

📄 server.js (Mongoose CRUD for Books)

```
require('dotenv').config();
```

```
const express = require('express');
```

```
const mongoose = require('mongoose');
```

```
const app = express();
```

```
app.use(express.json()); // Middleware to parse JSON
```

```
// 🔗 Connect to MongoDB (Make sure MongoDB is running locally)
```

```
mongoose.connect('mongodb://127.0.0.1:27017/libraryDB', {
```

```
  useNewUrlParser: true,
```

```
  useUnifiedTopology: true
```

```
}).then(() => console.log('✅ MongoDB Connected'))
```

```
.catch(err => console.error('❌ MongoDB Connection Error:', err));
```

```
// 📄 Define a Mongoose Schema & Model for Books
```

```
const bookSchema = new mongoose.Schema({
```

```
  title: String,
```

```
  author: String,
```

```
  yearPublished: Number,
```

```
  genre: String
```

```
});
```

```
const Book = mongoose.model('Book', bookSchema);
```


```
// 📄 CRUD Routes for Books
```

```
// 1️⃣ CREATE a new book
```

```
app.post('/books', async (req, res) => {  
  try {  
    const book = await Book.create(req.body);  
    res.status(201).json(book);  
  } catch (error) {  
    res.status(400).json({ error: error.message });  
  }  
});
```

// 2  READ all books

```
app.get('/books', async (req, res) => {  
  const books = await Book.find();  
  res.json(books);  
});
```


// 3  READ one book by title

```
app.get('/books/:title', async (req, res) => {  
  const book = await Book.findOne({ title: req.params.title });  
  if (!book) return res.status(404).json({ message: 'Book not found' });  
  res.json(book);  
});
```

// 4  UPDATE a book by ID

```
app.put('/books/:id', async (req, res) => {  
  try {  
    const book = await Book.findByIdAndUpdate(req.params.id, req.body, { new: true });  
    if (!book) return res.status(404).json({ message: 'Book not found' });  
    res.json(book);  
  } catch (error) {
```

```
    res.status(400).json({ error: error.message });  
  }  
});
```

```
// 5  DELETE a book by ID  
app.delete('/books/:id', async (req, res) => {  
  const book = await Book.findByIdAndDelete(req.params.id);  
  if (!book) return res.status(404).json({ message: 'Book not found' });  
  res.json({ message: 'Book deleted' });  
});
```

```
//  Start the server  
const PORT = process.env.PORT || 3000;  
app.listen(PORT, () => console.log(`✔ Server running on port ${PORT}`));
```