Clase 19 - Playground intermedio (parte I)

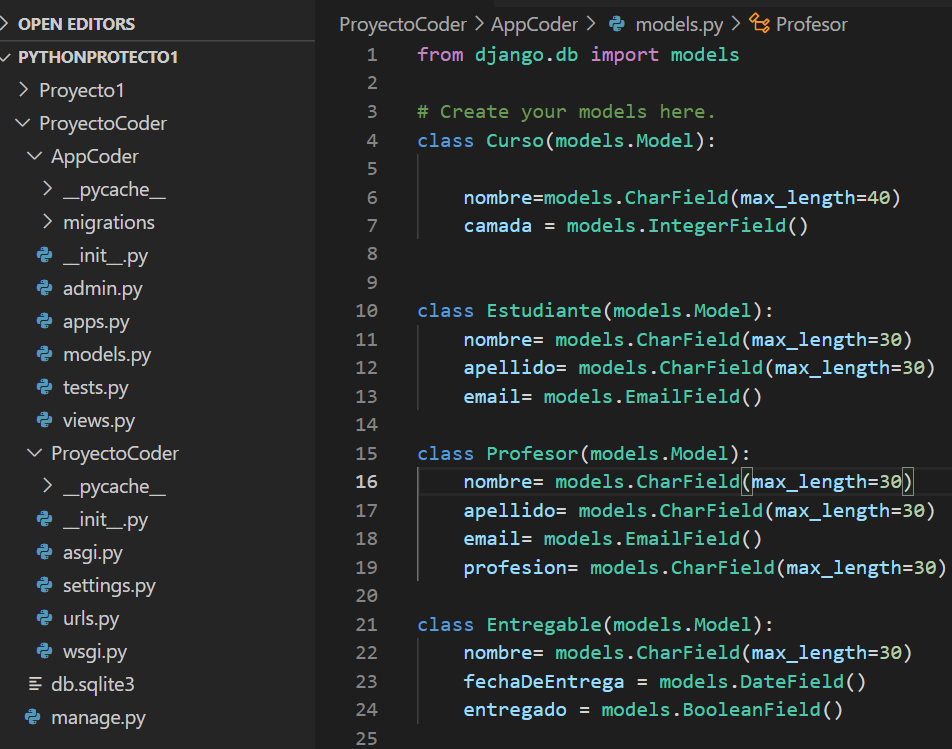
# **Profundizando MVT**

**¿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)

Recordando que por ahora tenemos algo así



**Controlar versiones**

**¿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.

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.

