Book sell management - TÀI LIỆU REST API

**Công nghệ sử dụng:**

Spring Boot 3

Spring Security + JWT

JPA / Hibernate

MySQL

Lombok

Email sender

**DanhGiaAPI - Quản lý đánh giá sách**

Base URL: /api/danhgia

**1. Thêm đánh giá mới cho sách**

URL: POST /api/danhgia/sach/{maSach}

Quyền: Người dùng đã đăng nhập

Mô tả: Thêm một đánh giá mới cho sách bởi người dùng hiện tại (lấy từ token).

**Headers:**

Authorization: Bearer <JWT Token>

**Path Variable:**

maSach (Integer): Mã sách cần đánh giá

Request Body (JSON):

json

{

"soSao": 5,

"nhanXet": "Nội dung rất hấp dẫn!"

}

**Response: 200 OK**

json

{

"id": 1,

"soSao": 5,

"nhanXet": "Nội dung rất hấp dẫn!",

"sach": { ... },

"nguoiDung": { ... }

}

**2. Cập nhật đánh giá**

**URL:** PUT /api/danhgia/{maDanhGia}

Quyền: Người dùng đã đăng nhập

Mô tả: Cập nhật lại nội dung và số sao của đánh giá do chính người dùng đã tạo.

**Headers:**

Authorization: Bearer <JWT Token>

**Path Variable:**

maDanhGia (Integer): Mã của đánh giá cần cập nhật

**Request Body (JSON):**

json

{

"soSao": 4,

"nhanXet": "Cũng ổn, nhưng phần kết hơi thiếu"

}

**Response:** 200 OK

json

{

"id": 1,

"soSao": 4,

"nhanXet": "Cũng ổn, nhưng phần kết hơi thiếu",

"sach": { ... },

"nguoiDung": { ... }

}

**3. Lấy danh sách đánh giá theo mã sách**

**URL:** GET /api/danhgia/sach/{maSach}

**Quyền:** Người dùng đã đăng nhập

**Mô tả:** Trả về tất cả các đánh giá của một sách cụ thể.

**Path Variable:**

maSach (Integer): Mã sách

**Response:** 200 OK

json

[

{

"id": 1,

"soSao": 5,

"nhanXet": "Rất hay!",

"sach": { ... },

"nguoiDung": { ... }

},

...

]

**4. Lấy danh sách đánh giá theo mã người dùng**

**URL:** GET /api/danhgia/user/{userId}

**Quyền:** Nhân viên (STAFF)

**Mô tả:** Trả về tất cả các đánh giá mà người dùng đã thực hiện.

**Path Variable:**

userId (Integer): Mã người dùng

**Response:** 200 OK

json

[

{

"id": 2,

"soSao": 4,

"nhanXet": "Khá tốt",

"sach": { ... },

"nguoiDung": { ... }

},

...

]

**5. Lấy điểm đánh giá trung bình theo sách**

**URL:** GET /api/danhgia/trungBinhDanhGia/sach/{sachId}

**Quyền:** Người dùng đã đăng nhập

**Mô tả:** Trả về điểm trung bình (float) của tất cả đánh giá dành cho sách cụ thể.

**Path Variable:**

sachId (Integer): Mã sách

**Response:** 200 OK

4.25

DonHangAPI – Quản lý đơn hàng

**Base URL**: /api/donHang

1. Tạo đơn hàng mới

**URL:** POST /api/donHang/hinhThucThanhToan/sach

**Quyền:** Người dùng đã đăng nhập

**Mô tả:** Tạo đơn hàng với hình thức thanh toán và giao hàng cụ thể.

**Headers:**

Authorization: Bearer <JWT Token>

**Request Parameters:**

tenHinhThucThanhToan (String)

tenHinhThucGiaoHang (String)

**Request Body (JSON):**

json

{

"chiTietDonHangs": [

{

"maSach": 1,

"soLuong": 2

}

]

}

**Response:**

201 Created: Đơn hàng tạo thành công

404 Not Found: Người dùng không tồn tại

500 Internal Server Error: Lỗi server khi tạo đơn hàng

**2. Xác nhận đơn hàng (STAFF)**

**URL:** PUT /api/donHang/xacnhan/{maDonHang}

**Quyền:** Nhân viên

**Mô tả:** Xác nhận đơn hàng đang chờ xử lý.

**Path Variable:**

maDonHang (Integer)

