Grupo 1401 Jesús Franco López
Pareja 02 Nuria Cuaresma Saturio

Proyecto de Sistemas Informáticos Práctica - 2

Django es un framework web de alto nivel que permite el desarrollo de aplicaciones seguras. Está basado en el paradigma Modelo-Vista-Controlador, más concretamente en el patrón de diseño Modelo-Vista-Template.

Antes de explicar Django, se expondrán las características de los patrones en los que se basa, para así facilitar la comprensión del servidor y comprender porque se considera un MVT y no tanto un MVC.

Paradigma MVC. Separando en sus distintas partes podemos mostrar sus características.

- Modelo se encargará del acceso a los datos de la Base de Datos, de su clasificación e información y de la relación entre ellos.
- Vista, que se encarga de cómo y qué datos mostrar, para ello necesitará los datos del Model y los presentará a través de las vistas que generarán la salida indicada en la aplicación.
- Controlador, que realiza las acciones solicitadas por la petición que el usuario manda a la aplicación, con los datos que obtiene de Modelo realizará la función generando una salida que será mostrada a través de la Vista. Se podría decir que sirve de enlace entre Modelo y Vista y se realizará en el propio framework.

Por otro lado MVT contendrá la misma función para el Modelo, variando tanto la funcionalidad de la Vista como la última de Template, en este patrón.

- La Vista en este caso es la encargada de realizar las funciones, mediante los datos de Modelo y realizando la acción solicitada por el usuario, obtendrá una salida. Será el nexo entre Modelo y Template.
- Template es la parte encargada de mostrar al usuario los datos obtenidos tras su petición, así como de la presentación mediante templates de todo lo indicado en nuestra aplicación.

Podemos ver que la arquitectura del servidor web Django se compone de un Proyecto Django al que se pueden conectar una o varias aplicaciones Django.
Dentro del proyecto Django también podemos encontrar la base de datos que creamos, el base HTML, del que se basan todo el resto de HTMLs que necesitaremos para los templates de la aplicación, y el parseador de URL del proyecto del que se fundamenta el parseador de URL específico de cada aplicación contenida en el proyecto.
En las aplicaciones tendremos los ficheros python views.py, que realiza las acciones solicitadas; models.py, que contiene las clases existentes en la aplicación, tantas como se quiera, que describen los diferentes datos a guardar en la base de datos; los HTML en el directorio templates y el parseador de URLs, que dota de direcciones válidas a cada una de las acciones descritas en el fichero views.py.
Su funcionamiento comienza cuando se realiza una petición desde Internet al servidor Web the Django. Este pasará la petición al proyecto Django, en el que se parseará la URL y se analizará. Esta información se mandará a la aplicación. Aquí tras haber sido interpretada la URL se obtiene la función a indicada por el usuario y mediante el fichero views.py se realizará. El models.py gestionará los datos introducidos en la petición y lo generado por dicha función y que guardará en la base de datos, en la clase correspondiente. Finalmente será mostrado al usuario a través de los HTML templates.

Grupo 1401 Jesús Franco López
Pareja 02 Nuria Cuaresma Saturio

Para resumir separamos la funcionalidad en sus distintas capas, al igual que se hizo en el paradigma MVT para ver con más facilidad como se corresponde con este patrón.

- Modelo: El tratamiento de los datos y comunicación con la base de datos, al igual que ambos patrones MVT y MVC. Esta comunicación se realizará en el fichero models.py de la aplicación
- Vista: Se encargará de la interpretación de las peticiones por parte del usuario y de la realización de las funciones, actuando así de nexo entre el Model y el Template. El análisis de la petición, es decir la URL, se realizará por el fichero url.py tanto o del proyecto o de la aplicación. Y a continuación de la ejecución de la acción se encarga el fichero view.py. Utiliza los datos de Model y mandará el resultado a Template, estableciendo así la conexión.
- Template: Finalmente esta capa mostrará al usuario el resultado de su petición y la presentación de nuestra aplicación. Esta funcionalidad se lleva a cabo por los distintos HTML templates que recogen los datos a mostrar al usuario.