Perfect — here's a **comprehensive and layered API design** to challenge and grow your Spring Boot skills.

---

## 📦 Controller Hierarchy Overview

```scss
com.library.management.controller
├── CustomerController
├── BookController
├── IssueController
├── ReturnController
├── ReviewController
├── PaymentController
├── AdminController
├── ReportController
├── SearchController
├── SystemController (Actuator, info, versioning)
```

---

## 📌 REST Endpoint List (Grouped by Controller)

### 👤 CustomerController

| Type | Endpoint | Description |
|---|---|---|
| `POST` | `/api/customers` | Register a new customer |
| `GET` | `/api/customers/{id}` | Fetch customer details |
| `PUT` | `/api/customers/{id}` | Update customer info |
| `GET` | `/api/customers/{id}/books` | List all currently issued books |
| `GET` | `/api/customers/{id}/history` | Complete borrowing history |
| `DELETE` | `/api/customers/{id}` | Delete a customer (if no active books) |
| `GET` | `/api/customers/active` | All customers with active issued books |
| `GET` | `/api/customers/defaulters` | Customers with pending fines or unpaid fees |
| `POST` | `/api/customers/{id}/deactivate` | Temporarily deactivate a customer |

---

### 📚 BookController

| Type | Endpoint | Description |
|---|---|---|
| `GET` | `/api/books` | Get all books with availability |
| `GET` | `/api/books/{id}` | Get book details by ID |
| `POST` | `/api/books` | Add a new book |
| `PUT` | `/api/books/{id}` | Update book details |
| `DELETE` | `/api/books/{id}` | Delete a book (if not issued) |
| `GET` | `/api/books/out-of-stock` | List books with 0 available copies |
| `GET` | `/api/books/recommended` | Recommend books based on reviews |
| `GET` | `/api/books/popular` | Books with highest borrow count |

| Type | Endpoint | Description |
|---|---|---|
| `PATCH` | `/api/books/{id}/adjust-stock` | Increase/decrease stock manually |

---

## 🔄 IssueController

| Type | Endpoint | Description |
|---|---|---|
| `POST` | `/api/issues` | Issue a book to a customer |
| `GET` | `/api/issues/{id}` | Get issue details |
| `GET` | `/api/issues/active` | Get all active issued books |
| `GET` | `/api/issues/overdue` | Books not returned by due date |
| `GET` | `/api/issues/customer/{customerId}` | All issues by a customer |
| `POST` | `/api/issues/batch` | Issue multiple books in one API |
| `POST` | `/api/issues/{id}/extend` | Extend due date of an issued book |
| `GET` | `/api/issues/fines/today` | List today's newly calculated fines |

---

## 🔁 ReturnController

| Type | Endpoint | Description |
|---|---|---|
| `PUT` | `/api/returns/{issueId}` | Return a book and calculate fine |
| `POST` | `/api/returns/bulk` | Bulk return by customer or list of issue IDs |
| `GET` | `/api/returns/late` | Returned books that incurred fine |
| `GET` | `/api/returns/today` | Books returned today |
| `GET` | `/api/returns/history/{customerId}` | Return history for a customer |

---

## ⭐ ReviewController

| Type | Endpoint | Description |
|---|---|---|
| `POST` | `/api/books/{bookId}/review` | Submit review for a book |
| `GET` | `/api/books/{bookId}/reviews` | Get all reviews for a book |
| `PUT` | `/api/reviews/{id}` | Update a review |
| `DELETE` | `/api/reviews/{id}` | Delete a review |
| `GET` | `/api/reviews/customer/{customerId}` | Get all reviews by a customer |
| `GET` | `/api/reviews/top` | Top-rated books by average rating |

---

## 💳 PaymentController

| Type | Endpoint | Description |
|---|---|---|
| `POST` | `/api/payments` | Record a fee payment |
| `GET` | `/api/payments/customer/{id}` | All payments made by a customer |
| `GET` | `/api/payments/pending` | Customers with no payment this month |

| Type | Endpoint | Description |
|---|---|---|
| `GET` | `/api/payments/monthly?month=07&year=2025` | Payments for a specific month |
| `GET` | `/api/payments/stats` | Monthly income, overdue, and totals |

---

### 📊 ReportController

| Type | Endpoint | Description |
|---|---|---|
| `GET` | `/api/reports/daily-summary` | Daily summary: issues, returns, fines |
| `GET` | `/api/reports/monthly-activity` | Monthly issue/return/payment stats |
| `GET` | `/api/reports/book-usage/{bookId}` | Book usage stats: who issued when |
| `GET` | `/api/reports/defaulter-list` | List of customers with unpaid fines or returns |
| `GET` | `/api/reports/system-usage` | Total books, users, active issues etc. |

---

### 🔍 SearchController

| Type | Endpoint | Description |
|---|---|---|
| `GET` | `/api/search/books?query=...` | Search books by title, author, ISBN |
| `GET` | `/api/search/customers?query=...` | Search customers by name/email |
| `GET` | `/api/search/issues?bookId=...&customerId=...` | Filter issues |
| `GET` | `/api/search/reviews?rating=5` | Filter reviews |

---

### 🛠️ SystemController (Optional Utility APIs)

| Type | Endpoint | Description |
|---|---|---|
| `GET` | `/api/system/status` | Custom app status |
| `GET` | `/api/system/version` | App version and build time |
| `GET` | `/api/system/health` | Custom health metrics |
| `POST` | `/api/system/seed` | Seed test data |
| `GET` | `/actuator/*` | All Spring Boot actuator endpoints |

---

## 🧪 API Complexity Levels

| Level | Examples |
|---|---|
| 🟢 Basic CRUD | `POST /customers`, `GET /books` |
| 🟡 Moderate Logic | `POST /issues` (check fee, count), `PUT /returns/{id}` (fine calc) |
| 🔴 Advanced/Complex | `GET /reports/monthly-activity`, `POST /issues/batch`, `GET /defaulters` |

---

## 🎯 Suggestions for You

- Start with simple CRUD (customers, books)

- Then move to business rules (issue, return, fee check)
- Later build advanced stats (reports, overdue, monthly insights)
- Keep controllers clean and use DTOs properly

---

Let me know once you start implementing — I can help you review structure, optimize service logic, or create challenges (like partial returns, scheduled fine cleanup, etc).

hii

Hey! 👋 What can I help you with today?