

**Nombre del alumno:** Rosa Mariana Gutiérrez Miranda.

**Materia cursada:** Programación Web 2.

**Nombre del asesor de la materia**: Mtra. Míriam Patricia Maldonado Benítez.

**Número de la actividad:** 7A.

**Tema de la actividad:** Creación de proyectos basados en MVC.

Con base en las lecturas, enlaces, apuntes y en una investigación personal sobre los temas del bloque, toma como referencia dentro de los apuntes el tema “Implementación del modelo”, y posteriormente:

1. Construye un proyecto que cumpla con el patrón MVC, la temática es libre, pero toma en cuenta:

a) Se crea la estructura de carpetas adecuadas para almacenar los archivos del patrón MVC.

b) La separación de la vista, el modelo y el controlador; debe poderse ver claramente la separación de los elementos de MVC y que se distingan los tres elementos.

c) El funcionamiento del proyecto presentado trabaja correctamente y no presenta ningún error.

2. Elabora un reporte en donde se describa claramente cómo se implementó el MVC y qué archivos corresponden a cada componente del modelo.

Primero creamos una Django app y los directorios con los siguientes comandos:

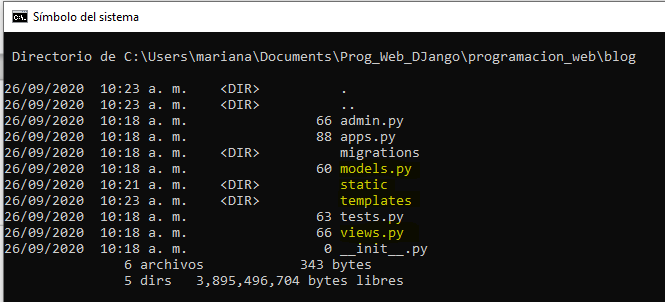
python manage.py startapp blog

mkdir blog\static

mkdir blog\templates

En **static** vamos a poner archivos estáticos (que no son dinámicos por que no cambia su contenido) los CSS, tipografías, imágenes y archivos de JavaScript.

En **templates** vamos a poner los archivos HTML que pueden o no llevar [tags](https://docs.djangoproject.com/en/3.1/ref/templates/builtins/) de Django para generar contenido dinámicamente.



En el archivo **models.py** es donde creamos las clases que representan una tabla en la base de datos, en este caso no vamos a utilizar bases de datos pues solo queremos demostrar el MVC de una manera simple. Pero en el siguiente trabajo de la ultima unidad estaré implementando modelos para base de datos.

El archivo **views.py** es en realidad el controlador especial que utilizaremos para mandar llamar las vistas que están implementadas en los templates de HTML. También podríamos utilizar las views.py para mediante el método return regresar el código HTML pero la verdad no es buena idea por que no es muy mantenible y se vuelve muy complicado por que tienes que estar utilizando ‘\’ barras invertidas para escapar caracteres reservados en Python que pueden ser parte del HTML, por lo que es mucho mas practico utilizar templates de HTML que te permiten insertar [tags](https://docs.djangoproject.com/en/3.1/ref/templates/builtins/) de Django especiales para desplegar contenido dinámicamente.

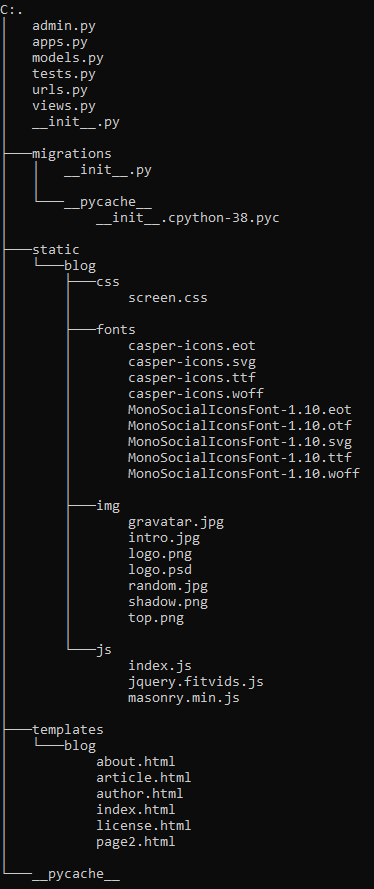
Los controladores, son archivos de python que nosotros podemos crear, por ejemplo **mis\_clases.py** o **mis\_funciones.py** o un archivo por clase, etc dentro del directorio “blog” de la app. Una vez teniendo listos los controladores estos para poder utilizarlos en el flujo del framework de Django los tenemos que importar dentro de **views.py**, y los mandamos llamar dentro de los métodos o HTTP Requests que se definen allí.

