Sitio Web con Django

Eidder Andres Ardila Pita

Servicio Nacional De
Aprendizaje Centro Minero
Análisis y desarrollo de software
(ADSO) Ficha:2874586

Andrés Felipe Sandoval Higuera

Sogamoso, Boyacá

Julio 2025

Tabla de contenido

INTRODUCCION

En el presente trabajo se explica cómo visualizar la información de una base de datos dentro de un sitio web utilizando Django. Para ello, se emplean *templates* creados en archivos HTML, los cuales contienen la estructura necesaria para mostrar correctamente los datos. Además, se configuran las *URLs* que permiten acceder a cada página y, mediante las *views*, se define la lógica que determina lo que se mostrará en el navegador.

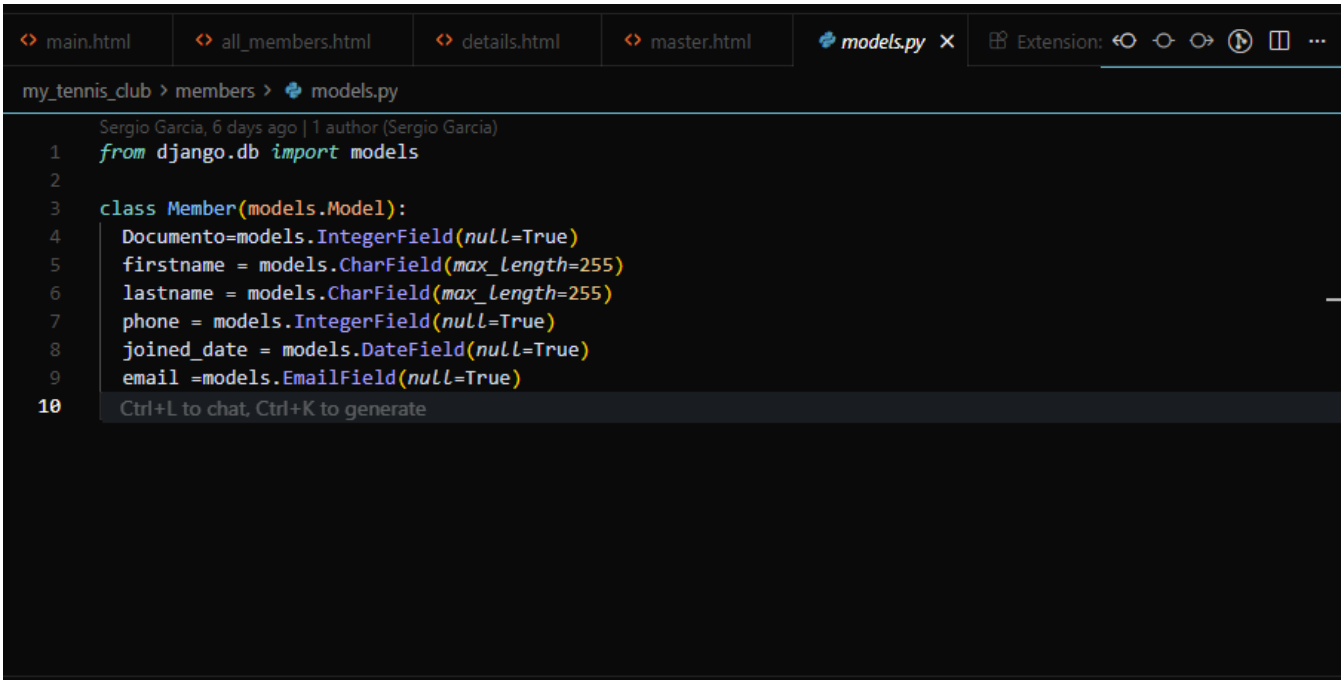
También se parte de los *modelos*, donde se estructura la base de datos a través de tablas con sus respectivos campos. Estos modelos son fundamentales para acceder a la información y presentarla de manera adecuada en el sitio web.

Durante el desarrollo se utilizaron comandos y configuraciones esenciales para enlazar las plantillas, aplicar estilos y asegurar que cada página estuviera correctamente conectada y funcional.

Gracias al uso de Django, se adquirieron conocimientos importantes que contribuyeron al aprendizaje y facilitaron la realización del proyecto.

