

## Coverage for app/routers/book\_router.py: 88%



34 statements    30 run    4 missing    0 excluded

« prev    ^ index    » next    coverage.py v7.11.3, created at 2025-11-21 23:38 -0800

```
1 from fastapi import APIRouter, HTTPException, status
2 from app.schemas.book import BookCreate, BookUpdate
3 from app.services.book_service import BookService
4
5 router = APIRouter(prefix = "/books", tags = ["Books"])
6 service = BookService()
7
8 @router.post("/")
9 def create_book(book: BookCreate):
10     """Create a new book entry in the system."""
11     try:
12         return service.create_book(book)
13     except ValueError as e:
14         raise HTTPException(status_code = 400, detail = str(e))
15
16 @router.get("/")
17 def get_all_books():
18     """Retrieves a list of books."""
19     return service.get_all_books()
20
21 @router.get("/{isbn}")
22 def get_book(isbn: str):
23     """Retrieves a specific book by ISBN."""
24     book = service.get_book(isbn)
25     if not book:
26         raise HTTPException(status_code = status.HTTP_404_NOT_FOUND, detail = "Book not found")
27     return book
28
29 @router.put("/{isbn}")
30 def update_book(isbn: str, book: BookUpdate):
31     """Updates an existing book's details."""
```

```
32     try:
33         updated_book = service.update_book(isbn, book)
34         if not updated_book:
35             raise HTTPException(status_code = status.HTTP_404_NOT_FOUND, detail = "Book not found")
36         return updated_book
37     except ValueError as e:
38         raise HTTPException(status_code = 400, detail = str(e))
39
40 @router.delete("/{isbn}")
41 def delete_book(isbn: str):
42     """Delete a book from the system by its ISBN."""
43     if not service.delete_book(isbn):
44         raise HTTPException(status_code = status.HTTP_404_NOT_FOUND, detail = "Book not found")
45     return {"message": "Book deleted successfully"}
```

« prev ^ index » next coverage.py v7.11.3, created at 2025-11-21 23:38 -0800