Luego de realizar el procesamiento necesario guardamos el resultado en una variable de tipo diccionario que la convención es llamarla contexto, esta variable cuando mandamos llamar a la vista que serian los templates de HTML la pasamos como argumento de esta manera dentro del template de HTML podemos insertar tags de Django para manipular los datos que vienen dentro del contexto y así desplegar el contenido dinámico. En resumen así es como Django implementa el modelo MVC.

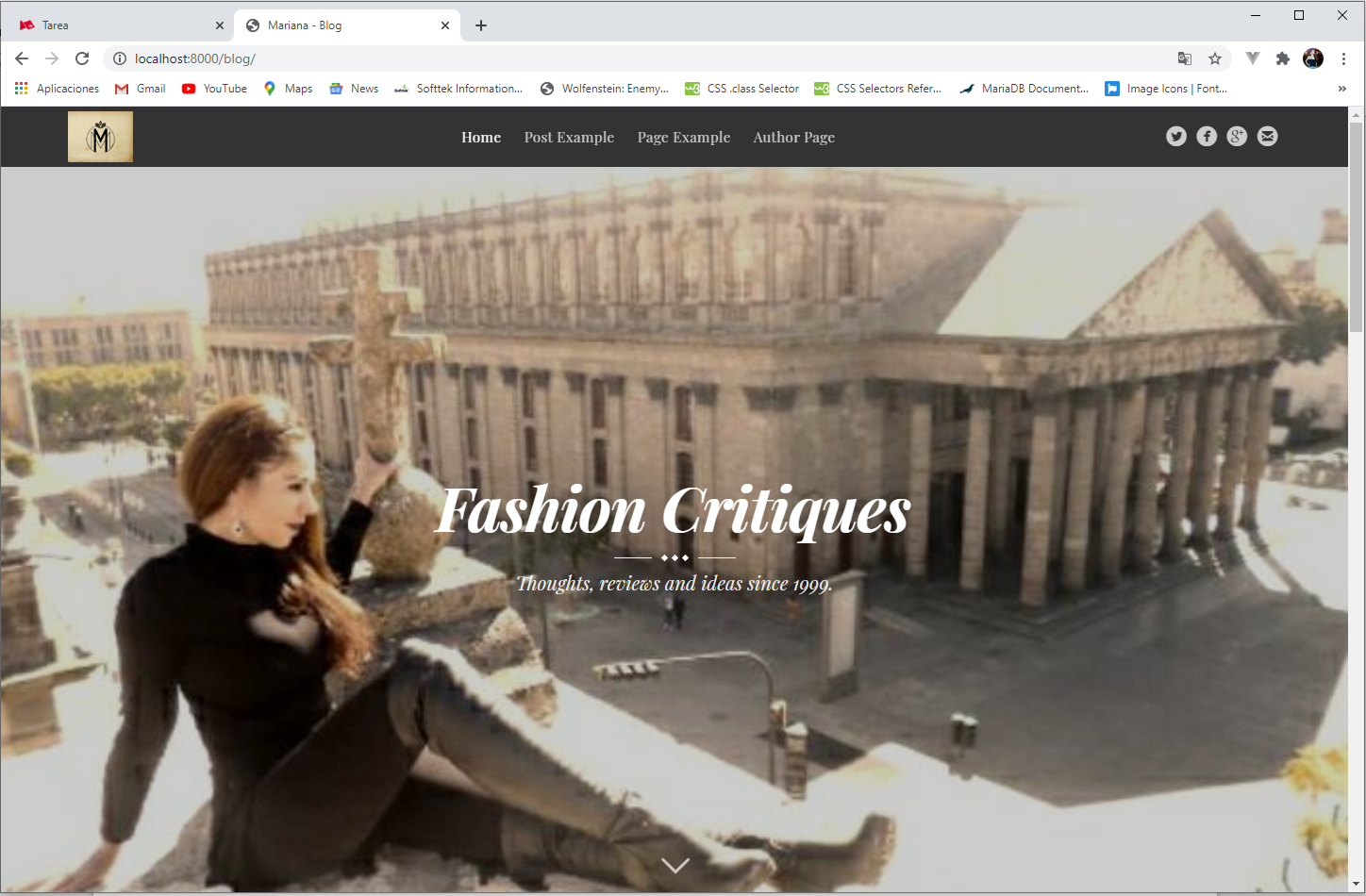
A continuación en vez de crear una pagina visualmente desde cero descargue un template gratis de CSS para crear el blog de aquí:

