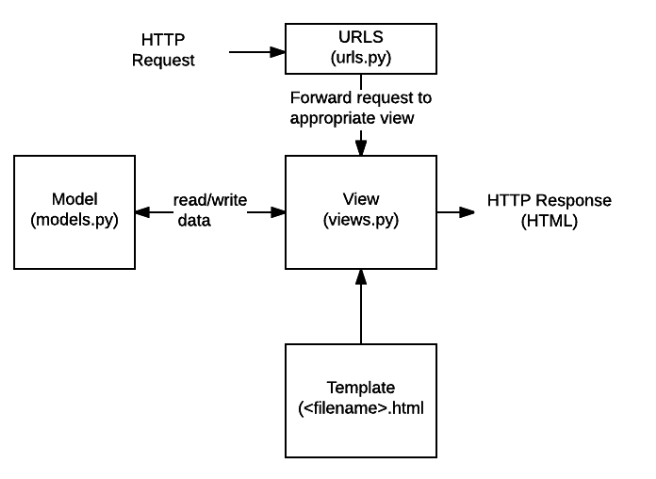
**CREACIÓN DE LA PAGINA DE INICIO**

Lo primero que necesitamos es determinar que información queremos mostrar en nuestras páginas, y definir una URL apropiada hacia estos recursos. Vamos a necesitar crear el mapeador de URLs, las vistas y plantillas para mostrar estas páginas.

Los principales elementos que necesitamos crear son:

* Mapeadores URL para reenviar las URLs admitidas (y cualquier información codificada en las URLs)  a las funciones de vista apropiadas.
* Funciones de vista para obtener los datos solicitados desde los modelos, crear una página HTML que muestre los datos, y devolverlo al usuario para que lo vea en el navegador.
* Plantillas usadas por las vistas para renderizar los datos.

**Recordemos este diagrama:**



Tendremos 5 páginas, comenzaremos con la página de inicio y en los siguientes documentos, tendremos crearemos las próximas páginas.

Las URL que vamos a necesitar para nuestras páginas son:

* catalog/ — La página home/index.
* catalog/books/ — La lista de todos los libros.
* catalog/authors/ — La lista de todos los autores.
* catalog/book/*<id>* — La vista detallada para el libro específico con un campo de clave primaria de *<id>* (el valor por defecto). Así por ejemplo, /catalog/book/3, para el tercer libro añadido.
* catalog/author/*<id>*— La vista detallada para el autor específico con un campo de clave primaria llamada *<id>.*Así por ejemplo, /catalog/author/11, para el 11vo autor añadido.

La tres primeras URLs son usadas para listar el índice, los libros y autores. Esto no codifica ninguna información adicional, y mientras los resultados retornados dependerán del contenido en la base de datos, las consultas que se ejecutan para obtener la información siempre serán las mismas.

Las últimas 2 URL, son utilizadas para mostrar información detallada sobre un libro o autor específico — estas codifican la identidad de los ítems a mostrar en la URL (mostrado arriba como *<id>* ).

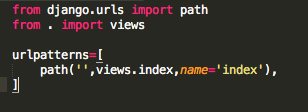
El mapeador URL puede extraer la información codificada y pasársela a la vista, donde se determinará que información extraer de la base de datos. Al codificar la información en nuestra URL solo necesitamos un mapeador de URL, una vista, y un plantilla para manejar cada libro (o autor).

CREANDO LA PAGINA INDEX

Para crear el index, necesitamos crear un mapeador URL, una vista y una plantilla.

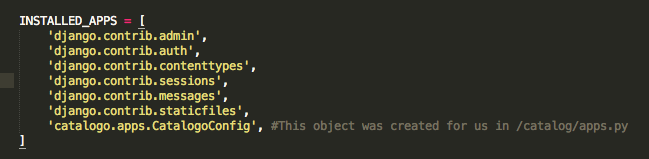
Mapeador URL:

Abra el archivo catalog/urls.py y escriba el siguiente código:



Esta función path() invoca desde la vista el método que invoca a index.html.

Adicionalmente, modifique el registro en settings de la aplicación catálogo, como se observa en la imagen:



Vista basada en funciones:

Abre catalog/views.py y nota que el archivo ya importa el atajo de la función render() que genera archivos html, usando plantillas y datos:



Al final de la función invocamos a la función render()  para crear y retornar una página  HTML como una respuesta. Esta recibe como parámetros el objeto request original (una Consulta Http), una plantilla HTML con marcadores para los datos, y una variable de contexto (un diccionario Python que contiene los datos que serán insertados en esos marcadores).

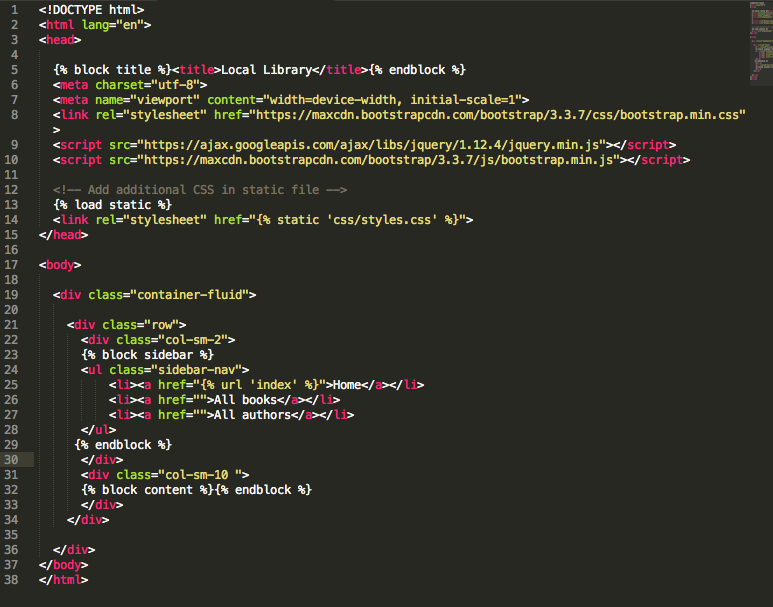
**Plantillas**

Django automáticamente buscará plantillas en un directorio llamado '**templates**' de su aplicación. Así por ejemplo, en la vista índice que acabamos de agregar, la función  render()  esperará poder encontrar el archivo **/locallibrary/catalog/templates/*index.html***, y entregará un error si el archivo no puede ser encontrado.