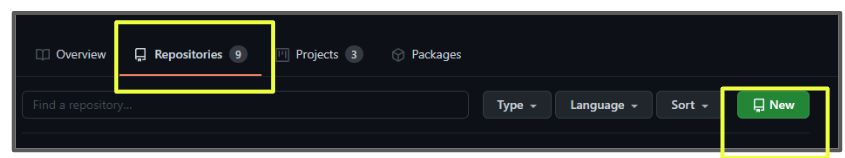
**Paso a Paso con Git**

**¿Cómo Controlar versiones compartiendo con Github?**

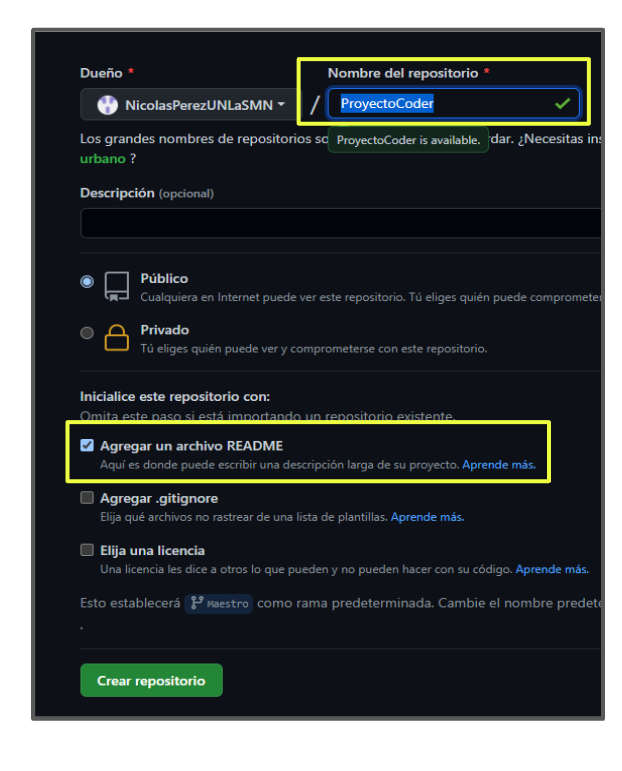
1. Entrar a [www.github.com](http://www.github.com) e ingresar con tu usuario y pass
2. Crear un nuevo repositorio (Mismo nombre del proyecto en lo posible)
3. Copiar el acceso a nuestro Repositorio Online
4. Clonar el repositorio
5. Ingresar al repositorio
6. Poner nuestro proyecto en el repositorio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

**Controlar versiones**

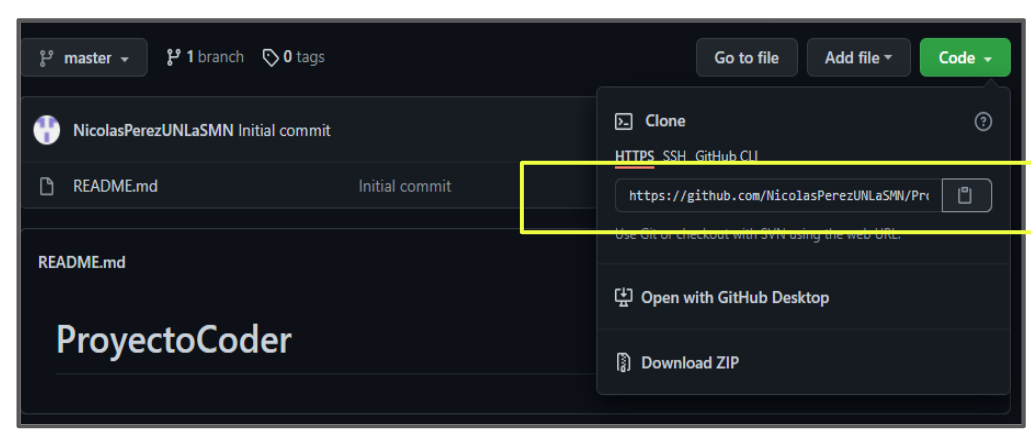
1. Entrar a www.github.com e ingresar con tu usuario y pass.



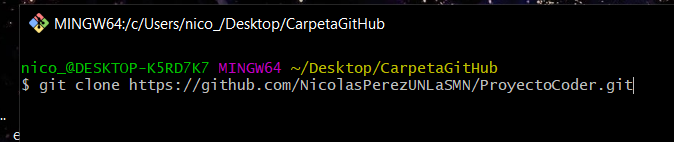
2. Crear un nuevo repositorio (Mismo nombre del proyecto en lo posible).



3. Copiar el acceso a nuestro Repositorio Online.

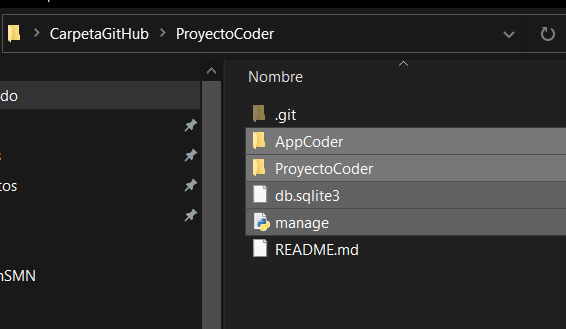
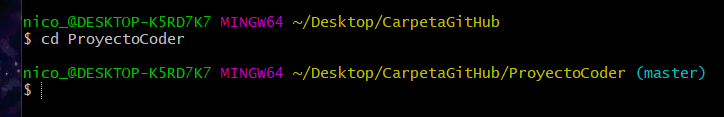