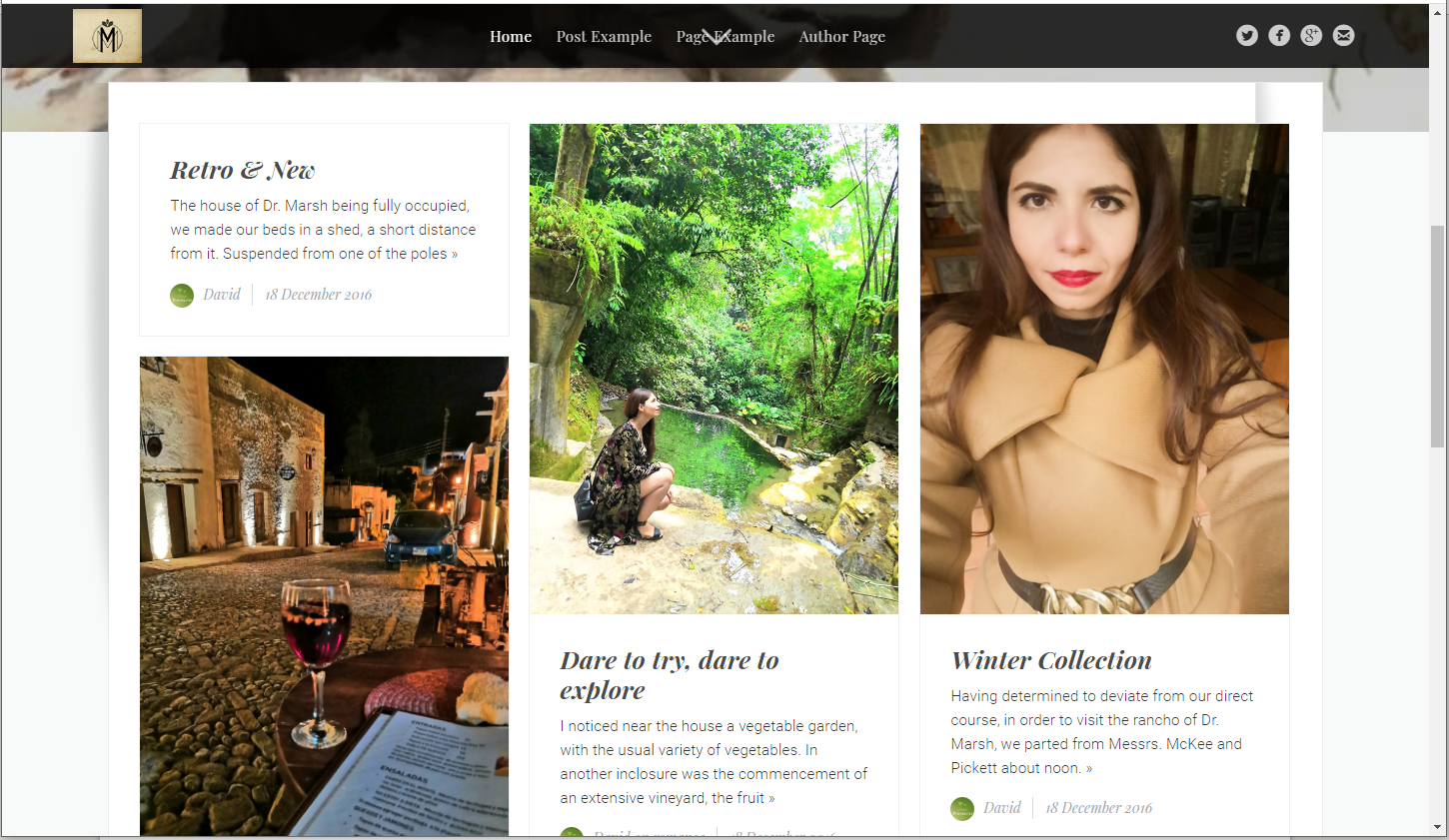
<https://www.free-css.com/free-css-templates/page257/nectaria>