SITIO WEB -DJANGO

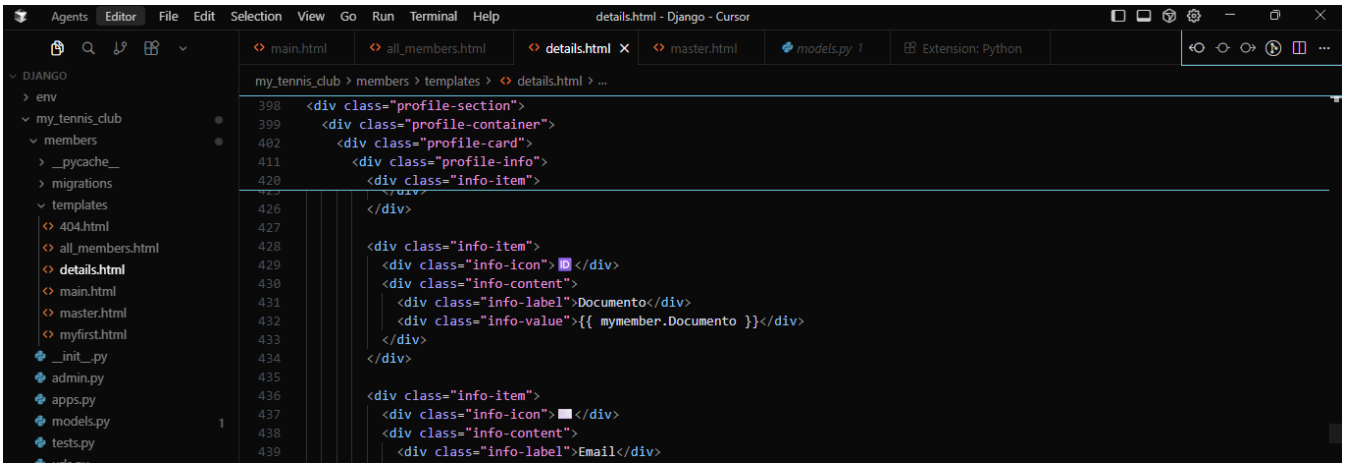
- **Modelo**



Para poder importar una tabla de una base de datos, insertamos los datos en el archivo **models.py** en el cual se pueden ver todos los campos que tiene nuestra base de datos, creamos una clase llamada **Member** y dentro de ella se todos los campos correspondientes a la base de datos, esta nos servirá para incorporarla a la vista de la página web.