4. Clonar el repositorio

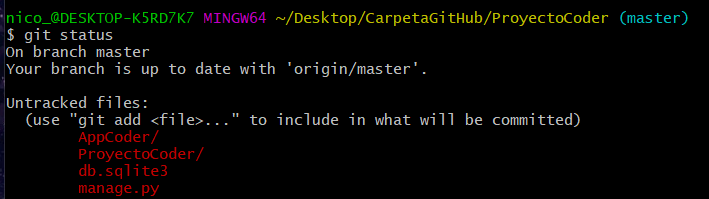


5. Ingresar al repositorio

6. Poner nuestro proyecto en el repositorio



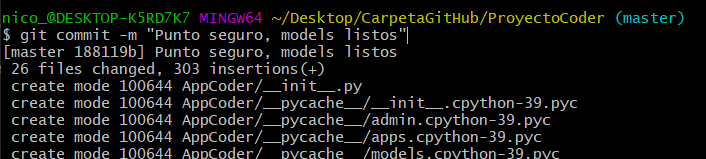
7. Ver que está todo listo con un git status.



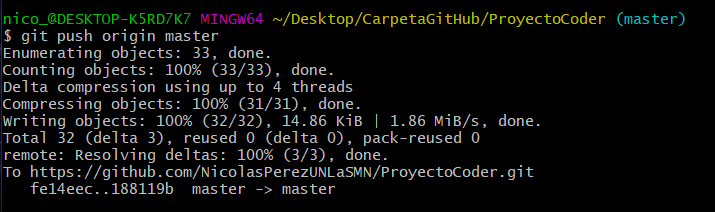
8. Cambiar de estado con git add.



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



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



11. Chequear que se subió a la web



**Fin del proceso**

¡Listo! Además de tener una versión segura en tu ordenador, a la que podrás volver siempre en caso 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). 😉

**Visitas y URLs**

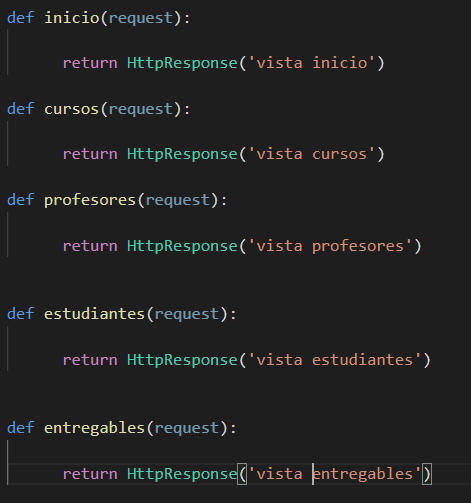
**(URLs avanzadas)**

**Visitas**

**Creando visitas**

Estamos tratando de estructurar nuestro proyecto para incorporar nuevos conceptos.

Iniciemos creando algunas vistas asociadas a nuestro modelo:

****

**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.

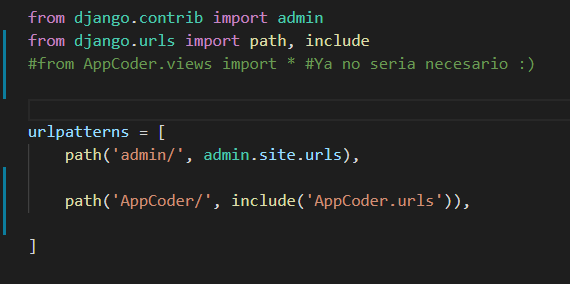
**Generar archivo URL. PY en nuestra app**