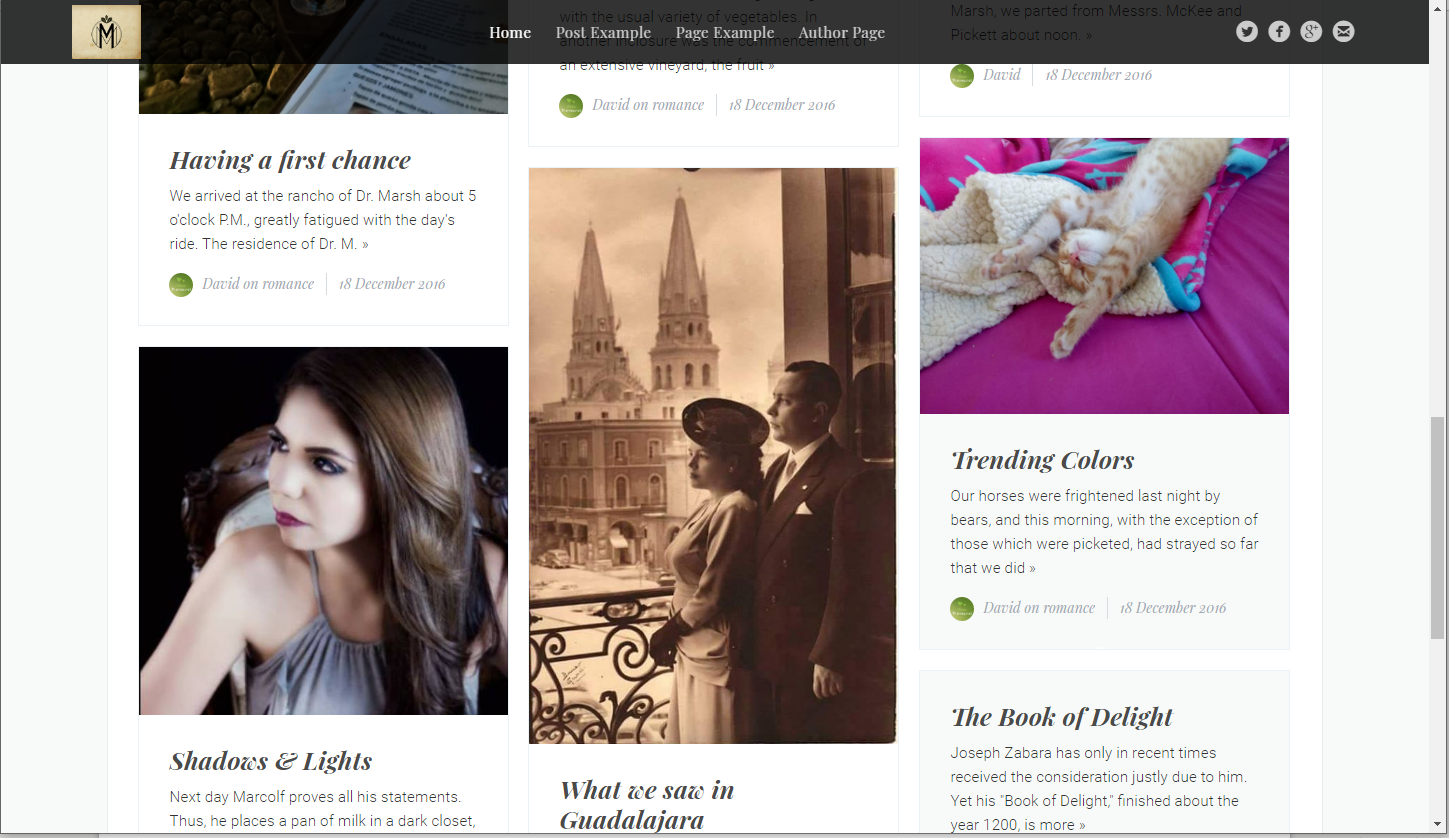
Luego lo modifique y adapte a la estructura de Django quedando la estructura de directorios y archivos de la siguiente manera:

