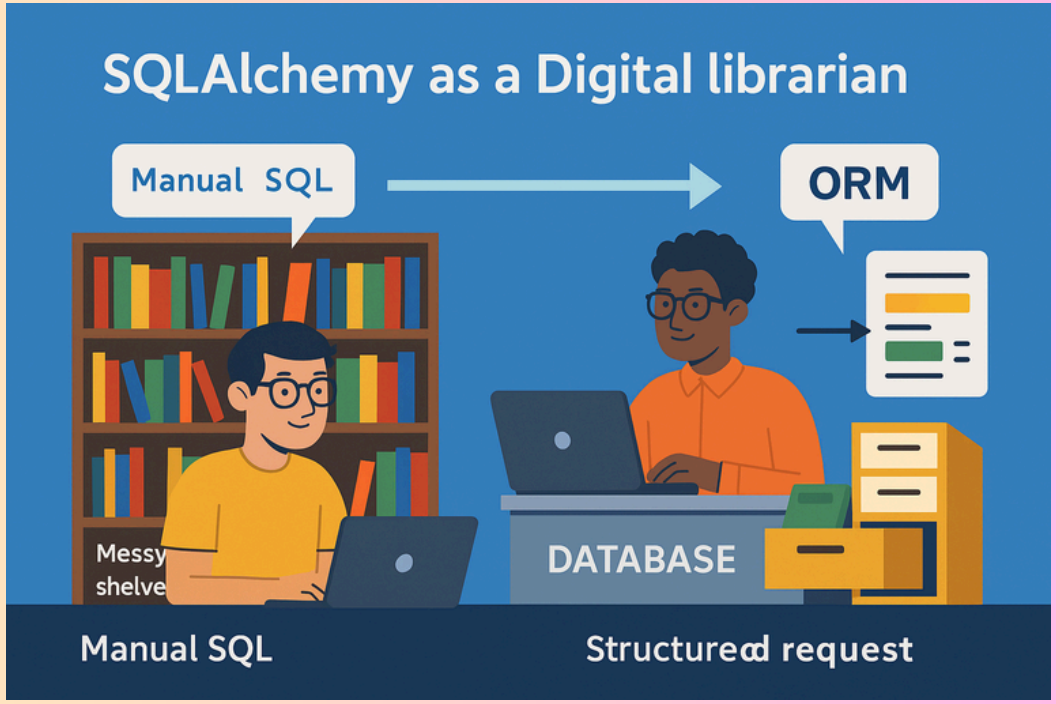
Python Challenge

DAY 27

Add database support to yesterday's FastAPI Book Management App using ***SQLAlchemy + SQLite***

By
G.Laxmi Prasanna

Analogy - 🔍 SQLAlchemy as a Digital Librarian



- Smart Requests – Turns Python code into accurate SQL queries.
- No Mess – Finds and updates data without manual digging.
- Fast & Safe – Avoids errors, ensures smooth data handling.

"Code Execution & Swagger UI Testing"

```python
from fastapi import FastAPI, HTTPException, Depends, Request
from fastapi.responses import JSONResponse
from pydantic import BaseModel
from sqlalchemy import Column, Integer, String, create_engine
from sqlalchemy.orm import declarative_base, sessionmaker, Session
from datetime import datetime
import logging

# -------------------- App Setup --------------------
app = FastAPI(title="📚 Book Manager API", version="1.0")

# -------------------- Logging Setup --------------------
logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

@app.middleware("http")
async def log_requests(request: Request, call_next):
    logger.info(f"{request.method} {request.url}")
    return await call_next(request)

# -------------------- Database Config --------------------
DATABASE_URL = "sqlite:///./books.db"
engine = create_engine(DATABASE_URL, connect_args={"check_same_thread":
False})
SessionLocal = sessionmaker(bind=engine)
Base = declarative_base()

# -------------------- Models --------------------
class Book(Base):
    __tablename__ = "books"
    id = Column(Integer, primary_key=True, index=True)
    title = Column(String)
    author = Column(String)
    year = Column(Integer)

Base.metadata.create_all(bind=engine)
```

```python
# -------------------- Pydantic Schemas --------------------
class BookBase(BaseModel):
    title: str
    author: str
    year: int

    class Config:
        orm_mode = True

class BookOut(BookBase):
    id: int

# -------------------- Dependency --------------------
def get_db():
    db = SessionLocal()
    try:
        yield db
    finally:
        db.close()

# -------------------- Routes --------------------

# ➕ Create a book
@app.post("/books/", response_model=BookOut, summary="Add a new book")
def create_book(book: BookBase, db: Session = Depends(get_db)):
    db_book = Book(**book.dict())
    db.add(db_book)
    db.commit()
    db.refresh(db_book)
    return db_book

# 📚 Get all books
@app.get("/books/", response_model=list[BookOut], summary="View all books")
def get_books(db: Session = Depends(get_db)):
    return db.query(Book).all()

# 🔍 Get book by ID
@app.get("/books/{book_id}", response_model=BookOut, summary="Get book by ID")
def get_book(book_id: int, db: Session = Depends(get_db)):
    book = db.query(Book).filter(Book.id == book_id).first()
    if not book:
        raise HTTPException(status_code=404, detail="Book not found")
    return book

# ❌ Delete book by ID
@app.delete("/books/{book_id}", summary="Delete book by ID")
def delete_book(book_id: int, db: Session = Depends(get_db)):
    book = db.query(Book).filter(Book.id == book_id).first()
    if not book:
        raise HTTPException(status_code=404, detail="Book not found")
    db.delete(book)
    db.commit()
    return {"message": f"Book {book_id} deleted successfully"}
```

