



Introduction Framework Django

UP Web

AU: 2025/2026

Plan



1.Introduction à Django

2.Architecture

3.Etapes d'installation

4.Création d'un projet

5.Création d'une application

Introduction à Django



Qu'est-ce que Django?

- Un « Framework » python open source.
- Conçu pour faciliter le développement des applications web.
- Communauté importante, un support disponible et plusieurs options de déploiement.

Principes de Django

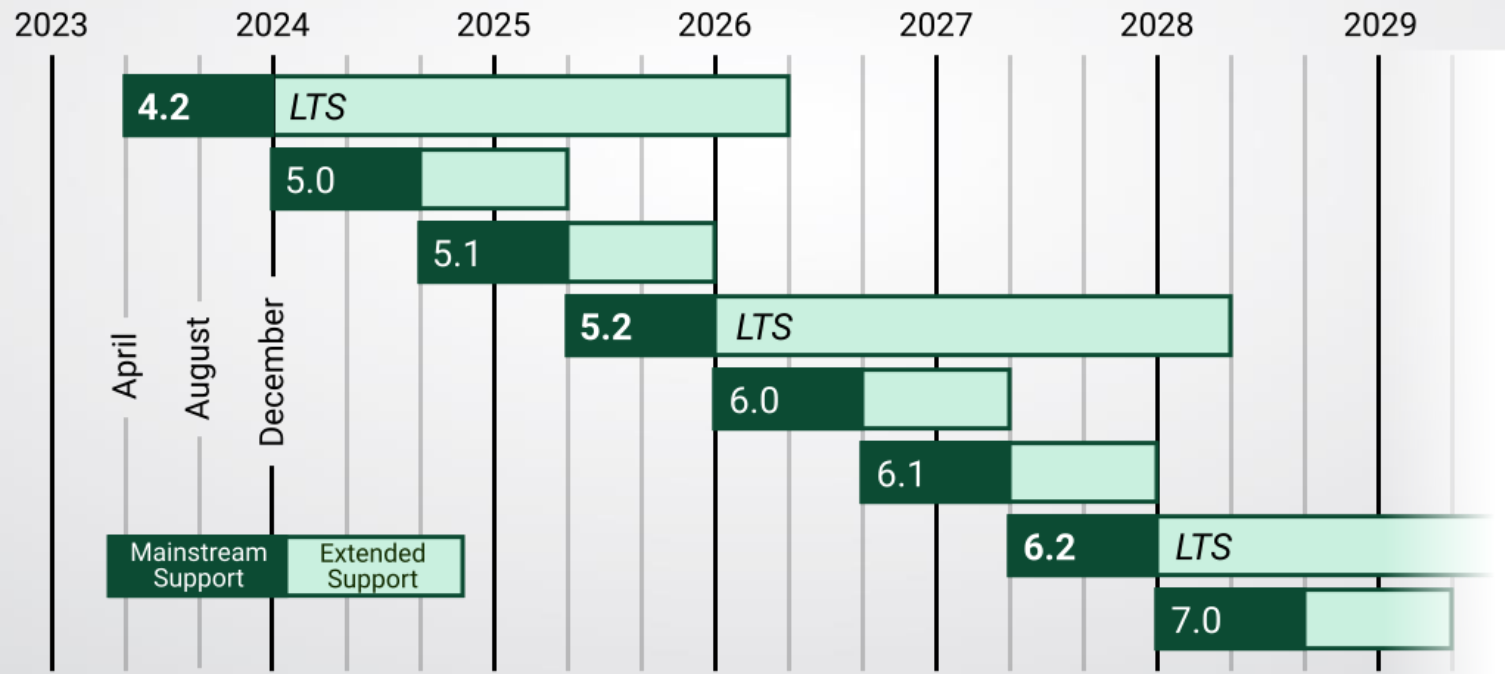
- **Réutilisabilité des Composants** : Django suit le principe **DRY** (Don't Repeat Yourself), qui encourage la réutilisation du code et la réduction de la duplication.
- **Batteries incluses** : Django fournit de nombreuses fonctionnalités intégrées, telles que des systèmes d'authentification, des connexions à des bases de données, et des opérations CRUD (Create, Read, Update, Delete).
- **Orienté Base de Données** : Django est particulièrement adapté pour les sites web basés sur des bases de données. Il utilise un ORM (Object-Relational Mapping) qui simplifie l'interaction avec la base de données.