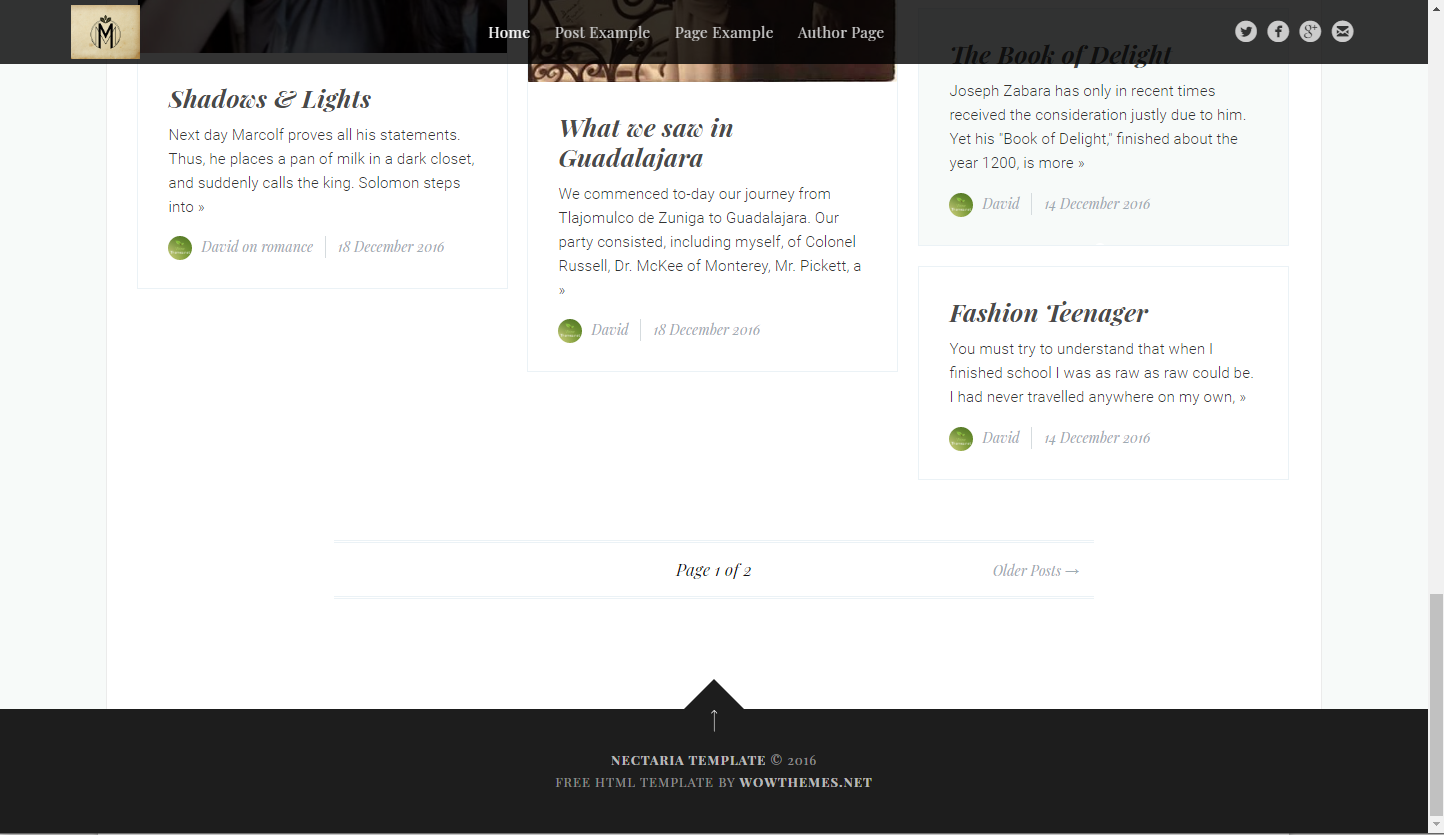


Screenshots de la pagina terminada.