**Paso a paso**

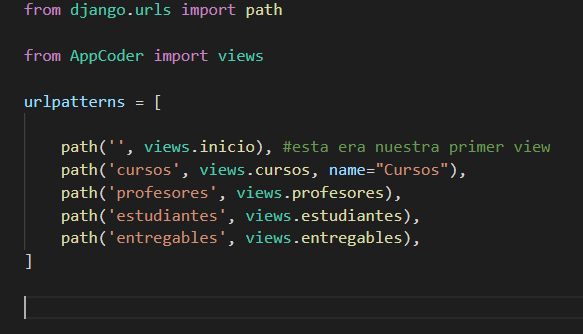
1. Crear en AppCoder urls.py
2. Le importamos el path: **from django.urls import path**
3. Importamos las vistas **from AppCoder import views**
4. Copiamos y pegamos el urlpatterns que ya teníamos, pero sin el admin
5. Dejamos el admin, SOLO, en el url del Proyecto (tampoco necesita las vistas)
6. Lo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

**URLs**

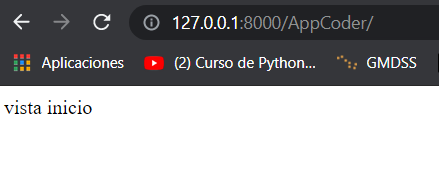
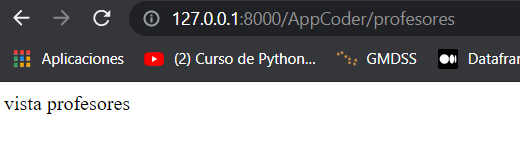
**URLs del Proyecto**