Introduction à Django



Historique & Versions

- Django a été développé à l'origine entre 2003 et 2005 par une équipe de développeurs web travaillant pour le journal en ligne Lawrence Journal-World.
- Django a été publié en open source en juillet 2005.



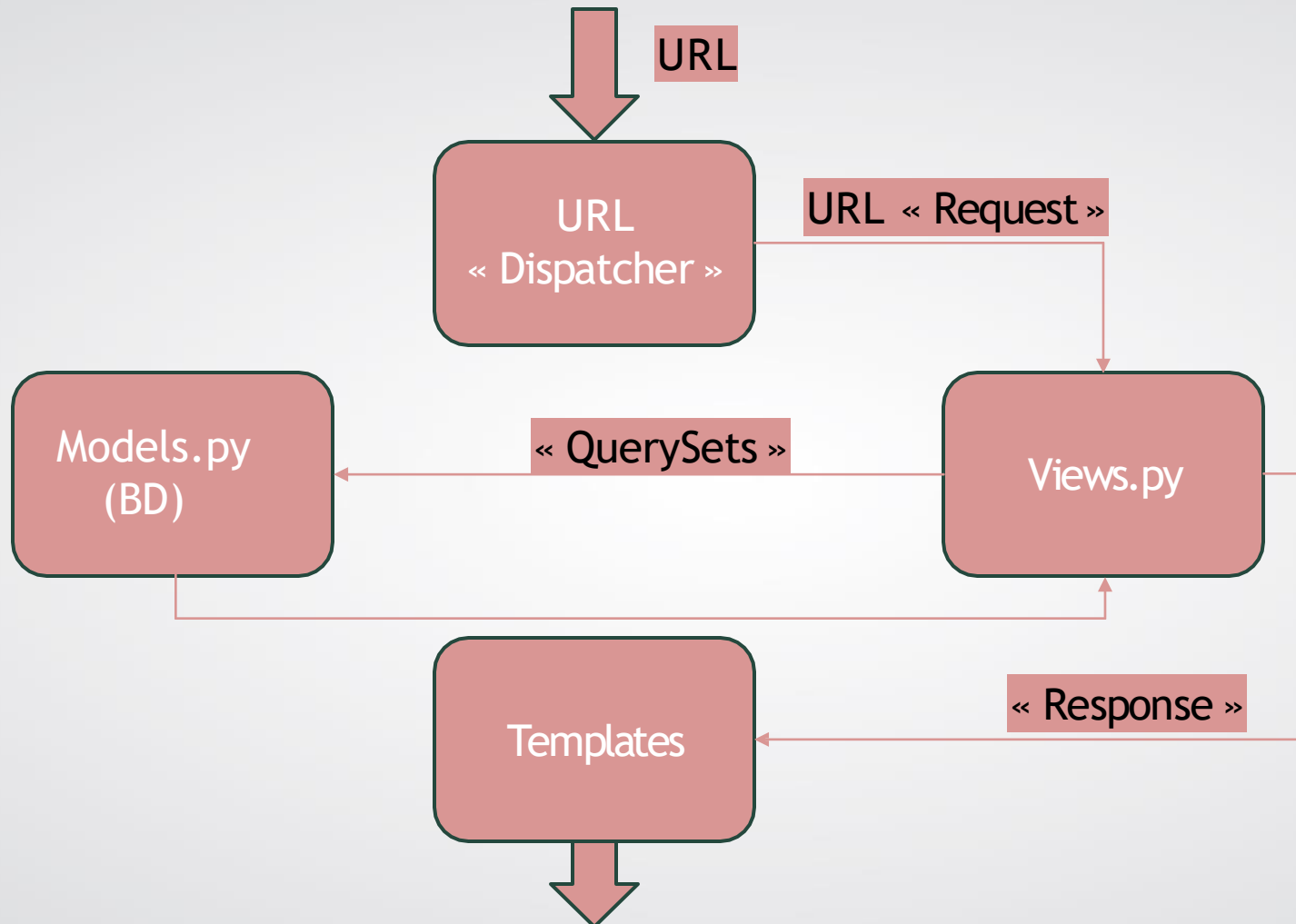
Introduction à Django



Quelques exemples d'applications qui utilisent Django



Architecture MVT



Etapes d'installation



Environnements virtuels

- Un environnement virtuel en Python est un outil qui permet de créer un espace isolé pour le développement de projets.
- Cela aide à gérer les dépendances de manière efficace et à éviter les conflits entre les différentes versions de bibliothèques utilisées dans divers projets.
- Il existe plusieurs outils pour créer des environnements virtuels:
 - Avec 'venv' (intégré à partir de Python 3.3)
 1. Créer un environnement: ***python3 -m venv nom_environnement***
 - Avec 'Virtualenv'
 1. Installer le paquetage: ***pip install virtualenv***
 2. Créer un environnement: ***virtualenv nom_environnement***

Etales d'installation



Environnements virtuels

- Pour activer l'environnement virtuel, venv ou virtualenv:

- Sur Windows:

nom_environnement\Scripts\activate

```
C:\Users\oumei_fvs20aj\Desktop\django2526>env\Scripts\activate
(env) C:\Users\oumei_fvs20aj\Desktop\django2526>
```

- Sur macOS / Linux:

source nom_environnement/bin/activate

```
linda in ~/Documents/Esprit/2025-2026/PFW/test_django
→ source .venv/bin/activate
linda in ~/Documents/Esprit/2025-2026/PFW/test_django
→ (.venv) |
```

- Pour quitter l'environnement, utilisez: **deactivate**
- Un environnement virtuel contient plusieurs dossiers et fichiers :
 - bin ou Scripts : Contient l'interpréteur Python et les exécutables nécessaires (comme pip).
 - lib ou Lib : Contient les bibliothèques installées dans l'environnement.
 - pyvenv.cfg : Fichier de configuration qui contient des informations sur l'environnement virtuel.

Etapes d'installation

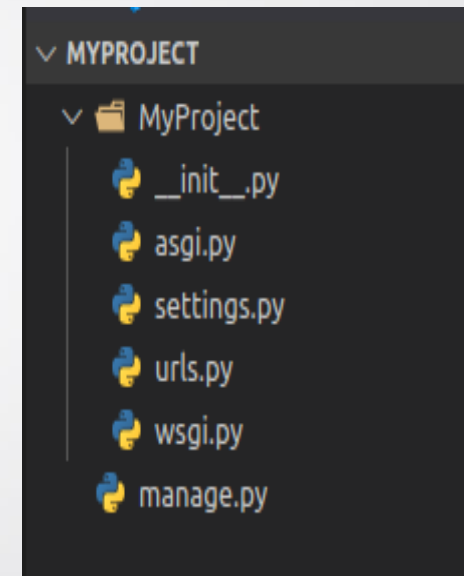


- Pour installer django, utilisez la commande suivante:
 - ***pip install django=="5.2.*"***
 - Pour vérifier que django a bien été installer, utilisez la commande suivante:
 - ***django-admin --version***
- Ou bien
- ***python3 -m django --version***