**Response:**

json

{

"noiDung": "da xac nhan don hang thanh cong"

}

404, 500 nếu lỗi

**3. Xử lý giao hàng qua MoMo (STAFF)**

**URL**: PUT /api/donHang/xulygiaohang-momo/{maDonHang}

**Quyền:** Nhân viên

**Mô tả:** Xác nhận thanh toán MoMo và cập nhật trạng thái giao hàng.

**Request Parameter:**

maThanhToan (String)

**Response:**

json

"successed"

404, 500 nếu lỗi

**4. Xử lý giao hàng COD (STAFF)**

**URL:** PUT /api/donHang/xulygiaohang-cod/{maDonHang}

**Quyền:** Nhân viên

**Mô tả:** Xác nhận và chuyển trạng thái đơn hàng COD sang "ĐANGGIAO".

**Response:**

json

"xu ly don hang thanh cong cap nhat trang thai DANGGIAO"

400 nếu điều kiện không hợp lệ

404 nếu đơn hàng không tồn tại

**5. Hoàn tất đơn hàng (STAFF)**

**URL:** PUT /api/donHang/hoantatdonhang/{maDonHang}

**Quyền:** Nhân viên

**Mô tả:** Đánh dấu đơn hàng đã giao thành công.

**Response:** 200 OK, 404 Not Found nếu đơn hàng không tồn tại

**6. Hủy đơn hàng (Người dùng)**

**URL:** PUT /api/donHang/huydonhang/{maDonHang}

**Quyền:** Người dùng đã đăng nhập

**Mô tả:** Hủy đơn hàng đang chờ xác nhận.

**Response:**

200 OK: Hủy thành công

404 Not Found: Đơn hàng không tồn tại hoặc không thể hủy

**7. Xóa đơn hàng (Admin)**

**URL:** DELETE /api/donHang/admin/{id}

**Quyền:** Admin

**Mô tả:** Xóa đơn hàng theo mã.

**Response:** 200 OK

**8. Thống kê doanh thu theo ngày (Admin)**

**URL:** GET /api/donHang/admin/doanhthu

**Quyền:** Admin

**Mô tả:** Tính tổng doanh thu theo ngày cụ thể.

**Request Param:**

ngay (yyyy-MM-dd)

**Response:**

json

3500000.0

**9. Lấy toàn bộ đơn hàng (Admin)**

**URL:** GET /api/donHang/admin

**Quyền:** Admin

**Mô tả:** Trả về danh sách tất cả đơn hàng trong hệ thống.

**Response:**

json

[

{

"id": 1,

"trangThai": "DANGGIAO",

"ngayDatHang": "2025-08-03",

...

}

]

HinhAnhAPI – Quản lý hình ảnh

**Base URL:** /api/hinhanh

1. Tải lên 1 hình ảnh

**URL:** POST /upload

**Quyền:** Người dùng đã đăng nhập

**Mô tả:** Tải một hình ảnh cho sách.

**Request Parameters:**

sachId (Integer)

file (MultipartFile)

**Response:**

json

{

"id": 1,

"duLieuAnh": "/uploads/abc.jpg",

"isDaiDien": false,

"sach": { "maSach": 1, "tenSach": "..." }

}

404 Not Found: Sách không tồn tại

**2. Tải lên nhiều hình ảnh**

**URL:** POST /upload-multiple

**Quyền:** Người dùng đã đăng nhập

**Mô tả:** Tải nhiều ảnh cùng lúc cho một cuốn sách.

**Request Parameters:**

sachId (Integer)

file[] (MultipartFile array)

**Response:** 200 OK

404 Not Found: Sách không tồn tại

**3.Cập nhật ảnh đại diện cho sách**

**URL:** PUT /sach/{sachId}/hinhanh/{maHinhAnh}

**Quyền:** Người dùng đã đăng nhập

**Mô tả:** Cập nhật ảnh đại diện cho một sách.

**Path Variables:**

sachId (Integer)

maHinhAnh (Integer)

**Response:** 200 OK

**4. Lấy danh sách ảnh theo sách**

**URL:** GET /sach/{sachId}

**Quyền:** Public

**Mô tả:** Trả về toàn bộ ảnh của một sách.

**Response:**

json

[

{

"id": 1,

"duLieuAnh": "/uploads/abc.jpg",

"isDaiDien": false,

"sach": {...}

}

]