Vistas

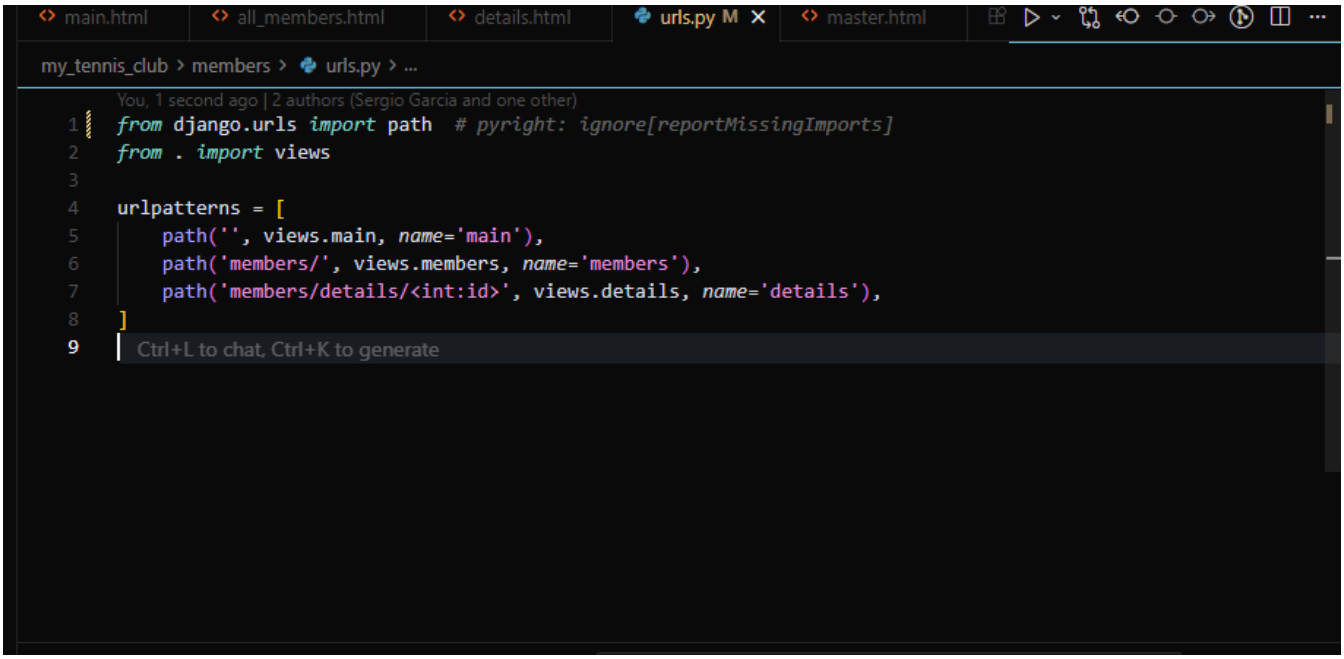
• Templates



Dentro de la carpeta **templates** se crearon varias plantillas html para poder visualizar las bases de datos desde una pagina web, sin embargo en esta creamos una principal **master** que va a ser la pagina principal que se mostrara junto a las anteriores que creamos, es decir qué las plantillas comparten el

código html, y por esto en cada plantilla estará **{% extends "master.html" %}**para poder tener conexión directamente con la plantilla que contiene todo el html, también usamos **{% block title %}{% endblock %}, {% block content %} {% endblock %}** para poder poner el contenido que será diferente a la plantilla de **master.html**.

• Urls



En las Urls se tendrán que importar las direcciones que tendrán las paginas para poder visualizarlas, por esto hacemos primero unas importaciones para las urls de **path** y de las **views** esto para que se vean correctamente a la hora de correr el código.

Dentro de **urlpatterns** se verán todas las agregaciones path con las urls que tendrán cada una de las paginas, dentro de las cuales es necesario llamar a las vistas y el nombre del html con sunombre.

• Views

main.html | all_members.html | details.html | urls.py M | views.py | master

my_tennis_club > members > views.py > ...

You, 1 second ago | 2 authors (Sergio Garcia and one other)

```python
from django.http import HttpResponse  # pyright: ignore[reportMissingImports]
from django.template import loader  # pyright: ignore[reportMissingImports]
from .models import Member

def members(request):
  mymembers = Member.objects.all().values()
  template = loader.get_template('all_members.html')
  context = {
    'mymembers': mymembers,
  }
  return HttpResponse(template.render(context, request))

def details(request, id):
  mymember = Member.objects.get(id=id)
  template = loader.get_template('details.html')
  context = {
    'mymember': mymember,
  }
  return HttpResponse(template.render(context, request))
def main(request):
  template = loader.get_template('main.html')
  return HttpResponse(template.render())
```

You, 1 second ago • Uncommi

En las vistas se harán las importaciones necesarias para poder ver el sitio web, Se hará la

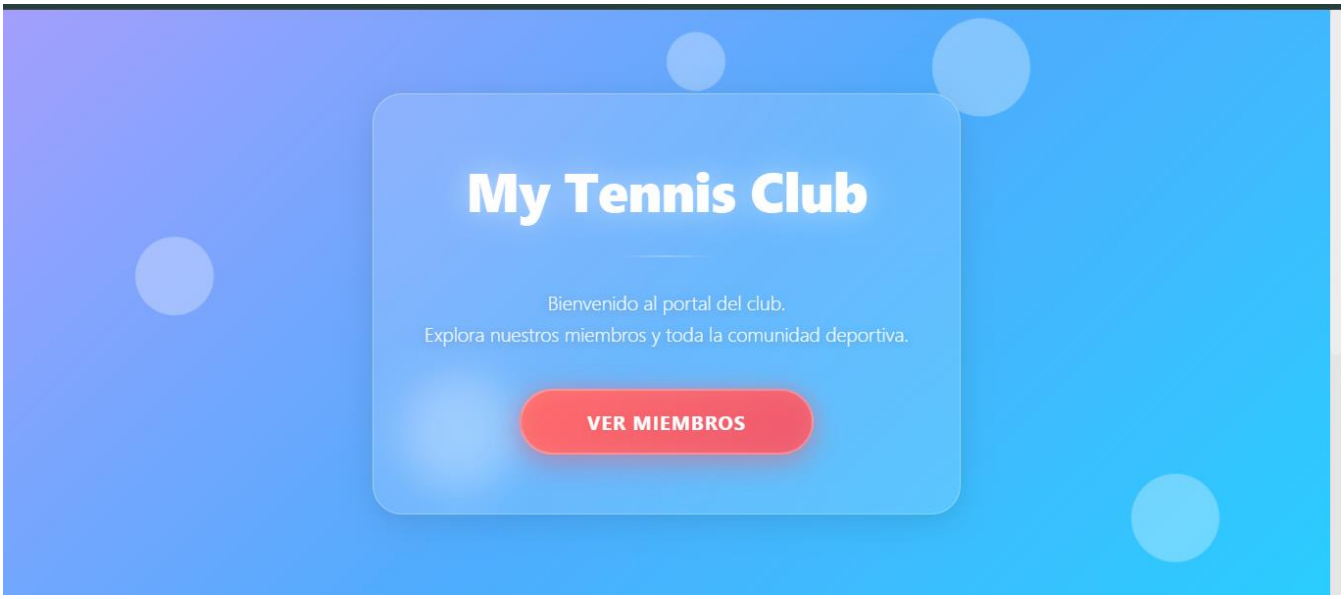
importación de **from django.http import HttpRespons** para poder tener una respuesta en http en el navegador web, también tenemos **from django.template import loader** la cual nos permite cargar todas las
templates es decir las plantillas de HTML, y por ultimo tenemos la importación **from .models import Member** este nos ayuda a importar la base de datos con que tenemos en los models.

