V. Environnement de développement et de déploiement avec Docker

01. Pourquoi utiliser Docker ?

- Problèmes classiques en développement : “Ça marche chez moi mais pas chez toi”
- Docker permet de **containeriser** une application avec toutes ses dépendances
- Avantages :
  - **Portabilité** : le container tourne partout de la même façon
  - **Reproductibilité** : même environnement pour toute l’équipe
  - **Isolation** : chaque service dans son propre container
  - **Facilité de déploiement** (sur un serveur, dans le cloud…)

02. Concepts clés de Docker

- **Image** : version figée d’un environnement (code + dépendances)
- **Container** : instance d’une image en cours d’exécution
- **Dockerfile** : recette pour construire une image
- **Volumes** : stockage persistant (ex : base de données)
- **Réseaux Docker** : communication entre containers
- **Variables d’environnement** : configuration externe au code

03. Création d'image personnalisée avec Dockerfile

- Écrire un `Dockerfile` pour une application Python
- Instructions de base : `FROM`, `COPY`, `RUN`, `CMD`
- Exemples d’instructions :
  - `FROM python:3.11-slim` : base légère
  - `COPY . /app` : copier le code dans l’image
  - `RUN pip install -r requirements.txt` : installer les dépendances
  - `CMD ["python", "main.py"]` : commande à exécuter au démarrage

04. Orchestration avec `docker-compose`

- Définir plusieurs services (API + DB) dans un fichier unique (`docker-compose.yml`)
- Lancer tous les services d’un coup (`docker-compose up`)
- Configuration des ports, réseaux, dépendances entre services

05. Exécuter et tester l'application

- Lancer l’application avec `docker-compose up`
- Accéder à l’API via http://localhost:8000/
- Lancer les tests avec `docker-compose run --rm api pytest`
- Consulter les logs avec `docker-compose logs -f`
- Arrêter et supprimer les containers avec `docker-compose down -v`

06. Bonnes pratiques Docker

- Séparer les environnements (dev/test/prod)
- Utiliser un `.dockerignore` (comme `.gitignore`)
- Ne jamais mettre de secrets dans les images
- Utiliser des images officielles ou vérifiées
- Construire des images légères (multistage build, Alpine...)