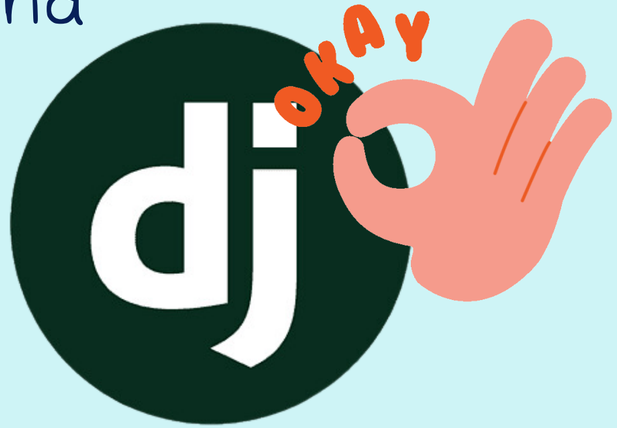
CONFERENCIA

¡Django a tu Alcance!

Da un gran paso en tus proyectos Frontend



Rodrigo Hernández

Comencemos con...

Frontend

Frontend

El desarrollo web Frontend consiste en la conversión de datos en una interfaz gráfica para que el usuario pueda ver e interactuar con la información de forma digital.

https://roadmap.sh/frontend

¿Qué es Django?

Y porque en el front?

Django

Django es un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como modelo-vista-controlador (MVC).

Fue desarrollado originalmente para gestionar páginas web orientadas a noticias de la World Company de Lawrence, Kansas, y fue liberada al público bajo una licencia BSD en julio de 2005; el framework fue nombrado en alusión al guitarrista de jazz gitano Django Reinhardt.

Django

En junio de 2008 fue anunciado que la recién formada Django Software Foundation se haría cargo de Django en el futuro.

La meta fundamental de Django es facilitar la creación de sitios web complejos. Django pone énfasis en el re-uso, la conectividad y extensibilidad de componentes, el desarrollo rápido y el principio «DRY» (del inglés Don't Repeat Yourself,«No te repitas»).

Características

- Un mapeador objeto-relacional.
- Aplicaciones "enchufables" que pueden instalarse en cualquier página gestionada con Django.
- Una API de base de datos robusta.
- Un sistema incorporado de "vistas genéricas" que ahorra tener que escribir la lógica de ciertas tareas comunes.
- Un sistema extensible de plantillas basado en etiquetas, con herencia de plantillas.

Características

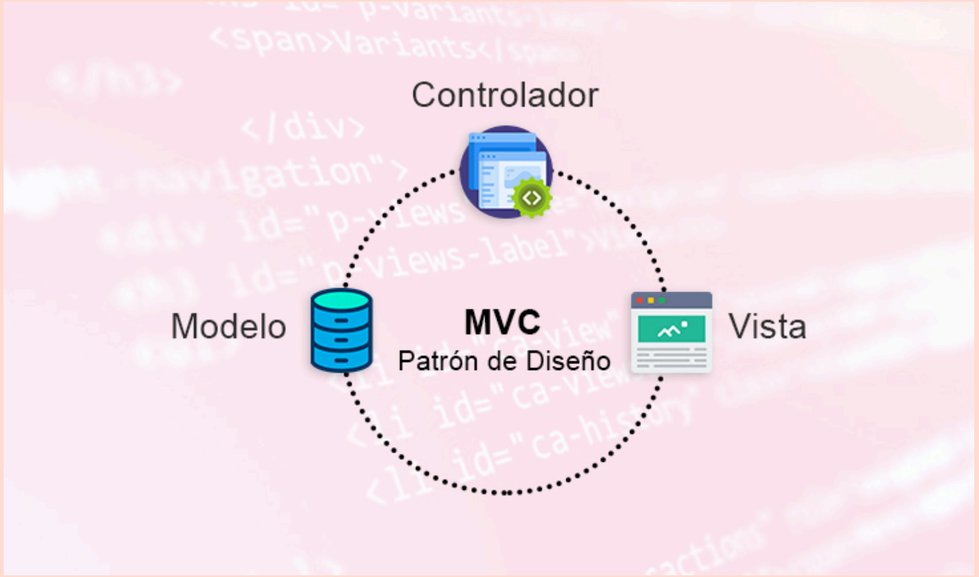
- Un despachador de URLs basado en expresiones regulares.
- Un sistema "middleware" para desarrollar características adicionales; por ejemplo, la distribución principal de Django incluye componentes middleware que proporcionan cacheo, compresión de la salida, normalización de URLs, protección CSRF y soporte de sesiones.
- Soporte de internacionalización, incluyendo traducciones incorporadas de la interfaz de administración.
- Documentación incorporada accesible a través de la aplicación administrativa (incluyendo documentación generada automáticamente de los modelos y las bibliotecas de plantillas añadidas por las aplicaciones).