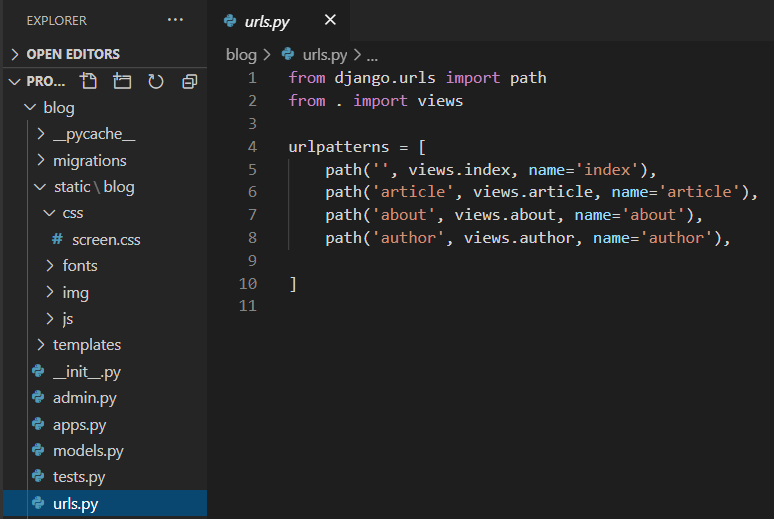
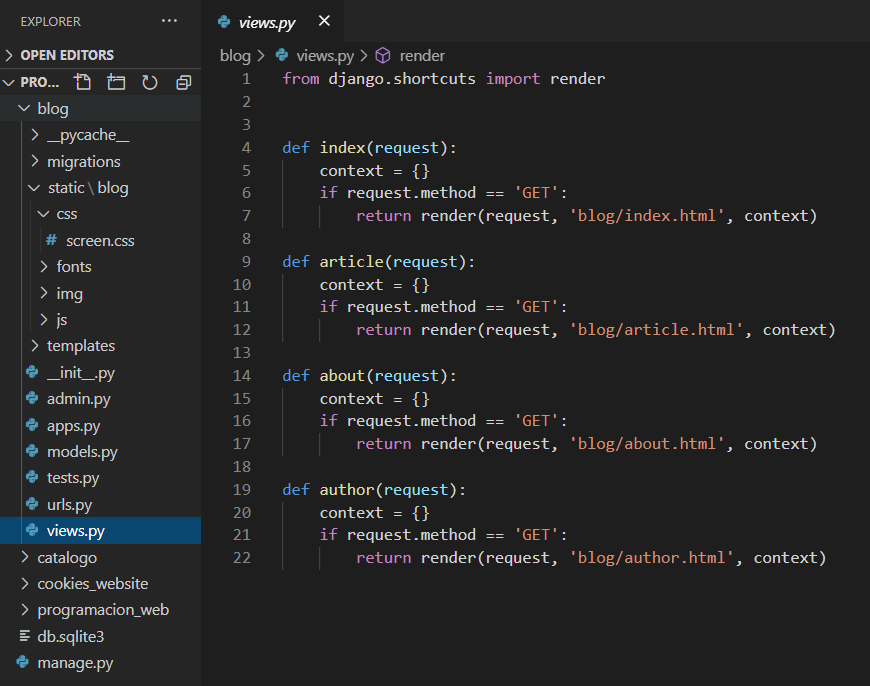