Run the code

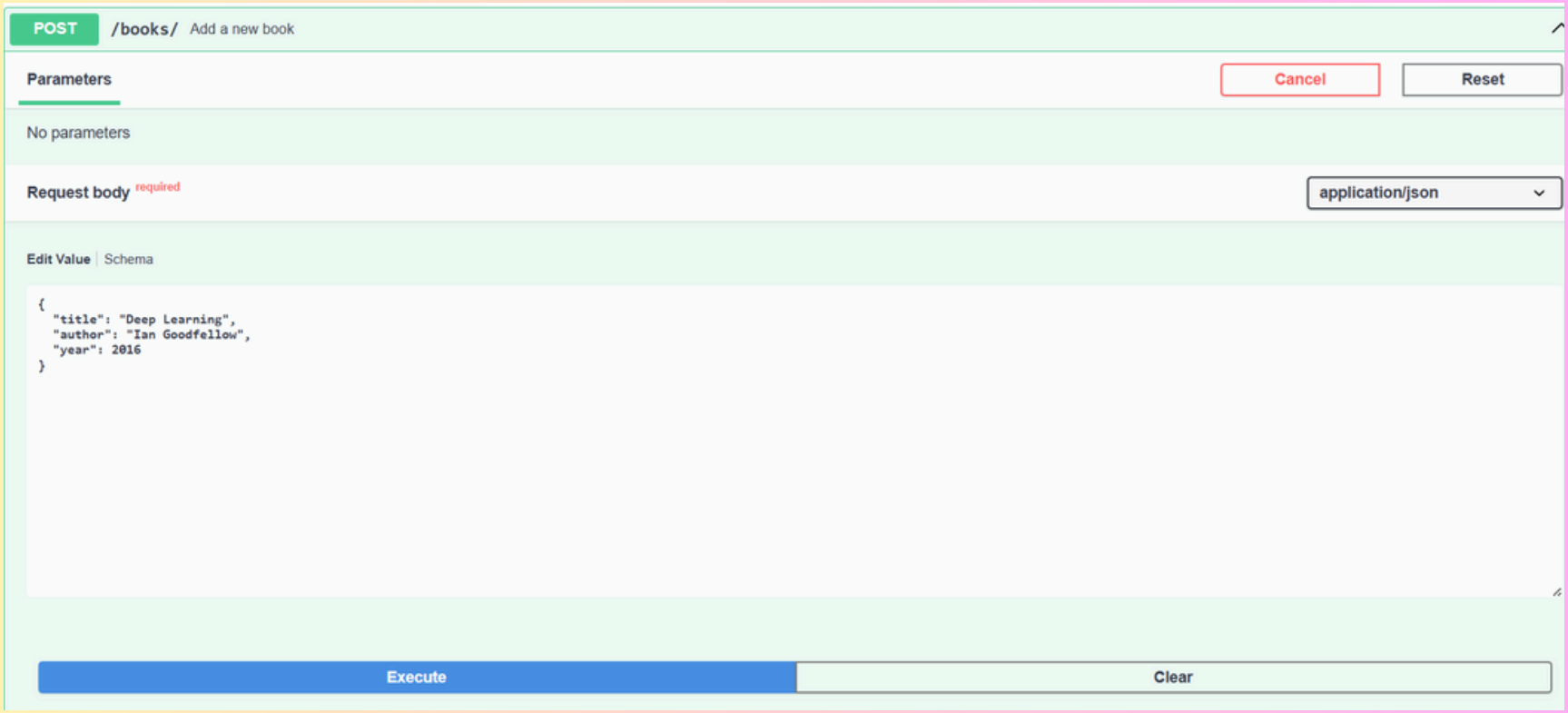
```
PS C:\Users\laxmi\OneDrive\Desktop\30> uvicorn Day27:app --reload
INFO:     Will watch for changes in these directories: ['C:\\Users\\laxmi
\\OneDrive\\Desktop\\30']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [5280] using StatReload
INFO:     Started server process [1892]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```

✅ Run the application using: uvicorn Day27:app --reload
🌐 Open **Swagger UI** at: http://127.0.0.1:8000/docs
🚀 Use the interactive API to add, view, and delete book records via **SQLAlchemy + SQLite.**

API Features Supported (via Swagger UI)

Features	Endpoints	Method
➕ Add a Book	/books/	POST
📚 View All Books	/books/	GET
🔍 View Book by ID	/books/{book_id}	GET
❌ Delete Book by ID	/books/{book_id}	DELETE

POST



✅ Expected Response:
(when ID = 1 is auto-generated)

POST RESPONSE

ID = 1 Generated

Responses

Curl