Arquitectura de Django

Aunque Django está fuertemente inspirado en la filosofía de desarrollo Modelo Vista Controlador, sus desarrolladores declaran públicamente que no se sienten especialmente atados a observar estrictamente ningún paradigma particular, y en cambio prefieren hacer "lo que les parece correcto". Como resultado, por ejemplo, lo que se llamaría "controlador" en un "verdadero" framework MVC se llama en Django "vista", y lo que se llamaría "vista" se llama "plantilla".

Gracias al poder de las capas mediator y foundation, Django permite que los desarrolladores se dediquen a construir los objetos Entity y la lógica de presentación y control para ellos.

Arquitectura MVC



Modelo-vista-controlador (MVC) es un patrón de arquitectura de software, que separa los datos y principalmente lo que es la lógica de negocio de una aplicación de su representación y el módulo encargado de gestionar los eventos y las comunicaciones.

Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador; es decir: por un lado define componentes para la representación de la información y, por otro lado, para la interacción del usuario.

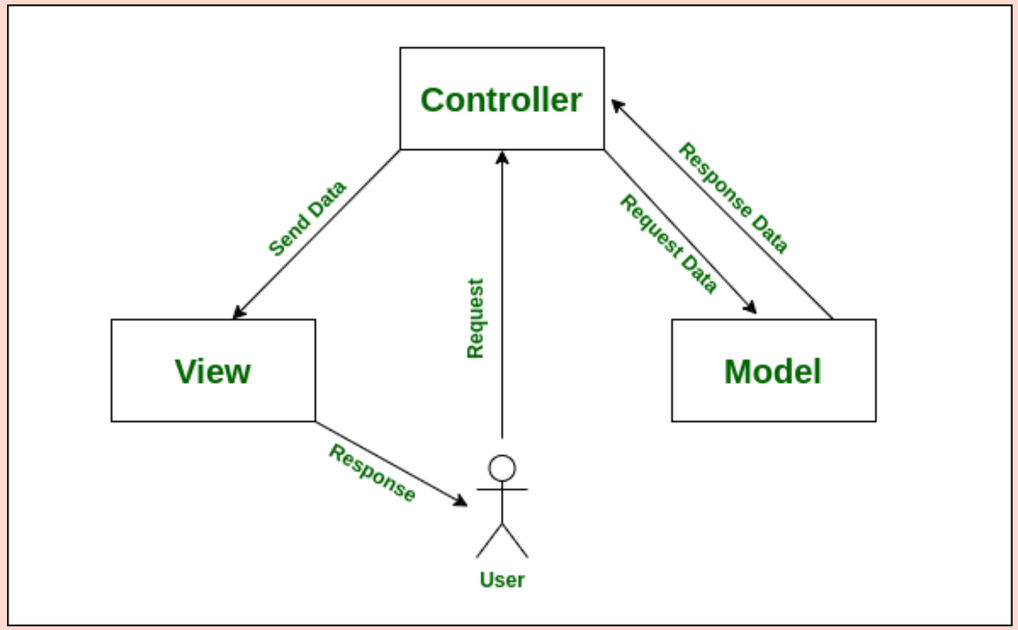
Modelo

Es la representación de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la 'vista' aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al 'modelo' a través del 'controlador'.