**URLs de la App**

****

**Todo está bien 🙂**

****

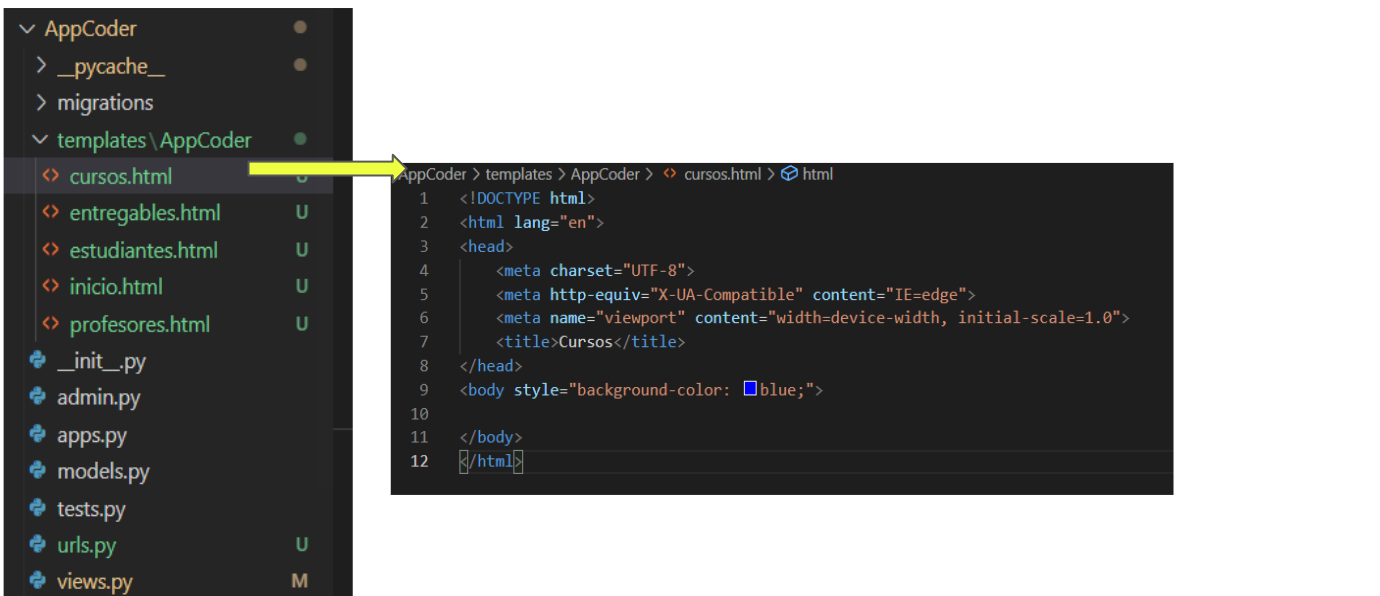
**Creando nuestros Templates**

**Puliendo estructura y definiciones**

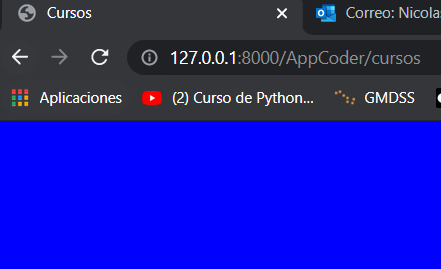
**1-** Vamos a la App y creamos una carpeta templates, y dentro

de ella una subcarpeta que se llame AppCoder

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

****

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



**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.

****

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

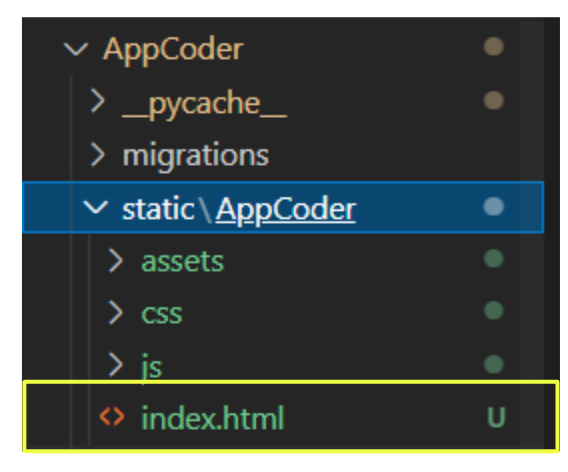
****

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

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

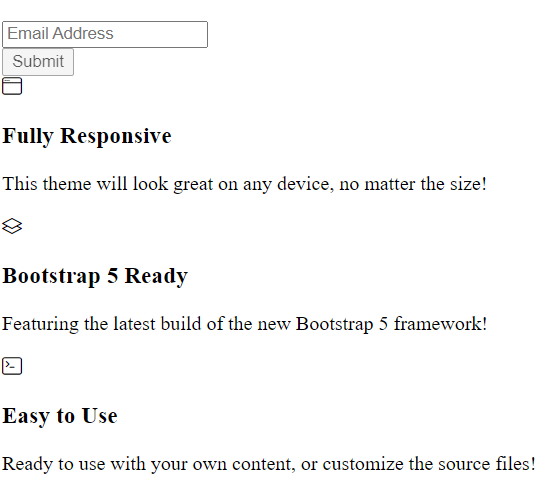
**Para eso copiamos todo el contenido en inicio.html**

Este será nuestro nuevo inicio.html

****

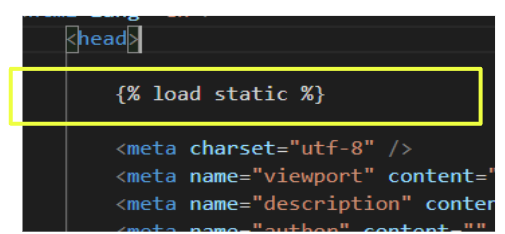
**4-** No queremos que se vea asi,

¿Verdad?:

****

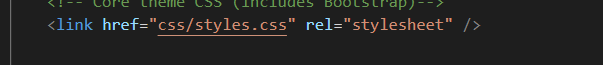
**5-** Para evitar eso, aplicamos lo siguiente:

a- Agregar los archivos de static:

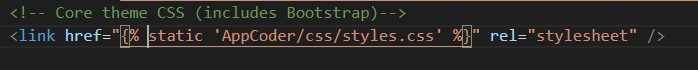


b- Cambiamos direcciones

**Antes 😔**

****

**Después 🙂**



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

