PROYECTO DE SISTEMAS INFORMÁTICOS: PRÁCTICA 2

Pareja 4

Óscar Gómez Borzdynski y Aitor Arnaiz del Val

Arquitectura de Django

Django usa el patrón Modelo Vista Controlador (MVC) como patrón de arquitectura para el desarrollo de aplicaciones, sólo que lo modifica un poco, implementando un patrón de diseño conocido en el caso de Django como Modelo Vista Template (MVT)

A continuación señalaremos las principales características de estos dos patrones de diseño:

- **Patrón MVC:**
  - ***Modelo:*** La 'M' del patrón MVC, se encarga del acceso a los datos, y contiene toda la información sobre estos, tanto en cuanto al acceso a estos como a validación, comportamiento y relaciones entre datos.
  - ***Vista:*** Por su parte, la 'V' de este patrón se refiere, como se ha mencionado, a las Vistas, que como su nombre indica, se encarga de seleccionar qué datos mostrar y cómo hacerlo. En las vistas se trabaja con los datos de forma indirecta: se requieren los datos a los Modelos y las vistas generarán la salida tal y como nuestra aplicación lo indique.
  - ***Controlador:*** El controlador contiene la funcionalidad necesaria para efectuar las acciones solicitadas en nuestra aplicación. En general, es una capa que sirve de enlace entre Vistas y Modelos, aunque ni manipula datos de forma directa (tarea del Modelo) ni genera ningún tipo de salida (trabajo de la Vista).
- **Patrón MVT:** Como se ha señalado, Django realiza una implementación especial del patrón MVC, que se traduce en el patrón MVT que describiremos a continuación:
  - ***Modelo:*** En Django, la capa de Modelo realiza una tarea similar a la que desempeña en el patrón MVC: el acceso a la Base de Datos y, como se ha dicho, contener toda la información relacionada con estos.
  - ***Vista:*** En Django, esta capa contiene la lógica que accede al Modelo y lo delega a la Plantilla (Template) apropiado; básicamente hace de nexo o puente entre Modelos y Templates.
  - ***Template:*** Los Templates o Plantillas se encargan de la capa de presentación en Django, la cual contiene todo lo relacionado con la presentación de nuestra página, esto es, cómo mostramos las cosas sobre nuestra página web.

La razón de que Django se considere más bien una arquitectura MVT que una MVC se debe a que, en Django, la 'C' es manejada por el mismo framework, y la gran parte de la funcionalidad se produce en Modelos, Templates (plantillas) y Vistas.

En la aplicación Django, la funcionalidad de cada parte del patrón MVT la realizan diversos ficheros:

- ***Modelo:*** Como se ha dicho antes, tanto en el patrón MVC como en la reinterpretación de este que hace Django, el MVT, la capa de Modelo se encarga de la comunicación con la Base de datos y el acceso, interpretación, validación, etc de estos.

  Esta funcionalidad en Django se lleva a cabo en el fichero (como es obvio) ***models.py***, donde se implementa toda la funcionalidad relacionada con la comunicación con la ***Base de Datos*** de la aplicación Django.

- ***Vista:*** En cuanto a la capa de Vista, se ha señalado anteriormente que esta realiza la función de nexo entre las otras dos capas. En las aplicaciones Django, la funcionalidad de esta capa se encuentra implementada en los ficheros ***views.py*** (al igual que antes, esto es obvio) y ***urls.py***

  Estos dos ficheros se comunican entre sí, el ***views.py*** para definir la información a mostrar una vez recogida de la capa de Modelo, pasándosela a la capa de Templates, mientras que ***urls.py*** se encarga de definir las urls (como es lógico) en las que se mostrarán los templates en nuestro navegador.

- ***Template:*** Por último, en lo tocante a la capa de Template, o de presentación, ya se ha dicho que esta se encarga en nuestra aplicación Django de tomar las decisiones de presentación de nuestra página, sin sin embargo desempeñar las tareas del controlador, pues la mayoría de estas las lleva a cabo el propio framework, como ya se ha dicho.

  Por lo general, la funcionalidad de esta capa se lleva a cabo en los ***ficheros html templates***, una serie de ficheros de extensión ***.html***, normalmente localizados en un directorio especial de nuestro proyecto Django, el ***directorio templates/***, en el que se almacenarán los templates que recogerán los datos a mostrar al usuario, pasados por la capa de Vista, que ha recogido los datos de la Base de datos, a la que ha accedido de forma directa la capa de Modelo, haciendo que finalmente se pueda visualizar la página web.

Una vez creadas y comunicadas de forma satisfactoria las capas de **Vista**, **Modelo** y **Template** de nuestra aplicación Django, sólo queda hacer **correr el proyecto Django en un servidor web**, pudiendo acceder a nuestra web a través de internet.