Vista

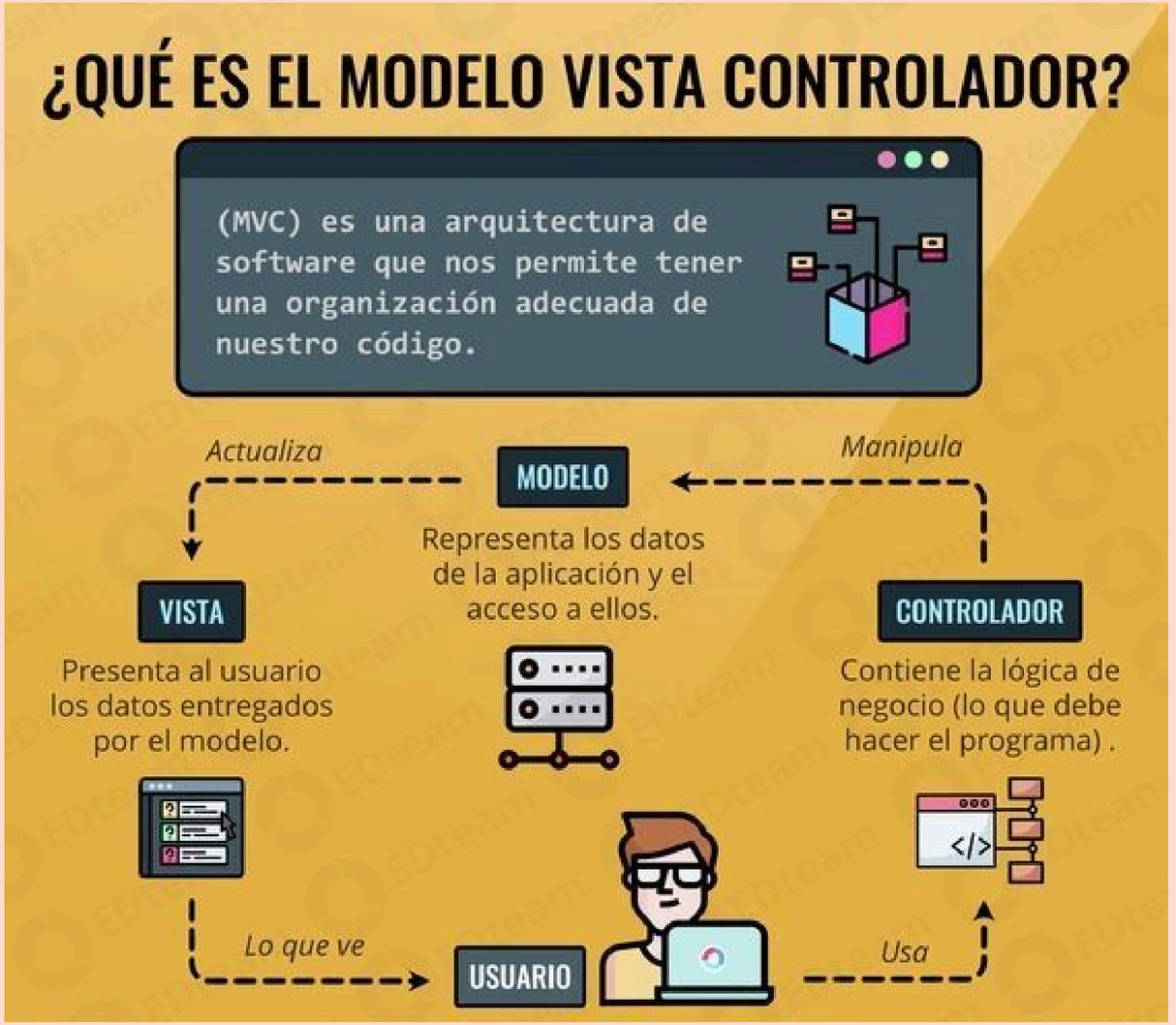
Presenta el 'modelo' (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario), por tanto requiere de dicho 'modelo' la información que debe representar como salida.

Controlador



Responde a eventos (usualmente acciones del usuario) e invoca peticiones al 'modelo' cuando se hace alguna solicitud sobre la información (por ejemplo, editar un documento o un registro en una base de datos). También puede enviar comandos a su 'vista' asociada si se solicita un cambio en la forma en que se presenta el 'modelo' (por ejemplo, desplazamiento o scroll por un documento o por los diferentes registros de una base de datos), por tanto se podría decir que el 'controlador' hace de intermediario entre la 'vista' y el 'modelo' (véase Middleware).

MVC


¿QUÉ ES EL MODELO VISTA CONTROLADOR?
(MVC) es una arquitectura de software que nos permite tener una organización adecuada de nuestro código.
Actualiza
MODELO
Manipula
Representa los datos de la aplicación y el acceso a ellos.
VISTA
Presenta al usuario los datos entregados por el modelo.
CONTROLADOR
Contiene la lógica de negocio (lo que debe hacer el programa) .
Lo que ve
USUARIO
Usa

Iniciemos con Django

Pasos para crear tu primer Proyecto en Django

Paso 1

Instala Django

```
#Inserta este comando en consola
pip install Django
```

Paso 2

Ingresa el siguiente comando

```
#Inserta este comando en consola
django-admin startproject Nombre_del_Proyecto
```

