Cours Django

Pourquoi apprendre Django ?

Avant de plonger dans le cours, il est important de comprendre **pourquoi Django est un excellent choix pour apprendre le développement web** et comment il peut bénéficier à votre parcours de développeur.

1. Django, un framework complet et robuste

- **Un framework "batteries incluses" :** Django fournit tout ce dont vous avez besoin pour développer une application web, sans avoir à réinventer la roue. Par exemple :
    - Gestion des utilisateurs (authentification, permissions).
    - Gestion de la base de données avec un ORM puissant.
    - Gestion des formulaires, des modèles, et des vues.
    - Un panneau d'administration prêt à l'emploi.
- **Pourquoi c'est utile ?**
    - En tant que débutants, vous pouvez vous concentrer sur les concepts essentiels du développement web sans être submergés par la complexité technique.

2. Un outil utilisé dans le monde réel

- **Des entreprises de renom utilisent Django :**
    - **Instagram** : Une des plus grandes plateformes de médias sociaux.
    - **Pinterest** : Une application populaire pour la découverte d'idées.
    - **Spotify** : Django est utilisé pour gérer certaines parties de l'infrastructure de Spotify.
    - **Mozilla** : Le navigateur Firefox a utilisé Django pour certains de ses outils.
- **Pourquoi c'est important ?**
    - Apprendre Django, c'est acquérir une compétence recherchée sur le marché du travail.

3. Django est adapté aux projets de toutes tailles

- **Projets simples :** Vous pouvez créer rapidement des blogs, des portfolios ou des sites vitrines.

- **Projets complexes :** Django peut gérer des sites e-commerce, des plateformes de médias sociaux ou des systèmes d'information pour les entreprises.

- **Pourquoi c'est pratique pour vous ?**

    - Avec Django, vous pouvez commencer par des projets simples et évoluer progressivement vers des applications complexes.

4. Un écosystème Python accessible et puissant

- Django utilise **Python**, l'un des langages de programmation les plus populaires et les plus faciles à apprendre.
- Pourquoi Python + Django ?
    - Python est lisible et intuitif, idéal pour les débutants.
    - La communauté Python est immense, donc beaucoup de ressources sont disponibles pour vous aider à apprendre.

5. Une grande communauté et de nombreuses ressources

- **Une communauté active :** Django est maintenu par une communauté internationale de développeurs qui contribuent à son amélioration continue.
- **Documentation de qualité :** La documentation officielle de Django est l'une des meilleures parmi les frameworks web.
- **Support facile :** Si vous avez des questions, vous trouverez des réponses sur des forums comme Stack Overflow.

6. Django vous prépare à des concepts avancés

- Avec Django, vous apprendrez des notions fondamentales du développement web, comme :
    - La séparation entre le **backend** (logique métier) et le **frontend** (interface utilisateur).
    - La gestion des bases de données avec des outils modernes comme les ORM.
    - La sécurisation des applications web (protection contre les attaques courantes comme les injections SQL ou le Cross-Site Scripting).
- Ces concepts sont universels et vous seront utiles même si vous décidez plus tard d'explorer d'autres technologies.

7. L'autonomisation par la création

- Django vous permet de transformer vos idées en réalité :
    - Vous voulez créer une boutique en ligne pour vendre vos produits ? Django peut le faire.
    - Vous voulez lancer un réseau social pour votre communauté ? Django est là.
    - Vous voulez automatiser certaines tâches de votre entreprise ? Django vous facilite la vie.

Conclusion : Pourquoi ce cours est important

- Apprendre Django, c'est apprendre à résoudre des problèmes réels grâce à un outil puissant et reconnu.
- Ce cours vous donne une double compétence :
    1. **Technique**, en maîtrisant Django et Python.
    2. **Créative**, en vous permettant de développer vos propres projets web.

TP 1 : Installation et création d'un projet Django

Objectifs

1. Installer Django sur leur environnement.
2. Comprendre la structure d'un projet Django.
3. Créer un projet Django et le lancer pour vérifier que tout fonctionne correctement.

Partie 1 : Installation de Django

1. Préparer l'environnement de travail

Installer Python

- Django nécessite Python pour fonctionner. Assurez-vous d'avoir Python 3.8 ou une version ultérieure.

Pour vérifier si Python est installé, exécutez la commande suivante dans un terminal ou une invite de commande :
python --version
Ou, selon votre système, utilisez :
python3 --version

-

Créer un environnement virtuel

- **Qu'est-ce qu'un environnement virtuel ?**
    - Un environnement virtuel est une copie isolée de Python qui permet d'installer des dépendances spécifiques à un projet sans affecter le reste du système.
- **Commandes pour créer et activer un environnement virtuel :**
    - Créer l'environnement virtuel dans un dossier nommé `env` :

      python -m venv env

- Activer l'environnement virtuel :
  - Sous Windows :
    env\Scripts\activate
  - Sous Mac/Linux :
    source env/bin/activate

- Une fois activé, vous verrez un préfixe `(env)` ou similaire dans votre terminal, indiquant que l'environnement est actif.

2. Installer Django

- Utilisez `pip`, le gestionnaire de paquets Python, pour installer Django :
  pip install django

- Une fois installé, vérifiez la version de Django pour confirmer :
  python -m django --version

Partie 2 : Création d'un projet Django

1. Commande pour créer un projet

- La commande suivante crée un projet Django nommé `monpremierprojet` :
  django-admin startproject monpremierprojet

- Une fois exécutée, un dossier `monpremierprojet` est créé dans le répertoire courant.

2. Explorer la structure du projet

- Naviguez dans le dossier `monpremierprojet` :
  cd monpremierprojet

- **Structure du projet :**
  - `manage.py` : Un script pour exécuter des commandes liées au projet, comme démarrer un serveur ou créer des applications.
  - `monpremierprojet/` :
    - `__init__.py` : Fichier nécessaire pour traiter le dossier comme un package Python.
    - `settings.py` : Contient la configuration globale du projet (base de données, applications installées, etc.).
    - `urls.py` : Définit les routes (URL) du projet.
    - `asgi.py` et `wsgi.py` : Utilisés pour déployer le projet en production.

3. Concept : Projet vs Application

- Un **projet** Django est une configuration globale qui peut contenir plusieurs **applications**.
- Une **application** est une fonctionnalité ou un module spécifique de votre projet (exemple : blog, e-commerce).

Partie 3 : Lancer le serveur de développement

1. Commande pour démarrer le serveur

- Exécutez cette commande dans le dossier contenant `manage.py` :
  python manage.py runserver

- Cela lance un serveur local accessible à l'adresse suivante dans un navigateur :
  http://127.0.0.1:8000/
-

2. Résultat attendu

- Une page par défaut de Django s'affiche avec le message : "The install worked successfully! Congratulations!"

3. Concept : Serveur de développement

- Ce serveur est destiné uniquement au développement. Il est simple à utiliser et permet de voir les changements en temps réel, mais il n'est pas optimisé pour la production.

Partie 4 : Créer une page personnalisée

1. Modifier le fichier

Ouvrez le fichier `urls.py` dans un éditeur de code. Il ressemble à ceci par défaut :

```
 from django.contrib import admin
from django.urls import path

urlpatterns = [
    path('admin/', admin.site.urls),
]
```

- Ajoutez une fonction pour afficher une page d’accueil personnalisée :

  ```
  from django.http import HttpResponse

  def home(request):
      return HttpResponse("Bienvenue sur mon premier projet Django !")

  urlpatterns = [
      path('', home),  # Route pour la page d'accueil
      path('admin/', admin.site.urls),
      ]
  ```

-

2. Recharger le serveur

- Si le serveur de développement est en cours d’exécution, il détectera automatiquement les modifications.

- Sinon, relancez-le avec :
  `python manage.py runserver`

3. Résultat attendu

- Accédez à `http://127.0.0.1:8000/` et voyez le message : **"Bienvenue sur mon premier projet Django !"**