**5. Lấy chi tiết ảnh theo ID**

**URL:** GET /{id}

**Quyền:** Public

**Mô tả:** Lấy thông tin một ảnh theo ID.

**Response:** 200 OK với object HinhAnh

**6. Xóa ảnh theo ID**

**URL:** DELETE /{id}

**Quyền:** Người dùng đã đăng nhập

**Mô tả:** Xóa một ảnh cụ thể.

**Response:** 200 OK

7. Lấy toàn bộ ảnh (Admin)

**URL:** GET /admin/get

**Quyền:** Admin

**Mô tả:** Lấy danh sách tất cả ảnh trong hệ thống.

**Response:** Danh sách ảnh

**8. Xóa ảnh theo sách (Admin)**

**URL:** DELETE /admin/sach/{sachId}

**Quyền:** Admin

**Mô tả:** Xóa toàn bộ ảnh của một sách.

**Response:** 200 OK

**9. Lấy ảnh đại diện của sách**

**URL:** GET /anhDaiDien

**Quyền:** Người dùng đã đăng nhập

**Request Parameter:**

sachId (Integer)

**Mô tả:** Trả về danh sách ảnh được đánh dấu là isDaiDien = true.

**Response:** Danh sách HinhAnh

**SachAPI – Quản lý sách**

**Base URL: /api/sach**

**1. Tìm sách theo ID**

**URL:** GET /{id}

**Quyền:** Public

**Mô tả:** Lấy thông tin chi tiết một cuốn sách.

**Path Variable:**

id (Integer) – mã sách

**Response:** Optional<Sach>

**2. Lấy tất cả sách**

**URL:** GET /

**Quyền:** Public

**Mô tả:** Trả về danh sách tất cả sách trong hệ thống.

**Response:** List<Sach>

**3. Cập nhật sách**

**URL:** PUT /capnhat/{id}

**Quyền**: Nhân viên (staff)

**Mô tả:** Cập nhật thông tin một sách.

**Path Variable:**

id (Integer)

**Body:** UpdateSachDto

Response: Sach cập nhật thành công

**4. Thêm sách mới**

**URL:** POST /add

**Quyền:** Nhân viên (staff)

**Mô tả:** Thêm sách mới vào hệ thống.

**Body:** AddSachDto

**Response:** Sach mới được thêm

**5. Xóa sách theo ID**

**URL:** DELETE /delete/{id}

**Quyền:** Admin

**Mô tả:** Xóa một sách khỏi hệ thống.

**Path Variable:**

id (Integer)

**Response:** 200 OK

**6. Tìm sách theo từ khóa tên**

**URL:** GET /conains

**Quyền**: Public

**Query Param:** keyword (String)

**Mô tả:** Tìm tất cả sách có chứa từ khóa trong tên.

**Response:** List<Sach>

**7. Tìm sách theo thể loại**

**URL:** GET /theloai

**Quyền:** Public

**Query Param:** theLoai (String)

**Mô tả:** Trả về danh sách sách thuộc thể loại nhất định.

**Response:** List<Sach>

**8. Tìm sách theo tác giả**

**URL:** GET /tacgia

**Quyền:** Public

**Query Param:** tacGia (String)

**Mô tả:** Trả về danh sách sách của tác giả.

**Response:** List<Sach>

**9. Tìm sách theo khoảng giá**

**URL:** GET /{min}/{max}

**Quyền:** Public

**Path Variable:**

min (double)

max (double)

