Fiche du cours

65 h

Titre :

Web250 - Web Flask & Django Fullstack

Description :

Construire le même mini-produit en Flask et Django pour comprendre à la fois les fondamentaux web (HTTP, routing, templates, formulaires, auth) et les différences d'architecture (SQLAlchemy vs Django ORM, Blueprints vs Apps/DRF). On code chaque brique dans les deux frameworks, avec HTMX (interactions légères) et Tailwind (UI rapide). À la fin, vous saurez choisir quand utiliser Flask ou Django, et migrer un MVP de l'un à l'autre.

Objectifs :

- Expliquer le cycle requête ↔ réponse et le routing dans Flask et Django.*
- Rendre des pages avec Jinja (Flask) et Django Templates, intégrer HTMX.*
- Modéliser les données avec SQLAlchemy (Flask) et Django ORM ; gérer les migrations.*
- Gérer formulaires & validation (Flask-WTF vs Django Forms/ModelForm).*
- Mettre en place auth (sessions) et API REST (Flask Blueprints/RESTX vs DRF).*
- Écrire des tests (pytest), conteneuriser (Docker) et configurer une CI simple.

Chapitres :

1. Panorama & Setup : environnements (venv/poetry), structure projet Flask & Django, routing de base, vues simples.*
2. Templates & HTMX : Jinja (Flask) vs Django Templates, layouts/partials, hx-get/post, hx-swap (dans les deux).*
3. Modèles & Migrations : SQLAlchemy + Alembic (Flask) ; Django ORM + migrations ; Django Admin ; (aperçu Flask-Admin).*
4. Requêtes & Relations : filtres, jointures/select_related/prefetch_related, pagination simple dans les deux.*
5. Formulaires & Validation : Flask-WTF/CSRF/messages vs Django Forms/ModelForm ; UX et erreurs.*
6. Auth (sessions) : Flask-Login (rôles simples) vs Django Auth (permissions/middleware utiles).*
7. API REST : Blueprints/Flask-RESTX (+ schémas/erreurs/pagination) vs DRF (serializers, viewsets, routers).*

8. Fichiers & Images : uploads sécurisés, static/media, miniatures ; CORS/CSRF—rappels dans les deux.*

9. Tests, CI & Docker : pytest (fixtures/factories), structure de tests pour 2 apps, GitHub Actions (lint+tests), Dockerfile & compose (Postgres).*

10. Capstone “Double-stack” : mêmes features livrées dans Flask et dans Django (CRUD + auth + API + HTMX + upload) + doc comparative (forces/faiblesses, coût).

À la fin :

Vous saurez bâtir un MVP en Flask et en Django, maîtriser les équivalents clés (routing, templates, ORM, formulaires, auth, API), tester & dockeriser deux apps, et argumenter le choix du framework selon le contexte (vitesse/contrôle avec Flask vs product-grade out-of-the-box avec Django). Ensuite, vous pourrez suivre des cours séparés pour aller plus loin dans chaque stack (DRF, Celery, sécurité avancée, déploiement).