UNIVERSIDAD AUTÓNOMA DE MADRID

PROYECTO DE SISTEMAS INFORMÁTICOS

(2019-2020)

PRÁCTICA 2

Alba Ramos Pedroviejo
Javier Lozano Almeda

Grupo 1363

Madrid, 22 de octubre de 2019

Presentación de Django

En la primera diapositiva podemos ver los diferentes niveles que tendrá la aplicación web desarrollada. El servidor web gestiona la comunicación en la red Internet, esto es, las peticiones y respuestas utilizando el protocolo Http (HttpRequest y HttpResponse, respectivamente). Para esta gestión va a utilizar el framework de Django, en el que hemos construido nuestro proyecto. Dentro del mismo proyecto puede haber varias aplicaciones que quieran hacer unas cosas u otras según la petición recibida. En proyecto encontramos la base de datos que usará nuestro sistema web, y en la aplicación encontramos los modelos necesarios para comunicarnos con esta base de datos.

Cuando llega la petición, el fichero de URLs del proyecto Django determinará a qué función de las vistas se debe llamar. Según la petición (url) que llegue, este fichero dirá directamente a qué vista llamar o le transferirá el control al fichero de URLs de la aplicación (en nuestro caso solo hemos creado una aplicación, pero podríamos crear varias y cada una podría hacer diferentes cosas ante la misma petición). Esto se conoce como mapeo de URLs.

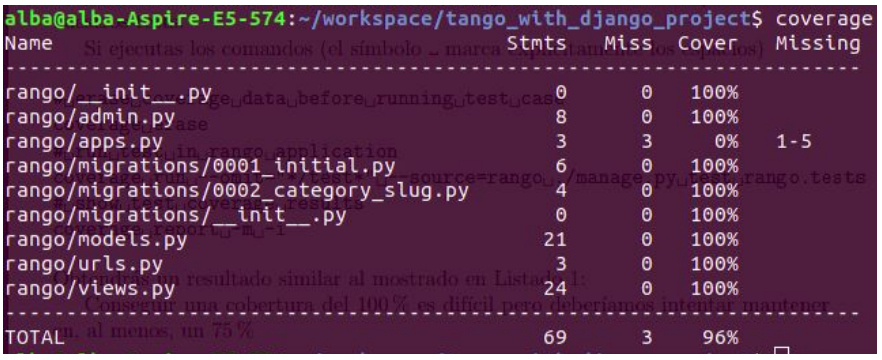
Las vistas se encargan de conectar con la base de datos para obtener toda la información necesaria para gestionar la petición a través de los modelos definidos en models.py. La base de datos, recordemos, está definida en el proyecto Django (en settings.py), y nosotros definimos las comunicaciones necesarias utilizando los modelos, sin necesidad de realizar consultas: esto lo gestionará Django por nosotros. De esta forma, podemos crear nosotros una base de datos en PostgreSQL, por ejemplo, de la que conocemos su estructura, y crear los modelos correspondientes para poder trabajar con los campos de cada una de las tablas que haya en nuestra base en el resto de la aplicación sin tener que realizar una sola consulta.

Cuando la vista tiene toda la información necesaria de la base de datos, retornará la respuesta: una página HTML (template) que será enviada al cliente. En esta página se pueden utilizar variables recibidas desde la vista para mostrar un contenido u otro según el valor de las variables.

Django es un ejemplo de arquitectura MVC. En esta arquitectura, el controlador es el elemento que respondería ante las peticiones y con el que interactúa el cliente, invocando al modelo tras la petición. El modelo gestiona el comportamiento que hay que realizar ante esa petición. Según este comportamiento, actualiza la vista para mostrar esta lógica de forma gráfica para el cliente. En Django, el modelo sería la parte de models.py, donde gestionamos la información que se muestra y se actualiza usando la base de datos. La vista serían los ficheros HTML que se retornan al cliente para que visualice esta información. Finalmente, el controlador es la parte de gestión de peticiones, es decir, los ficheros de URLs y las vistas. Estos detectan la petición del cliente (URLs) y acceden al modelo para modificarlo de acuerdo a la petición.

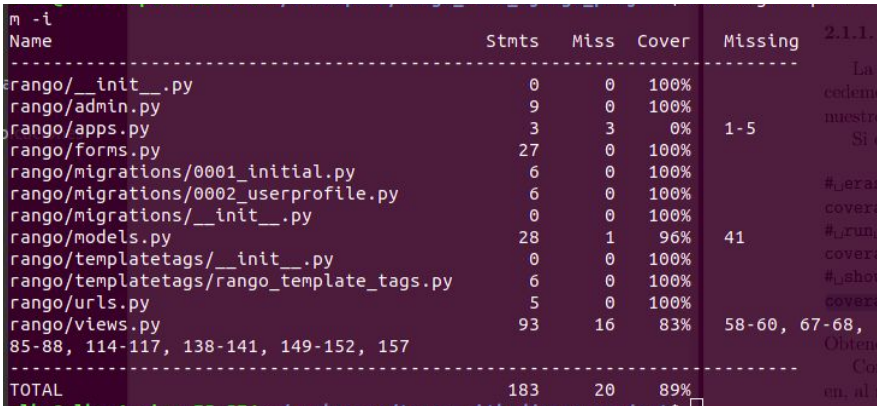
Cobertura de los tests

Cobertura durante la segunda semana:



```
alba@alba-Aspire-E5-574:~/workspace/tango_with_django_project$ coverage
Name                                 Stmts   Miss  Cover   Missing
------------------------------------------------------------------
rango/__init__.py                        0      0   100%
rango/admin.py                           8      0   100%
rango/apps.py                            3      3     0%   1-5
rango/migrations/0001_initial.py         6      0   100%
rango/migrations/0002_category_slug.py   4      0   100%
rango/migrations/__init__.py             0      0   100%
rango/models.py                         21      0   100%
rango/urls.py                            3      0   100%
rango/views.py                          24      0   100%
------------------------------------------------------------------
TOTAL                                   69      3    96%
```

Cobertura durante la última semana. La diferencia en el total, aunque está dentro del rango admisible, puede deberse a que, al añadir nuevas funcionalidades, los tests no alcanzan a probar todos los posibles caminos, incluidos los de error.



```
m -i
Name                                 Stmts   Miss  Cover   Missing
------------------------------------------------------------------
rango/__init__.py                        0      0   100%
rango/admin.py                           9      0   100%
rango/apps.py                            3      3     0%   1-5
rango/forms.py                          27      0   100%
rango/migrations/0001_initial.py         6      0   100%
rango/migrations/0002_userprofile.py     6      0   100%
rango/migrations/__init__.py             0      0   100%
rango/models.py                         28      1    96%   41
rango/templatetags/__init__.py           0      0   100%
rango/templatetags/rango_template_tags.py       6      0   100%
rango/urls.py                            5      0   100%
rango/views.py                          93     16    83%   58-60, 67-68,
85-88, 114-117, 138-141, 149-152, 157
------------------------------------------------------------------
TOTAL                                  183     20    89%
```