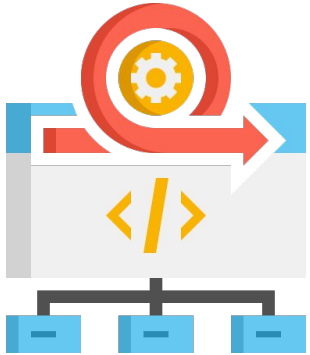
CODERHOUSE

Mejorando nuestros Templates

CODERHOUSE

Mejorando nuestros Templates

Si bien no ahondaremos sobre desarrollo web, necesitamos archivos .**html**. Para simplificar ésto usaremos otro Framework **(como Django).** Esta nueva herramienta nos ayudará a que nuestra web sea más “linda” sin saber mucho.



Mejorando nuestros Templates

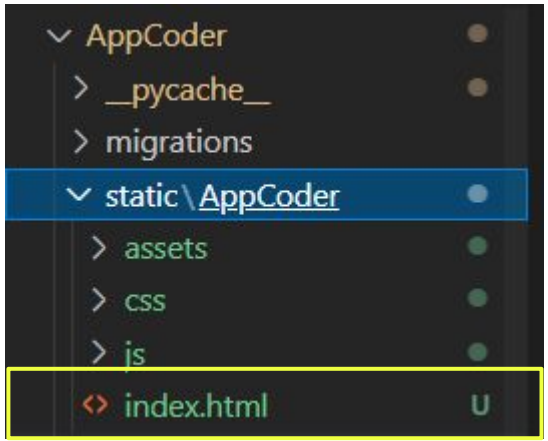
1- Haremos templates con Bootstrap. Vamos a https://getbootstrap.com/ para saber más, y nos descargamos algún esqueleto cualquiera.

2- En nuestros ejemplos usaré: https://startbootstrap.com/previews/landing-page

Ejemplo en vivo

Seguimos puliendo nuestro Template.

Mejorando nuestros Templates



3- Creamos una carpeta en nuestra App, debe llamarse static, dentro de ella AppCoder. Dentro de esta última pondremos el esqueleto que nos bajamos.

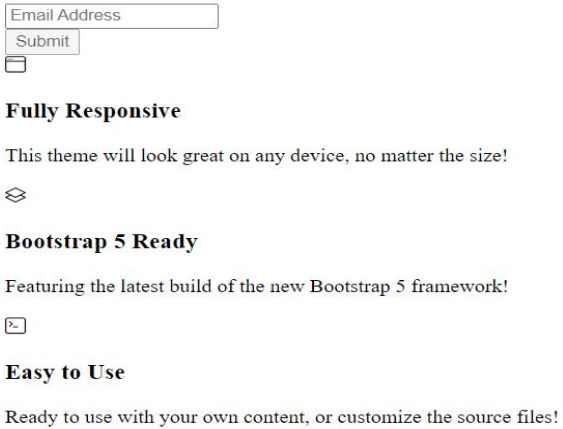
Para eso copiamos todo el contenido en inicio.html

Este será nuestro nuevo inicio.html

CODERHOUSE

Mejorando nuestros Templates

4- No queremos que se vea asi, ¿Verdad?:



5 - Para evitar eso, aplicamos lo siguiente:

a- Agregar los archivos de static:

```
<head>
    {% load static %}
    <meta charset="utf-8" />
    <meta name="viewport" content="
    <meta name="description" conter
```

b- Cambiamos direcciones

CODERHOUSE

Mejorando nuestros Templates

Antes 😔

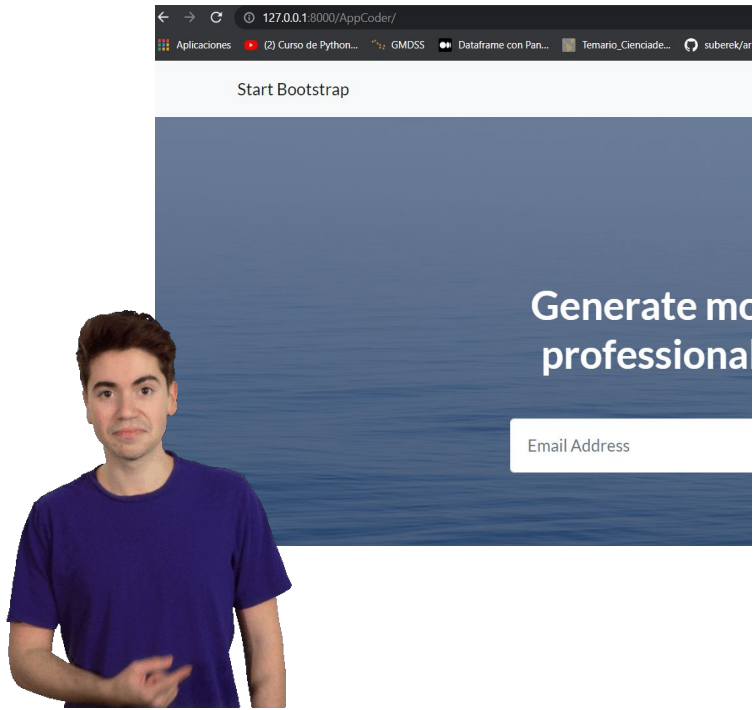
```
<!-- Core theme CSS (includes Bootstrap)-->
<link href="css/styles.css" rel="stylesheet" />
```

Después 🙂

```
<!-- Core theme CSS (includes Bootstrap)-->
<link href="{% static 'AppCoder/css/styles.css' %}" rel="stylesheet" />
```

Mejorando nuestros Templates

¡Listo! Tenemos una web estéticamente correcta que iremos completando con datos y lógica que proviene de Django.