Création d'un projet



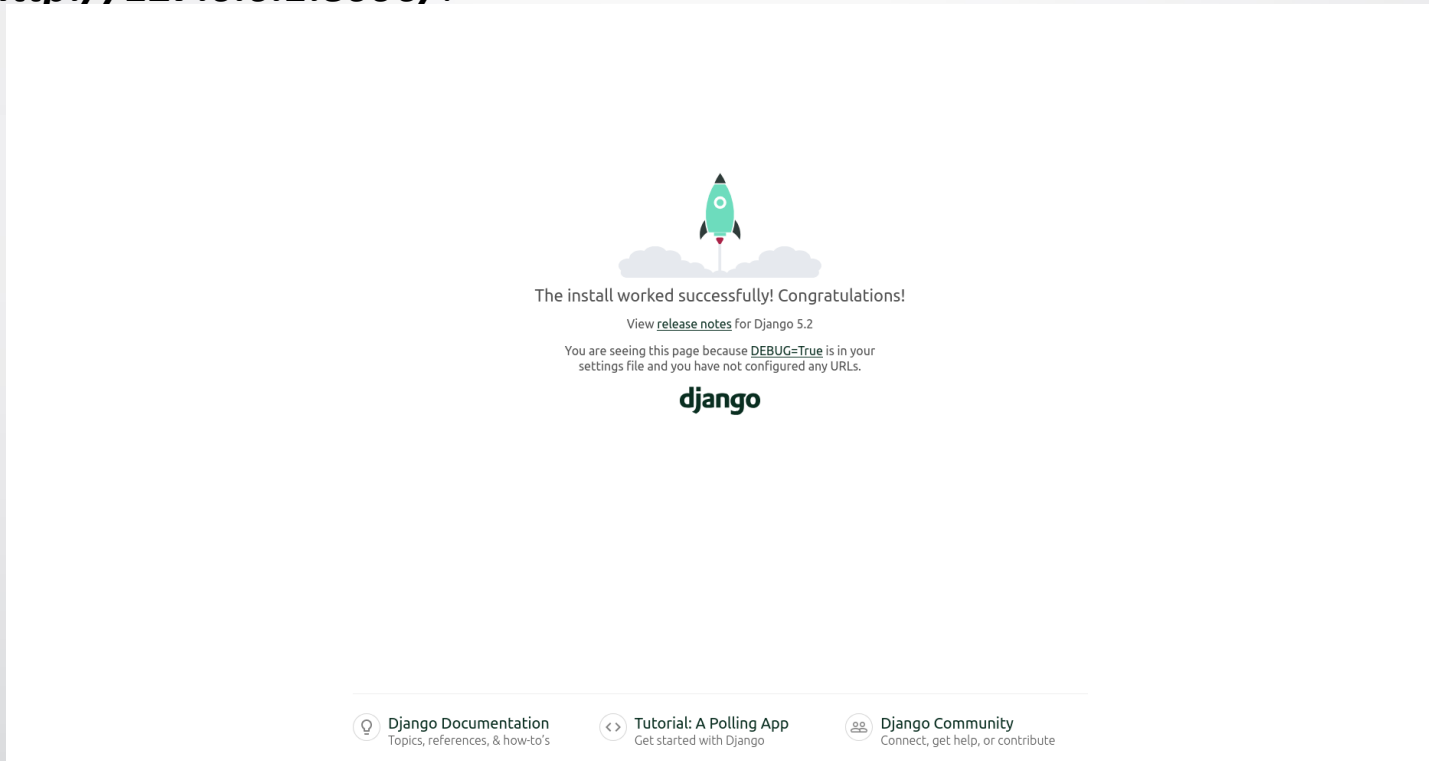
- Pour créer un projet Django, utilisez la commande suivante:
 - ***django-admin startproject nom_du_projet***
- Lors de la création d'un projet Django, la structure de fichiers est généralement la suivante :
 - ***manage.py*** : Un script utilitaire qui permet d'interagir avec le projet Django.
 - ***MyProject/*** (répertoire principal du projet) : Ce répertoire contient les fichiers de configuration du projet.
 - ***__init__.py*** : Indique que ce répertoire doit être traité comme un package Python.
 - ***settings.py*** : Contient les configurations de votre projet.
 - ***urls.py*** : Définit les routes URL de votre projet.
 - ***asgi.py* / *wsgi.py*** : Fichier de configuration pour les applications, utilisé pour le déploiement d'applications asynchrones et synchrones respectivement.



Création d'un projet



- Pour lancer le serveur Django, utilisez les commandes suivantes:
 1. `cd nom_du_projet`
 2. `python manage.py runserver` ou bien `python manage.py runserver num_port`
- Vous pouvez accéder à votre application Django en ouvrant un navigateur et en allant à l'adresse **`http://127.0.0.1:8000/`**.



Création d'une application



- Pour créer une application, utilisez la commande suivante:
 - ***python manage.py startapp nom_de_l_application***
- Lorsque vous créez une nouvelle application, Django génère la structure de fichiers suivante :
 - **migrations/** : Contient les fichiers de migration pour gérer les modifications de la base de données.
 - **admin.py** : Permet de configurer les modèles pour l'interface d'administration de Django.
 - **apps.py** : Contient la configuration de l'application.
 - **models.py** : Définit les modèles de données de l'application, qui représentent la structure de la base de données.
 - **tests.py** : Contient les tests unitaires pour l'application.
 - **views.py** : Contient la logique des vues, qui traite les requêtes et renvoie les réponses.