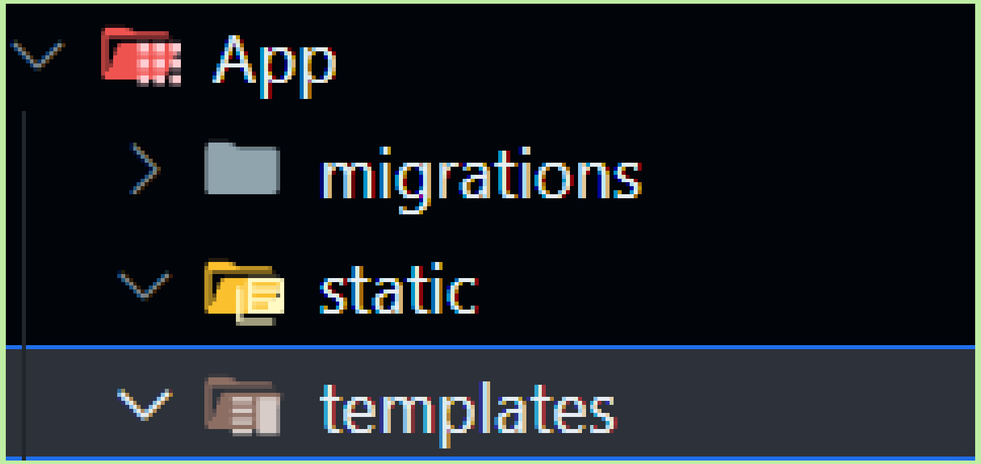
Paso 3

Nos ubicamos en la carpeta del proyecto y escribimos lo siguiente para crear nuestra App:

```
#Inserta este comando en consola
#Ingresa a la carpeta del Proyecto
cd NombreDelProyecto
#Comando para levantar Django
python manage.py startapp App
```

Paso 4

En la carpeta App agregamos las carpetas static y templates



Paso 5

Vamos a settings.py y encontraremos la siguiente linea:

```
STATIC_URL = '/static/'
```

Se dice entonces que en la carpeta que definamos acá, se guardarán los archivos estáticos que se necesiten (CSS, JS, imágenes, etc).

Paso 6

Ahora en el mismo archivo de settings.py, en la sección de Application definition, aparecerá algo como esto:

```
INSTALLED_APPS = [
    '<apps instaladas>',
]
```

Es aquí donde debemos agregar la aplicación que creamos.

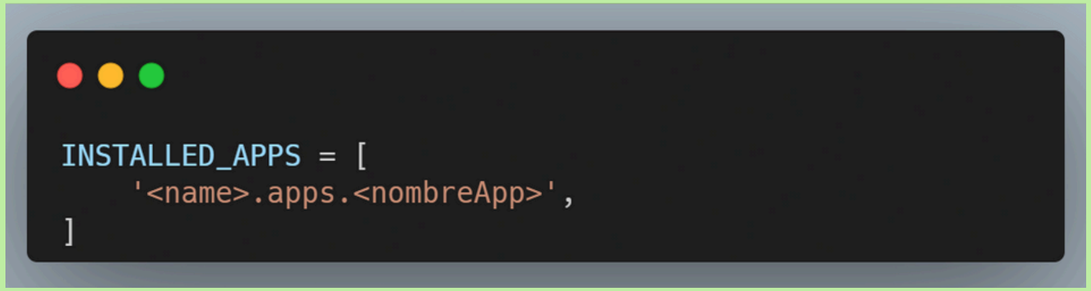
Paso 6.1

Para encontrar el nombre de la aplicación vamos a apps.py dentro de la carpeta APP y encontraremos lo siguiente:

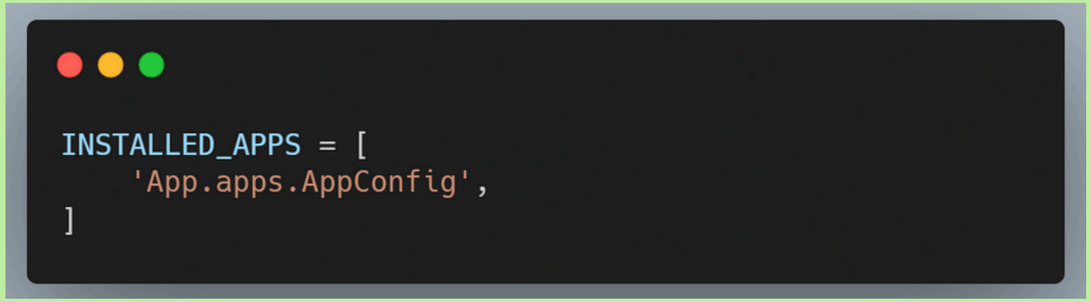
```python
from django.apps import AppConfig


class AppConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'App'
```