```
curl -X 'POST' \
  'http://127.0.0.1:8000/books/' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "title": "Deep Learning",
  "author": "Ian Goodfellow",
  "year": 2016
}
'
```

Request URL

```
http://127.0.0.1:8000/books/
```

Server response

Code	Details
200	Response body

```
{
  "title": "Deep Learning",
  "author": "Ian Goodfellow",
  "year": 2016,
  "id": 1
}
```

Download

Response headers

```
content-length: 70
content-type: application/json
date: Mon,23 Jun 2025 16:29:29 GMT
server: uvicorn
```

GET

📚 Get All Books Output:

GET /books/ View all books

Parameters

Cancel

No parameters

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:8000/books/' \
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/books/
```

Server response

Code Details

200

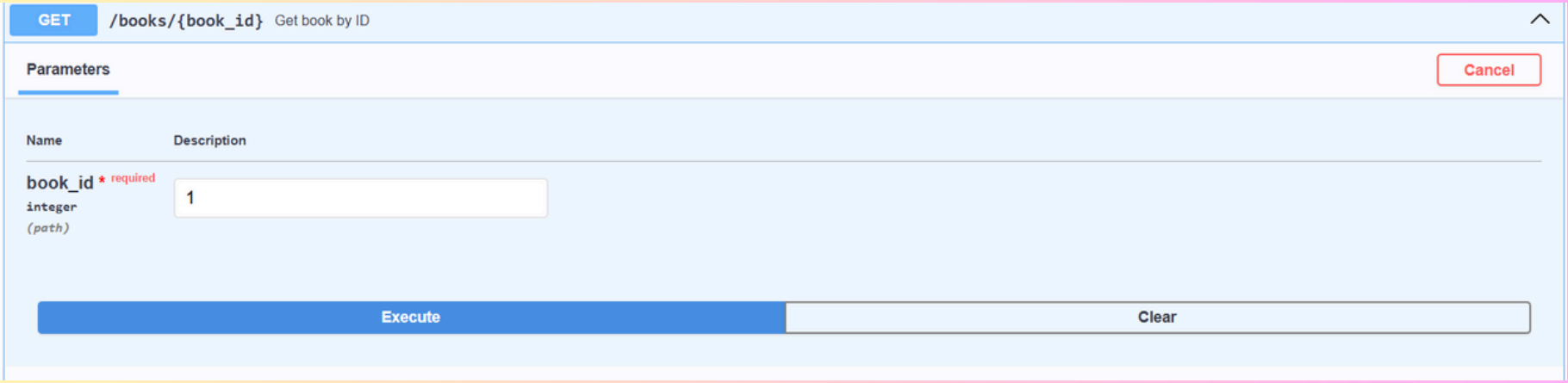
Response body

```
[
  {
    "title": "Deep Learning",
    "author": "Ian Goodfellow",
    "year": 2016,
    "id": 1
  }
]
```

Download

Returns list of all added books from the database.

GET BOOK ID



- When we input book_id = 1, the API returns the details of the corresponding book.

