Python - Django

Développement Python et bonnes pratiques

sommaire

Django - Présentation

- Framework web open-source pour Python
- Créé en 2005, maintenu par la Django Software Foundation
- Basé sur le principe « **DRY** » : ***Don't Repeat Yourself***
- Idéal pour développer des ***applications web*** rapidement et proprement

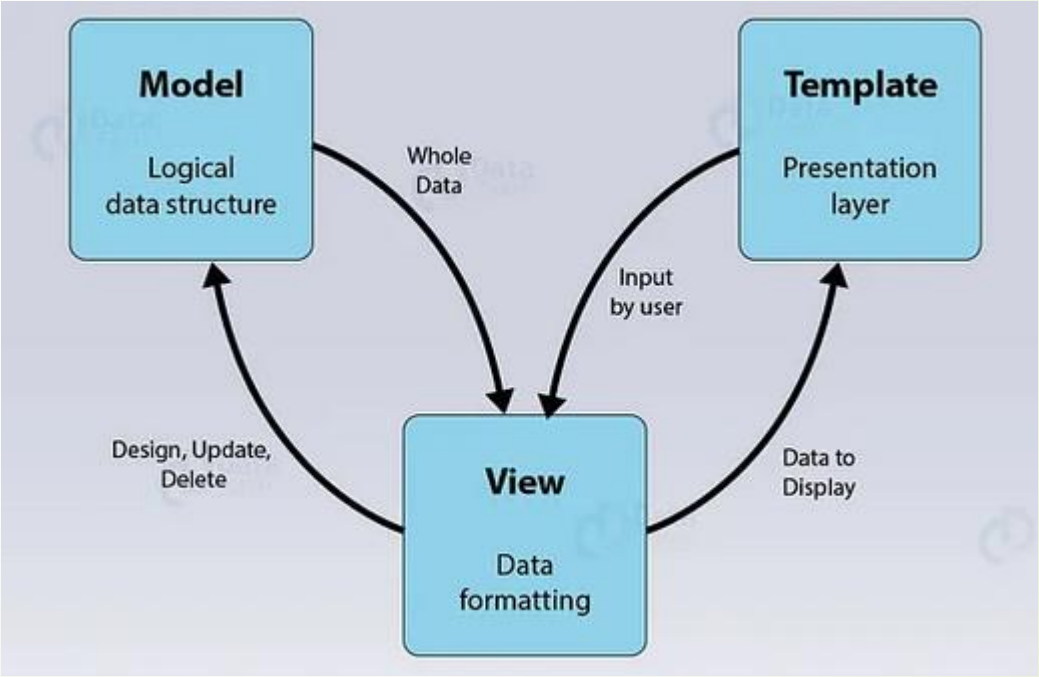
Django - Pourquoi utiliser Django ?

- Rapide à développer (RAD)
- Hautement sécurisé
- Complet (ORM, admin, routing, forms, etc.)
- Très utilisé (***Instagram***, ***Pinterest***, ***Mozilla, …***)
- Bonne documentation et grande communauté

Django - L'architecture MTV

- **M**odèle : gère les données (ORM)
- **T**emplate : interface utilisateur (HTML)
- **V**ue : logique métier
- Comparable au MVC, mais adapté à Django

Django – MTV - schéma



Django - L'architecture MTV

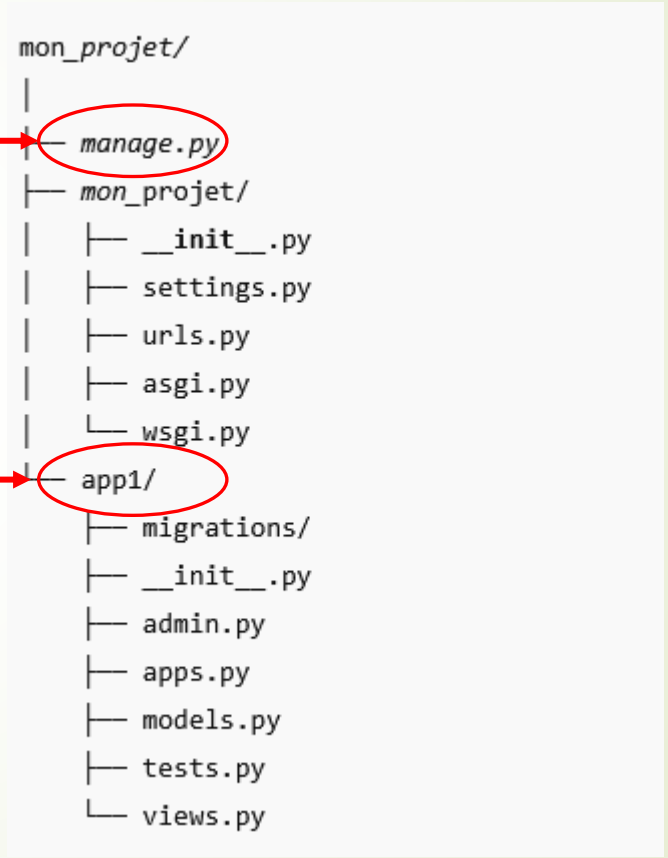
Django – Installation

- pip install django
  - Installation du framework
- django-admin startproject [projet]
  - Crée la structure du projet
- python manage.py runserver
  - Lance le serveur

Structure de projet Django

Outil en ligne de commande pour gérer le projet

Application Django (module)



Structure - répertoire principale

- Le fichier ***manage.py***
  - Sert à exécuter des commandes Django
  - Fonctionne comme une interface vers le projet
- Le répertoire ***mon_projet/***
  - ***settings.py :*** toutes les configurations du projet
  - ***urls.py :*** le routage principal (inclusion des routes d'apps)
  - ***asgi/wsgi :*** pour le déploiement (serveur ASGI/WGI)

Les composants d'une application Django

- Structure d'une application :

Contenu :

- `models.py` : structure des données (tables)
- `views.py` : logique des réponses HTTP
- `admin.py` : enregistrement des modèles dans l'interface admin
- `apps.py` : configuration de l'app
- `tests.py` : tests unitaires
- `migrations/` : scripts de migration de la base de données

Django – ORM Django

- Django possède son propre ORM
  - On définira les entités (models) dans le répertoire [module]***.models.py***

Django – Vues et routage URL

- Structure du projet
- Fonction ou classe Python qui traite une requête
- Mapping via urls.py
- Séparation claire entre logique et présentation

Django – Système de templates

- Structure du projet
- HTML avec balises de template ({{ variable }}, {% for %})
- Séparation logique / affichage
- Héritage de templates : DRY encore appliqué

Django – Admin et autres fonctionnalités

- Interface d'administration automatique avec admin.py
- Authentification utilisateur
- Formulaires dynamiques
- Internationalisation, middleware, tests, sécurité…

Conclusion et ressources

- Django = complet, robuste, prêt pour la production
- Idéal pour les startups comme les grandes entreprises
- Ressources :
  - https://www.djangoproject.com
  - https://docs.djangoproject.com/fr/
  - https://realpython.com/tutorials/django/

Questions/Réponses

Question ?