CODERHOUSE

Último retoque al Template

CODERHOUSE

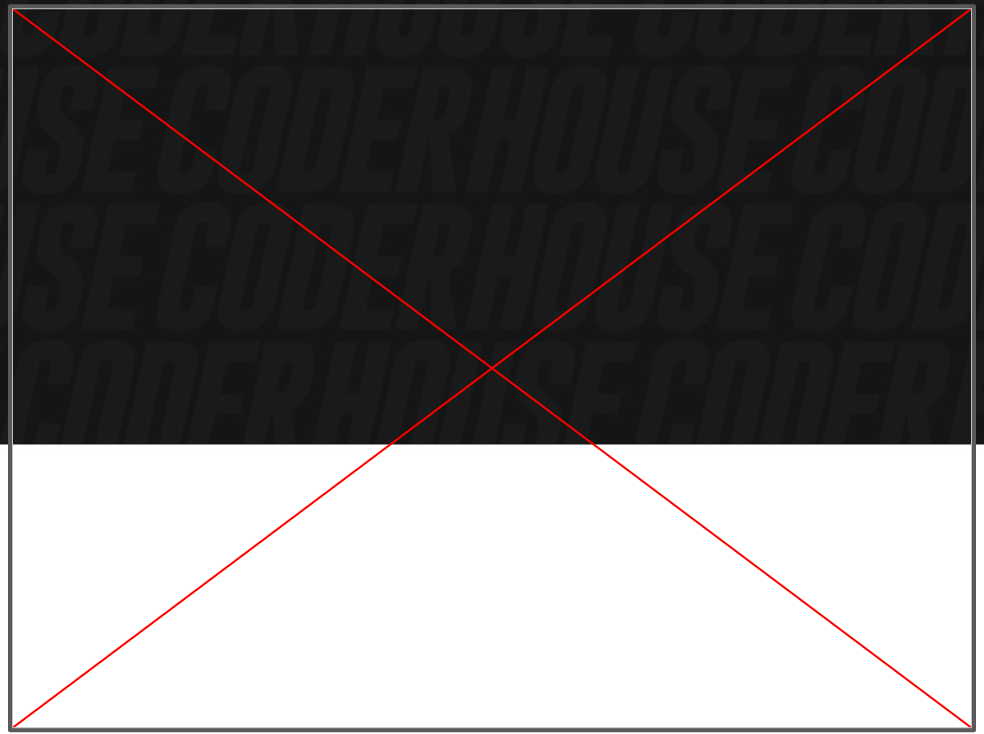
Mejorando nuestros Templates

Dejaremos la Web con este aspecto, para tener un menú inicial, un pie de página y un cuerpo.

¿Cómo? Veámoslo en el vivo...

Ejemplo en vivo

Vamos a modificar el template por defecto para solo dejar lo que nos interesa sin perder la estética y sin saber nada de desarrollo web.



Jugando con HTML

Descargar un HTML cualquiera y alterarlo a gusto.

Duración: **15 minutos**

CODERHOUSE

ACTIVIDAD EN CLASE

Jugando con HTML

Descargarás un HTML cualquiera y alterarlo a gusto. Tratar de conocer cada una de las partes que forman a un archivo HTML y las clásicas y principales para cualquier web, el navm e footer, os botones, el body, las img, etc.

NOTA: En caso de no saber cuál descargar o cuál probar les dejamos este link de descarga:

Free One Page Website Templates (501) | free-css.com

#Codertraining

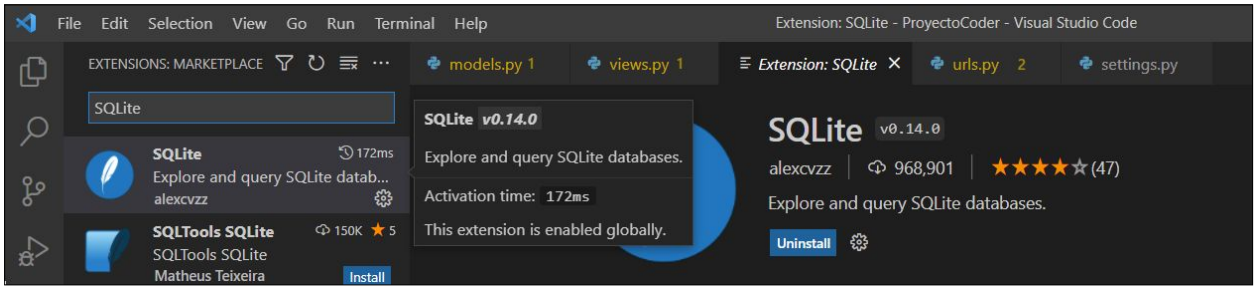
¡No dejes para mañana lo que puedes practicar hoy! Te invitamos a revisar la Guía de Ejercicios Complementarios donde encontrarás un ejercicio para poner en práctica lo visto en la clase de hoy.

CODERHOUSE

Actividad extra Nº 7

Tutorial SQLite para VSC

¡Empecemos!
Instalamos la extensión de SQlite para VSC



CODERHOUSE

¿Aún quieres conocer más? Te recomendamos el siguiente material

CODERHOUSE

MATERIAL AMPLIADO

Recursos multimedia

- ✓ Cómo instalar BootStrap 5 en tu proyecto| **Kiko Palomares**

Disponible en nuestro repositorio.

¿Preguntas?

Opina y valora esta clase

Muchas gracias.

#DemocratizandoLaEducación

Resumen
de la clase hoy

- ✓ Mejoramos la relación Template - URLs
- ✓ Construimos URLs en nuestra APP
- ✓ Creamos templates más profesionales