TP 2-2 : mini API REST

Etapes et prompts (FastAPI en Python, à titre d'exemple)

- **Structure:**
  - **Prompt:** # Générer la structure du projet avec les dossiers src, tests, docs
- **API de base:**
  - **Prompt:** # Créer une API REST avec FastAPI pour gérer des tâches (CRUD)
  - **Prompt:** # Définir un modèle Pydantic Task(id: int, title: str, done: bool=False)
  - **Prompt:** # Implémenter des routes: GET /tasks, POST /tasks, PATCH /tasks/{id}/toggle, DELETE /tasks/{id}
- **Doc auto-générée:**
  - **Action:** Vérifier /docs et /redoc (OpenAPI auto).
  - **Prompt:** # Ajouter des docstrings aux routes et exemples de requêtes
- **Tests:**
  - **Prompt:** # Générer des tests unitaires pour chaque route avec pytest et TestClient
- **Adaptation aux conventions:**
  - **Prompt:** # Adapter le code aux conventions de l'entreprise (naming, logs, erreurs)

Livrables attendus

- Code source structuré: fichiers, dossiers, modules clairs.
- Fonctionnalités opérationnelles: CLI ou API REST avec CRUD complet.
- Tests unitaires: ciblés, lisibles, avec mocks si nécessaire.
- Documentation: README + docstrings, usage reproductible, conseils de prompts.
- Refactoring: améliorations justifiées (types, erreurs, séparation des responsabilités).