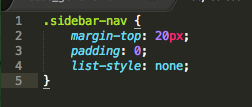
La plantilla índice va a necesitar etiquetas de marcado HTML estándar para cabecera y cuerpo junto con diversas secciones para navegar. La mayoría de esta estructura será la misma para cada página del sitio. Para evitar duplicar ese código en las diversas páginas, Django permite trabajar con una plantilla base y luego extenderla, reemplazando solo las porciones que son distintas para cada página específica, a esto se le conoce como plantilla base.

Plantilla base del proyecto

Crea un nuevo archivo -- **/locallibrary/catalog/templates/*base\_generic.html*** -- y escriba el siguiente contenido:

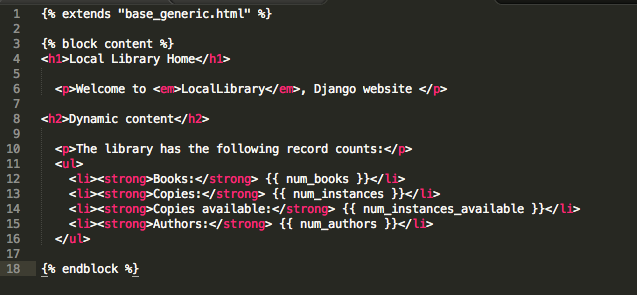


En la plantilla base, puede observar que se vincula con un archivo .css por medio de {% load static %}. Adicionalmente, puede observar el uso de {%block %} {% endblock %} permite definir bloques, los cuales pueden ser modificados. Crea **/locallibrary/catalog/static/css/styles.css** y escribe en él el siguiente contenido:



**La plantilla index**

Crea el archivo HTML **/locallibrary/catalog/templates/*index.html*** y escribe en él el contenido que se muestra abajo. Como puedes ver, extendemos nuestra plantilla base en la primera línea, y luego reemplazamos el bloque content por defecto con uno nuevo para esta plantilla.



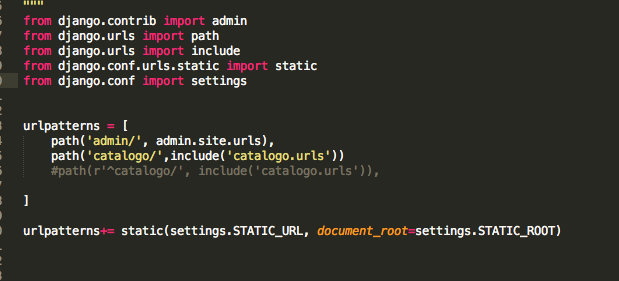
Las variables de plantillas están definidas entre dos llaves; estas variables fueron creadas en la vista. Las etiquetas están encerradas entre llaves simples con signos de porcentaje.

Lo importante de todo esto es que estas variables se nombran con las claves que enviamos dentro del diccionario context en la función render() de nuestra vista (mira abajo); estas variables serán reemplazadas por sus valores asociados cuando la plantilla sea renderizada.

return render( request, 'index.html', context={'**num\_books**':num\_books,'**num\_instances**':num\_instances,'**num\_instances\_available**':num\_instances\_available,'**num\_authors**':num\_authors}, )

Referenciando archivos estáticos

Abra el archivo locallibreria/urls.py y agregue las líneas de código que invocan a los archivos estáticos:



Django te permite especificar la ubicación de los mismos dentro de tus plantillas de forma relativa al parámetro global STATIC\_URL (el sitio web esqueleto por defecto establece el valor de STATIC\_URL a '/static/'.

Dentro de la plantilla, primero llamas a la etiqueta de plantilla load especificando "static" para añadir esta biblioteca de plantilla (como se muestra abajo). Luego de que static se carga, puedes usar la etiqueta de plantilla static especificando la URL relativa del archivo de interés.

<!-- Add additional CSS in static file --> {% load static %} <link rel="stylesheet" href="{% static 'css/styles.css' %}">

Al añadir una imagen la página sería de forma similar:

{% load static %} <img src="{% static 'catalog/images/local\_library\_model\_uml.png' %}" alt="My image" style="width:555px;height:540px;"/>

**Enlazando con URLs**

En la plantilla base de arriba se introdujo la etiqueta de plantilla url.

<li><a href="{% url 'index' %}">Home</a></li>

Esta etiqueta toma el nombre de una función path() llamada en tu archivo **urls.py,** y valores para cualquier argumento que la vista asociada recibirá desde tal función, y devuelve una URL que puedes usar para enlazar con el recurso.

Configurando donde registrar las plantillas

Para que Django encuentre los archivos de plantillas es necesario editar el archivo **settings.py** agregando el directorio donde creamos nuestras plantillas en el objeto TEMPLATES, como indica la línea en negrita a continuación:



Una vez que haya realizado todos los cambios necesarios, puede ejecutar el servidor y luego ir a la dirección: <http://127.0.0.1:8000/catalogo/> deberá poder ver lo siguiente:

