# Travaux Pratiques : Développement d'une Application Multi-IA avec Flask et Docker

## 1. Introduction

Ce TP a pour objectif de guider les étudiants dans la création d’une application web permettant de comparer les réponses de plusieurs modèles d’intelligence artificielle (OpenAI, Claude, Groq).  
Le projet se base sur Flask pour le backend, Tailwind CSS pour le frontend, et utilise Docker pour la containerisation. Le TP couvrira également la gestion des dépendances avec uv, l’intégration d’APIs externes, ainsi que la mise en œuvre d’une interface moderne avec rendu markdown.

## 2. Prérequis

Avant de commencer, assurez-vous d’avoir installé :  
- Python 3.12+  
- Docker et Docker Compose  
- Le gestionnaire de dépendances uv  
- Une clé API valide pour au moins un fournisseur IA (OpenAI, Claude, ou Groq)

## 3. Clonage et préparation du projet

Cloner le dépôt GitHub :  
 git clone https://github.com/MamadouBousso/promptmulti\_ia\_docker.git  
 cd promptmulti\_ia\_docker  
Rendre le script exécutable :  
 chmod +x docker-build.sh

## 4. Structure du projet

Le projet est structuré suivant le pattern MVC :  
- src/application : logique métier  
- src/domaine : entités métier  
- src/infrastructure : communication avec les APIs IA  
- templates/ : les vues HTML  
- static/ : les ressources CSS/JS (incluant Tailwind et marked.js)  
- app.py : point d’entrée Flask

## 5. Création de l’environnement local

Créer un environnement virtuel avec uv :  
 uv venv .venv  
 source .venv/bin/activate  
Installer les dépendances :  
 uv pip install flask openai anthropic groq python-dotenv

## 6. Variables d’environnement

Créer un fichier `.env` à la racine du projet avec le contenu suivant :  
 OPENAI\_API\_KEY=sk-votre-cle  
 ANTHROPIC\_API\_KEY=sk-votre-cle  
 GROQ\_API\_KEY=sk-votre-cle

## 7. Lancement de l’application

Démarrer le serveur Flask :  
 python app.py  
Accéder à l'application via http://localhost:8000

## 8. Interface Utilisateur

L’interface web permet :  
- La sélection du fournisseur IA  
- La saisie de prompts  
- La visualisation comparative des réponses des 3 IA  
- Un rendu markdown grâce à marked.js

## 9. Containerisation avec Docker

Construire l’image :  
 docker build -t promptmulti\_ia\_docker .  
Lancer le conteneur :  
 docker run -d -p 8000:8000 --env-file .env promptmulti\_ia\_docker

## 10. Travaux pratiques recommandés

- Ajouter un fournisseur IA supplémentaire  
- Ajouter un historique de conversation côté utilisateur  
- Afficher les temps de réponse de chaque API  
- Ajouter des tests unitaires dans le dossier test/