page précédente : Terminal

2. Qu'est-ce que Django?

Django (/ˈdʒæŋgoʊ/) est un framework web gratuit et open-source écrit en Python. Un framework web est un ensemble de composants qui vous aide à développer des sites web plus rapidement et plus facilement.

Lorsque vous créez un site web, vous avez souvent besoin de la même chose : une manière de gérer l'authentification de vos utilisateurs (créer un compte, se connecter, se déconnecter), une partie dédiée à la gestion de votre site, des formulaires, une manière de mettre en ligne des fichiers, etc.

Heureusement pour vous, d'autres personnes ont réalisé depuis longtemps que les développeurs web font face aux mêmes problèmes lors de la construction d'un nouveau site, donc ils travaillent ensemble et créent un framework (Django étant un) qui vous donne des composants prêts à l'emploi.

Les frameworks existent pour vous éviter d'avoir à réinventer la roue et aider à réduire les frais généraux lorsque vous construisez de nouveaux sites.

Pourquoi auriez-vous besoin d'un framework?

Pour comprendre ce qu'est exactement Django, nous devons regarder de plus près le serveur. La première chose est que le serveur doit savoir que vous voulez vous servir une page Web.

Imaginez une boîte aux lettres (un port) dont l'arrivée de lettres (une requête) serait surveillée. C'est le travail qu'effectue le serveur. Le serveur Web lit la lettre et renvoie une page Web en réponse. Généralement, lorsque vous voulez envoyer quelque chose, vous avez besoin de contenu. Django est un outil qui va vous aider à créer ce contenu.

Que se passe-t-il quand quelqu'un demande un site web à votre serveur?

Lorsqu'une requête arrive sur un serveur web, elle est transmise à Django dont le premier travail va être de comprendre ce qui est demandé. Il s'occupe tout d'abord de l'adresse de la page Web et essaie de savoir quoi faire. Ce travail est effectué par le routeur de Django, l'urlresolver (à savoir qu'une adresse web est appelée URL - Uniform Resource Locator - d'où le nom d'urlresolver). Ce n'est pas très intelligent - il faut renseigner une liste de modèles pour faire correspondre une URL. Django vérifie dans l'ordre les modèles, et si correspondance il y a, alors il transmet la requête à la fonction associée (appelée vue).

Afin d'y voir un peu plus clair, imaginez un facteur transportant une lettre. Il descend la rue et vérifie à chaque maison si le numéro de celle-ci correspond à celui de la lettre. Si les deux numéros correspondent, il met la lettre dans la boîte aux lettres de cette maison. C'est à peu près comme cela que fonctionne l'urlresolver !

C'est dans la fonction de la vue que les choses se passent : nous allons pouvoir jeter un œil dans la base de données pour obtenir davantage d'informations. Par exemple, peut-être que l'utilisateur

demande à changer quelque chose dans ces données ? Ce serait comme une lettre dont le contenu serait : "Merci de changer la description de mon emploi actuel". La vue va tout d'abord vérifier que l'utilisateur est bien autorisé à effectuer ce changement puis elle corrigera la description de l'emploi. Enfin, la vue générera une réponse de type "Travail terminé !" que Django pourra retourner à l'utilisateur.

Ceci n'est qu'une description très simplifiée du processus. Vous n'avez pas besoin de connaître tous les détails techniques pour le moment : cette vue d'ensemble suffira largement.

Au lieu de vous assommer avec des détails complexes, nous allons plutôt commencer à construire quelque chose avec Django et nous allons apprendre les choses importantes au fur et à mesure !

3. Installation de Django