Por otro lado, hacemos varias funciones para cada vista entre las cuales están:

- **def members(request)** dentro de la cual se llamara a los datos de la base de datos **members** en esta con el comando **Member.objects.all()** se hará la consulta los datos que se encuentran en la base de datos, también tenemos en cuenta llamar a la plantilla de **all_members** que va a ser en la cual se muestre todos nos nombres de los miembros del club y gracias al **return** gracias al **context** proporcionado y la información de la **request** se dará una respuesta en http.
- **def details(request, id)** dentro de esta función se tendrá en cuenta para llamar un solo dato de la base de datos con el comando **Member.objects.get(id=id)** con **template = loader.get_template('details.html')** llamamos al archivo html, dentro del cual está la información con el **context** podemos pasar por el dato que queremos tener de la plantilla y **return HttpResponse(template.render(context,request))** nos mostrara la respuesta en http.
- **def main(request)** en esta función vamos a llamar a la plantilla de main es decir nos mostrara solo la plantilla del inicio.
  Gracias al comando **template=loader.get_template('main.html')** podemos llamar a la plantilla html en la cual se va a cargar todo el contenido y con **returnHttpResponse(template.render())** nos mostrara la respuesta en http.

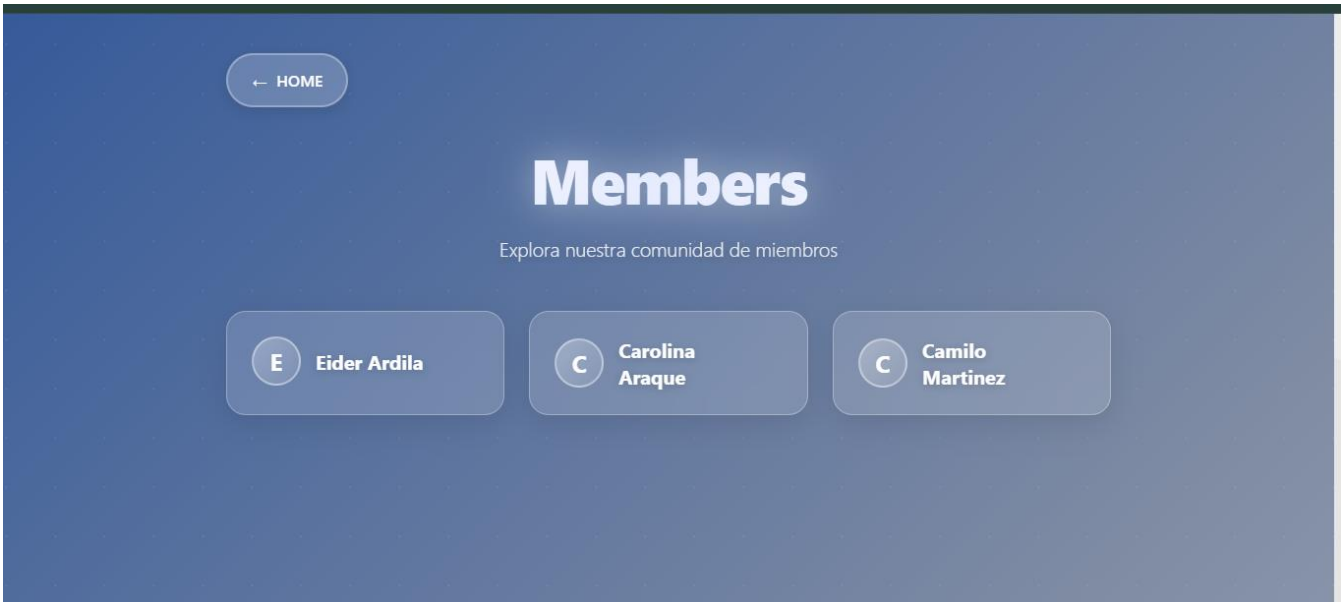
• Páginas del navegador

1. Main



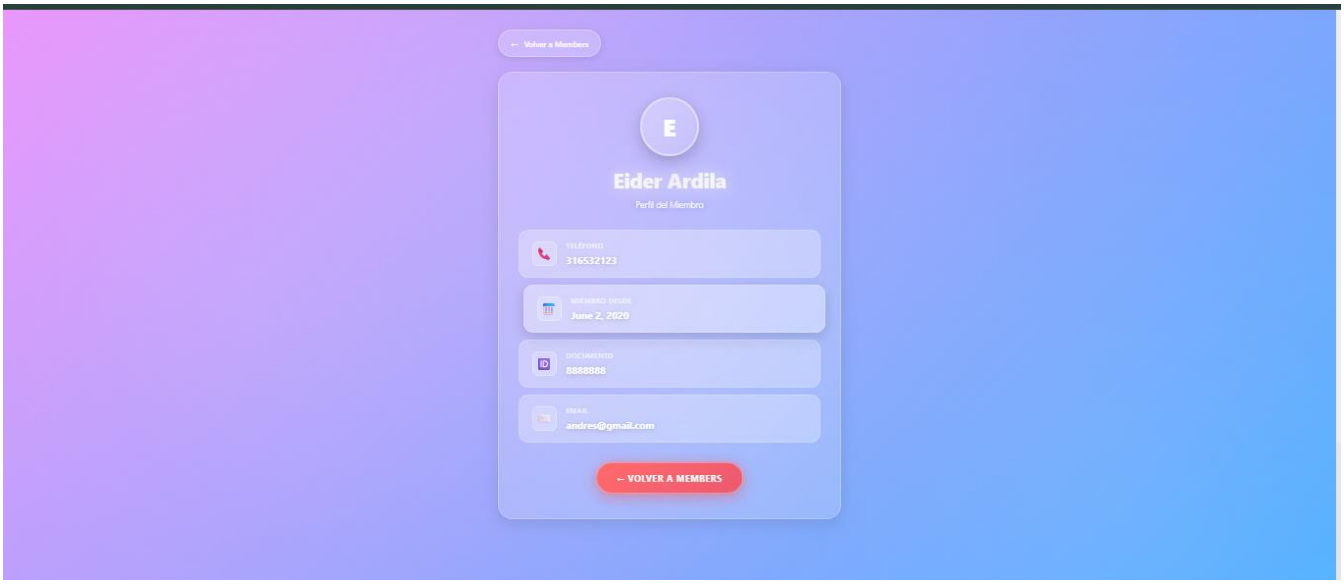