Response for GET /books/{book_id}

Responses

Curl

```
curl -X 'GET' \
  'http://127.0.0.1:8000/books/1' \
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/books/1
```

Server response

Code	Details
200	Response body

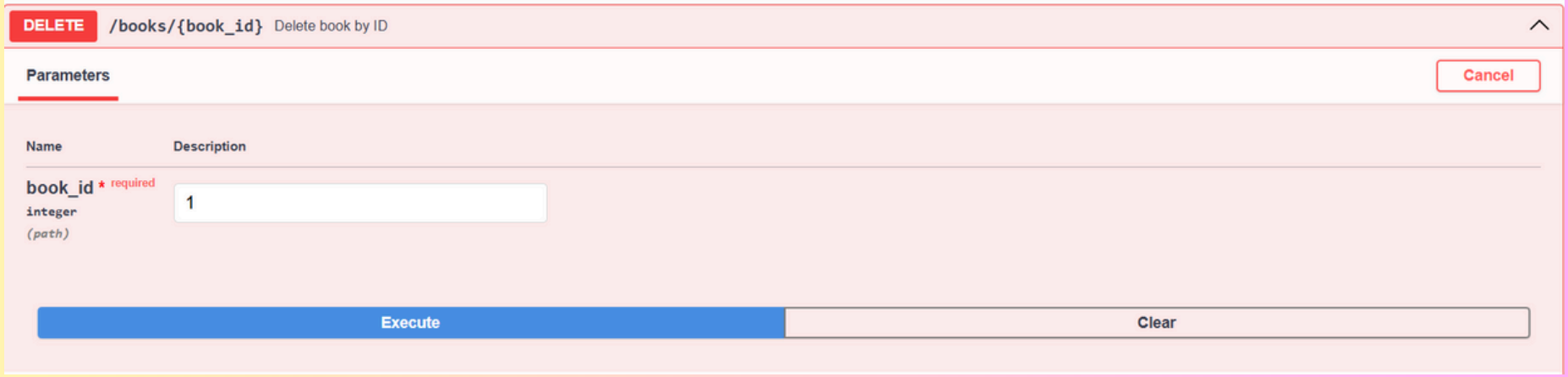
```
{
  "title": "Deep Learning",
  "author": "Ian Goodfellow",
  "year": 2016,
  "id": 1
}
```

Download

Response headers

```
content-length: 70
content-type: application/json
date: Mon,23 Jun 2025 16:31:18 GMT
server: uvicorn
```

DELETE BOOK ID



✅ Expected Response:
(when book_id = 1 is deleted)

✅ Response after deleting book with ID = 1

Confirms the **book is successfully removed from the database.**

Responses

Curl

```
curl -X 'DELETE' \
  'http://127.0.0.1:8000/books/1' \
  -H 'accept: application/json'
```

Request URL

```
http://127.0.0.1:8000/books/1
```

Server response

Code	Details
200	Response body

```
{
  "message": "Book 1 deleted successfully"
}
```

Download

Response headers

```
content-length: 41
content-type: application/json
date: Mon,23 Jun 2025 16:32:08 GMT
server: uvicorn
```

✅ Swagger UI Testing Completed Successfully
All endpoints performed as expected using SQLite + FastAP

```
PS C:\Users\laxmi\OneDrive\Desktop\30> uvicorn Day27:app --reload
INFO:     Will watch for changes in these directories: ['C:\\Users\\laxmi
\\OneDrive\\Desktop\\30']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [5280] using StatReload
INFO:     Started server process [1892]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:Day27:GET http://127.0.0.1:8000/docs
INFO:     127.0.0.1:61246 - "GET /docs HTTP/1.1" 200 OK
INFO:Day27:GET http://127.0.0.1:8000/openapi.json
INFO:     127.0.0.1:61246 - "GET /openapi.json HTTP/1.1" 200 OK
INFO:Day27:POST http://127.0.0.1:8000/books/
INFO:     127.0.0.1:61247 - "POST /books/ HTTP/1.1" 200 OK
INFO:Day27:GET http://127.0.0.1:8000/books/
INFO:     127.0.0.1:61257 - "GET /books/ HTTP/1.1" 200 OK
INFO:Day27:GET http://127.0.0.1:8000/books/1
INFO:     127.0.0.1:61259 - "GET /books/1 HTTP/1.1" 200 OK
INFO:Day27:DELETE http://127.0.0.1:8000/books/1
INFO:     127.0.0.1:61261 - "DELETE /books/1 HTTP/1.1" 200 OK
```