Paso 6.2

Debemos de copiar el nombre de la clase que está en el archivo apps.py y pegarla en la lista de INSTALLED_APPS en settings.py.



```
INSTALLED_APPS = [
    '<name>.apps.<nombreApp>',
]
```



```
INSTALLED_APPS = [
    'App.apps.AppConfig',
]
```

Paso 6.3

¡Listo, ya tenemos agregada nuestra aplicación a nuestro proyecto de Django!

Paso 7

Ahora, configuraremos lo que son las URLs, entonces hay que ir a urls.py en la carpeta del Proyecto, lo que haremos será Incluir otro URLconf. Para ello, debemos de importar la función include de Django. Para que quede así:

```
from django.urls import path, include
```

Paso 7.1

En la carpeta de la aplicación, creamos un archivo llamado urls.py. Y en este archivo, escribimos lo siguiente:

```
from django.urls import path
from . import views

urlpatterns = [
]
```

Paso 7.2

Regresamos al archivo urls.py del proyecto y lo dejamos de la siguiente manera:

```python
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    #path('admin/', admin.site.urls),
    path('', include('app.urls'))
]
```

Paso 8

Escribimos el siguiente comando en consola para levantar el Frontend:

```
python manage.py runserver
```

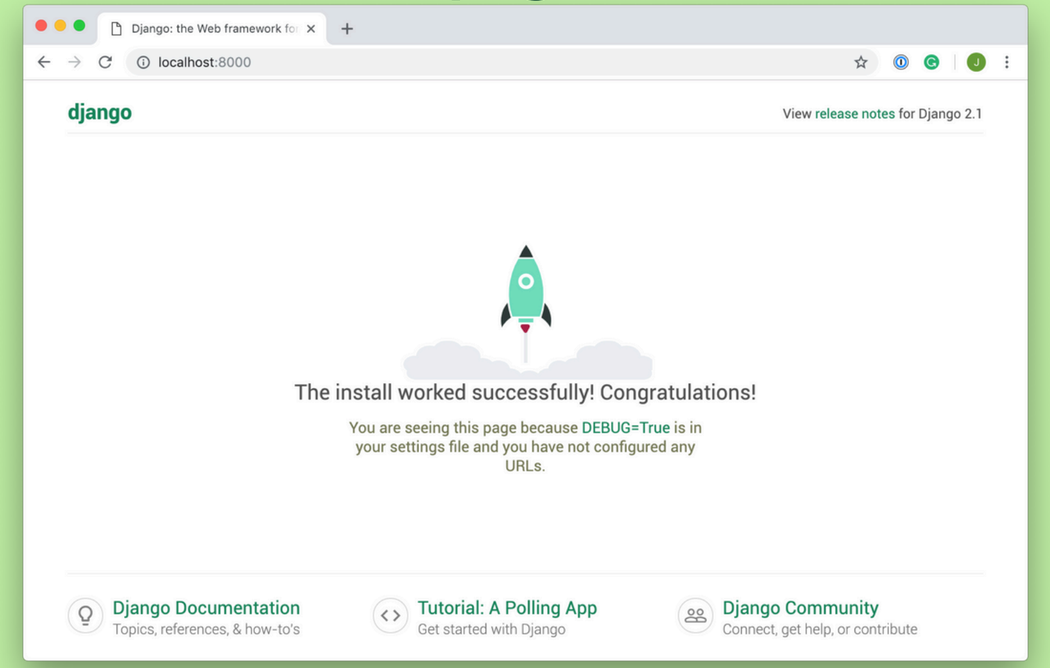
Paso 9

En el navegador, escribimos la dirección http://{{host}}:{{port}}/.

Donde host y port son los valores que nos da la terminal al correr el servidor o pueden ser las que nosotros definimos si es que lo hicimos.

Paso 10

Ya tenemos nuestra página levantada



Para más info

Repositorio que les puede servir:

https://github.com/rodrialeh01/IPC2-VJ2024/blob/main/Unidad6/Manual%20de%20Inicio%20Django.md

Listo!, Ya podemos empezar a aprender más de Django

Ejemplo

¿Preguntas?
¿Dudas?

No dudes en contactarme.

Correo electrónico

rodrialehdl@gmail.com

GitHub

rodrialeh01