Esta es la vista de la pagina principal la cual es de la plantilla **main.html** esta es la vista principal de la página, la cual tiene un acceso a members por si se quiere ir a miembros del club de una vez.

2. Members



Esta pagina nos mostrara lo que se encuentra en la plantilla **all_members.html** se pueden ver los miembros que están en el club.

3. Details



En esta vista de la página web se ve la información de el miembro del club con sus detalles, esta vista es de la plantilla **details.html** en la cual se ver la información de un solo miembro y incluimos un botón para volver a miembros.

CONCLUSIONES

- **Integración eficiente entre base de datos y sitio web:**
- El proyecto evidencia cómo Django simplifica la visualización de información almacenada en bases de datos gracias a la integración coherente entre los modelos (datos), las vistas (lógica) y las plantillas (presentación), permitiendo construir páginas dinámicas de manera ordenada y funcional.
- **Relevancia de la arquitectura MVC/MTV:**
- Se destaca la importancia del patrón Modelo–Template–Vista (MTV) de Django, ya que facilita una estructura de código organizada, separando adecuadamente la gestión de datos, la lógica de negocio y la interfaz de usuario, lo que contribuye a un desarrollo más limpio y mantenible.
- **Desarrollo de competencias esenciales en aplicaciones web:**

- El trabajo permitió fortalecer habilidades clave en Django, como la creación y uso de modelos, vistas, URLs y plantillas, además de comandos fundamentales del framework. Estos conocimientos son esenciales para el desarrollo de aplicaciones web dinámicas y para la correcta manipulación de datos dentro de un entorno estructurado.