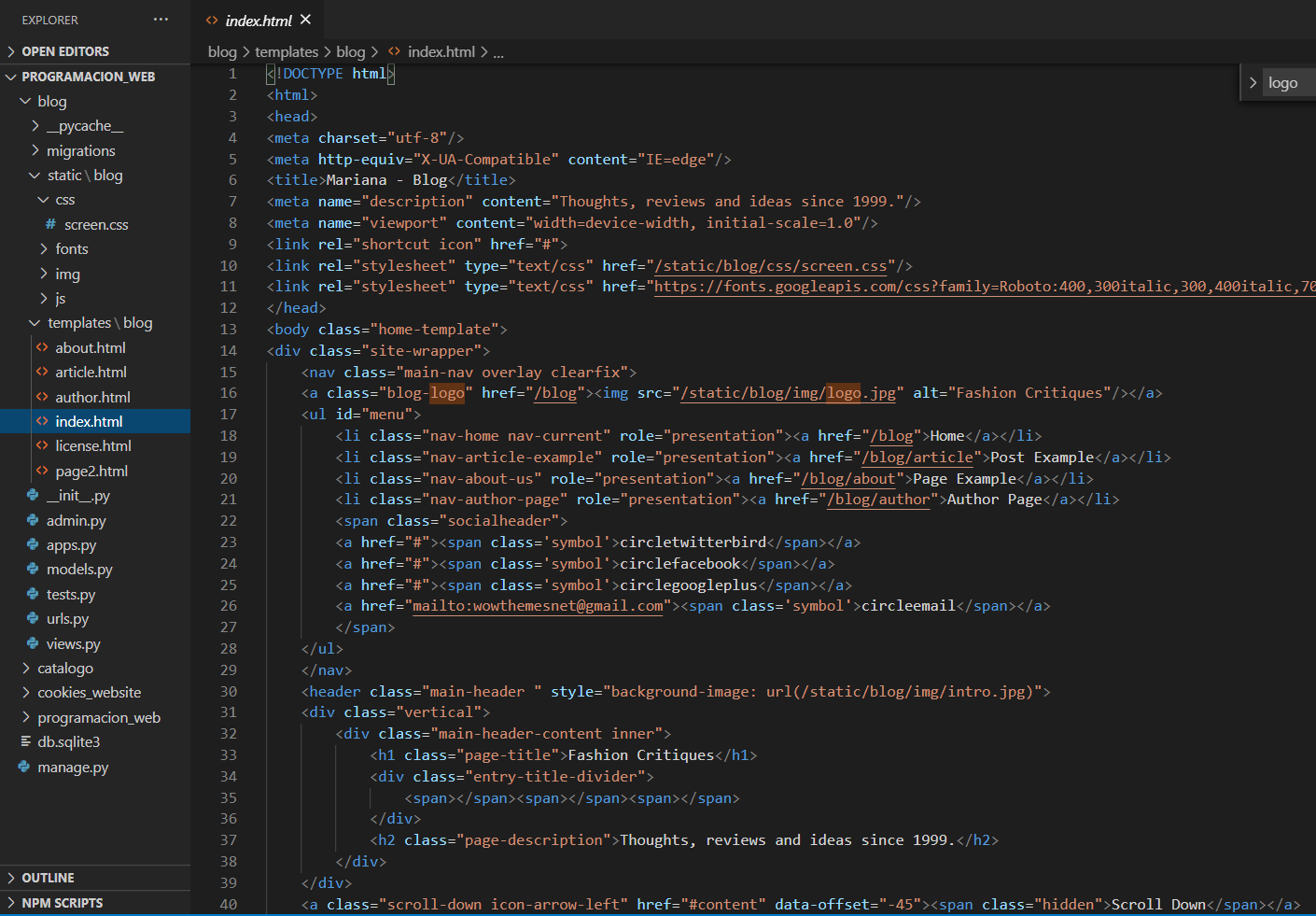


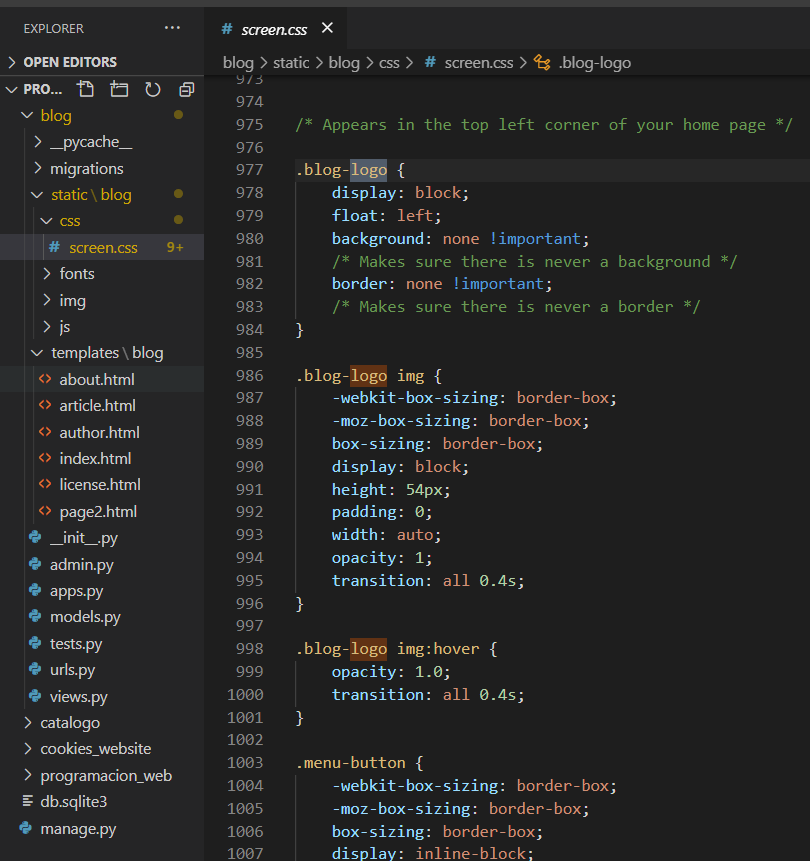


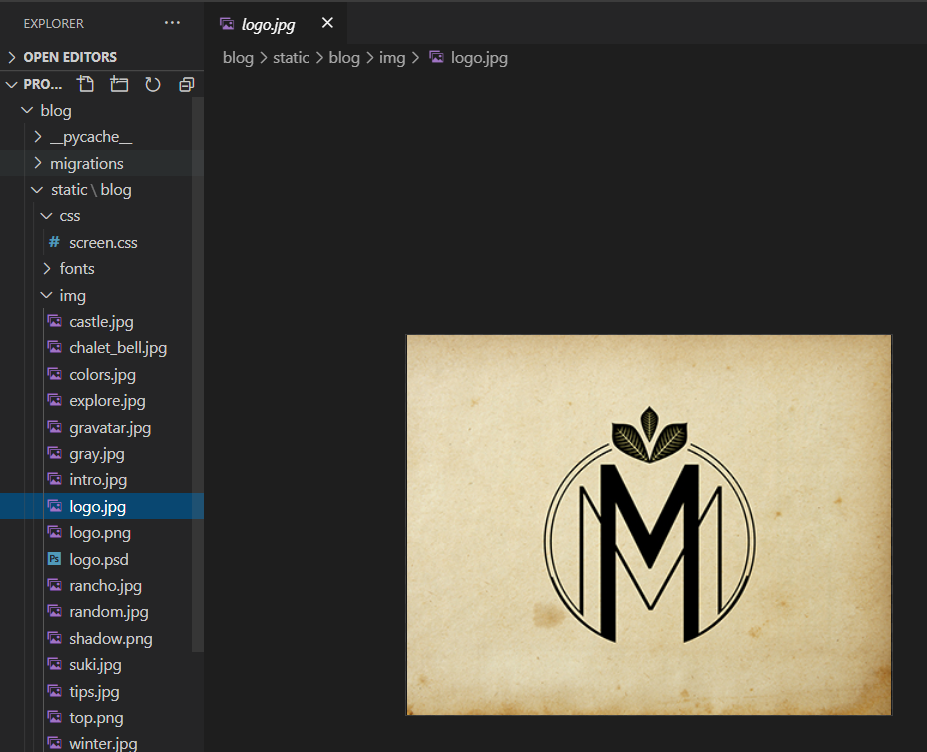


Screenshots del código:









Referencias:

<https://docs.djangoproject.com/en/3.1/>