

TP : Création d'une API FastAPI avec SQLite

Ammar Djebabla

5 novembre 2025

Objectif du TP

Ce TP a pour but de construire une petite API REST pour gérer une liste de tâches (*Todo List*) en utilisant **FastAPI** et une base de données **SQLite**. À la fin, vous serez capable de :

- Créer un projet FastAPI structuré.
- Connecter une base de données SQLite avec SQLAlchemy.
- Implémenter des endpoints CRUD.
- Tester vos endpoints via le navigateur ou `curl`.

1 Préparation du projet

Création du dossier et installation

```
mkdir fastapi_todo
cd fastapi_todo
python -m venv venv
source venv/bin/activate      # sous Linux/macOS
venv\Scripts\activate         # sous Windows

pip install fastapi uvicorn sqlalchemy
```

Structure du projet

```
fastapi_todo/

main.py
database.py
models.py
```

2 Fichier database.py

```
from sqlalchemy import create_engine
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker

SQLALCHEMY_DATABASE_URL = "sqlite:///./todo.db"

engine = create_engine(
    SQLALCHEMY_DATABASE_URL, connect_args={"check_same_thread": False}
)
SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=
    ↪ engine)
Base = declarative_base()
```

3 Fichier models.py

```
from sqlalchemy import Column, Integer, String, Boolean
from database import Base

class Todo(Base):
    __tablename__ = "todos"

    id = Column(Integer, primary_key=True, index=True)
    title = Column(String, index=True)
    description = Column(String, nullable=True)
    completed = Column(Boolean, default=False)
```

4 Fichier main.py

```
from fastapi import FastAPI, Depends, HTTPException
from sqlalchemy.orm import Session
from database import Base, engine, SessionLocal
from models import Todo

app = FastAPI(title="FastAPI Todo Challenge")

# Cr ation automatique de la base
Base.metadata.create_all(bind=engine)

# D pendance de session
def get_db():
    db = SessionLocal()
    try:
        yield db
    finally:
        db.close()

@app.get("/")
def home():
    return {"message": "Bienvenue dans le TP FastAPI - Todo API"}
```

```

# 1 Lire toutes les tâches
@app.get("/todos")
def read_todos(db: Session = Depends(get_db)):
    return db.query(Todo).all()

# 2 Créer une nouvelle tâche
@app.post("/todos")
def create_todo(title: str, description: str = "", db: Session = Depends(
    ↪ get_db)):
    todo = Todo(title=title, description=description)
    db.add(todo)
    db.commit()
    db.refresh(todo)
    return todo

# 3 Mettre à jour une tâche
@app.put("/todos/{todo_id}")
def update_todo(todo_id: int, completed: bool, db: Session = Depends(
    ↪ get_db)):
    todo = db.query(Todo).filter(Todo.id == todo_id).first()
    if not todo:
        raise HTTPException(status_code=404, detail="Tâche non trouvée"
            ↪ )
    todo.completed = completed
    db.commit()
    db.refresh(todo)
    return todo

# 4 Supprimer une tâche
@app.delete("/todos/{todo_id}")
def delete_todo(todo_id: int, db: Session = Depends(get_db)):
    todo = db.query(Todo).filter(Todo.id == todo_id).first()
    if not todo:
        raise HTTPException(status_code=404, detail="Tâche non trouvée"
            ↪ )
    db.delete(todo)
    db.commit()
    return {"message": "Tâche supprimée"}

```

5 Lancement du serveur

uvicorn main:app --reload

L'API sera accessible sur : <http://127.0.0.1:8000> Et la documentation interactive sur : <http://127.0.0.1:8000/docs>

6 Tests des endpoints

Créer une tâche

```
curl -X POST "http://127.0.0.1:8000/todos?title=Apprendre_FastAPI&descripti
```

Lire toutes les tâches

```
curl "http://127.0.0.1:8000/todos"
```

Mettre à jour une tâche

```
curl -X PUT "http://127.0.0.1:8000/todos/1?completed=true"
```

Supprimer une tâche

```
curl -X DELETE "http://127.0.0.1:8000/todos/1"
```

7 Challenge final

Ajoutez un nouvel endpoint pour afficher uniquement les tâches terminées :

```
@app.get("/todos/completed")
def read_completed_todos(db: Session = Depends(get_db)):
    return db.query(Todo).filter(Todo.completed == True).all()
```

Testez votre nouveau endpoint :

```
curl "http://127.0.0.1:8000/todos/completed"
```

Objectif : filtrer les tâches selon leur état d'avancement.