**Mô tả:** Lọc sách có giá nằm trong khoảng.

**Response:** List<Sach>  
**SachYeuThichAPI – Quản lý danh sách yêu thích**

**Base URL: /api/sachYeuThich**

**1. Thêm sách vào danh sách yêu thích**

**URL:** POST /sach/{sachId}

**Quyền:** Người dùng đăng nhập (loginuser)

**Headers:**

Authorization: Token JWT

**Path Variable:**

sachId (Integer): Mã sách cần thêm

**Mô tả:** Thêm một cuốn sách vào danh sách yêu thích của người dùng.

Response: 201 Created

**2. Xóa sách khỏi danh sách yêu thích**

**URL:** DELETE /{sachId}

**Quyền:** Người dùng đăng nhập (loginuser)

**Headers:**

Authorization: Token JWT

**Path Variable:**

sachId (Integer): Mã sách cần xóa

**Mô tả:** Xóa sách ra khỏi danh sách yêu thích của người dùng hiện tại.

**Response:** 200 OK

**TaiKhoanAPI – Xác thực người dùng**

**Base URL: /api/auth**

**Mô tả:** Quản lý đăng ký, đăng nhập, kích hoạt tài khoản, quên mật khẩu, đổi mật khẩu

**1. Đăng ký tài khoản**

**URL:** POST /dang-ky

**Request Body:** DangKyRequest

**Mô tả:** Người dùng gửi thông tin đăng ký để tạo tài khoản mới.

**Response:** 200 OK nếu thành công, 400 Bad Request nếu dữ liệu không hợp lệ.

**2. Kích hoạt tài khoản**

**URL:** POST /kich-hoat

**Request Body:** KichHoatRequest

**Mô tả:** Nhập mã OTP để kích hoạt tài khoản sau khi đăng ký.

**Response:** 200 OK nếu thành công, 400 Bad Request nếu mã sai.

**3. Đăng nhập**

**URL:** POST /dang-nhap

**Request Body**: LoginRequest

**Mô tả:** Đăng nhập bằng tên đăng nhập và mật khẩu, trả về JWT token nếu hợp lệ.

**Response:** 200 OK (token), 400 Bad Request nếu sai thông tin.

**4. Đổi mật khẩu**

**URL:** PUT /reset-password

**Request Body:** ResetPasswordRequest

**Mô tả:** Đổi mật khẩu khi đã đăng nhập hoặc có mã xác nhận.

**Response:** 200 OK nếu thành công, 400 Bad Request nếu không hợp lệ.

**5. Gửi mã đổi mật khẩu**

**URL:** POST /get-maResetPassWord

**Request Body:** KichHoatRequest

**Mô tả:** Gửi mã OTP về email để xác nhận trước khi đặt lại mật khẩu.

**Response:** 200 OK nếu gửi thành công, 404 Not Found nếu email không tồn tại.

**6. Quên mật khẩu**

**URL:** PUT /forgetPassword

**Request Body:** ForgetPasswordRequest

**Mô tả:** Dùng mã OTP + email để thiết lập lại mật khẩu.

**Response:**

200 OK nếu đặt lại thành công

400 Bad Request nếu OTP sai

404 Not Found nếu email không đúng

**TheLoaiAPI – Quản lý thể loại sách**

**Base URL: /api/theloai**

**Mô tả:** Quản lý thông tin các thể loại sách và truy xuất sách theo thể loại.

**1. Lấy thông tin thể loại theo ID**

**URL:** GET /{id}

**Quyền truy cập:** Công khai

**Mô tả:** Lấy thông tin chi tiết của một thể loại cụ thể.

Response: 200 OK với TheLoai

**2. Lấy danh sách tất cả thể loại**

**URL:** GET /

**Quyền truy cập:** Công khai

**Mô tả:** Lấy toàn bộ danh sách thể loại sách có trong hệ thống.

**Response:** 200 OK với List<TheLoai>

**3. Tạo mới thể loại**

**URL:** POST /add

**Quyền truy cập:** Admin

**Request Body:** TheLoai

**Mô tả:** Thêm một thể loại sách mới vào hệ thống.

**Response:** 200 OK với TheLoai đã tạo

**4. Cập nhật thể loại**

**URL:** PUT /update/{id}

**Quyền truy cập:** Admin

**Request Body**: TheLoai

**Mô tả**: Cập nhật thông tin thể loại theo ID.

**Response:** 200 OK với thể loại đã cập nhật, hoặc lỗi nếu không tồn tại.

**5 Xóa thể loại theo ID**

**URL:** DELETE /delete/{id}

**Quyền truy cập:** Admin

**Mô tả:** Xóa thể loại sách theo ID nếu không còn liên kết với sách.

**Response:** 200 OK nếu xóa thành công.

**6. Lấy danh sách sách thuộc thể loạ**i

**URL:** GET /sach/{id}

**Quyền truy cập:** Công khai

**Mô tả:** Lấy danh sách các sách có trong một thể loại cụ thể.

**Response:** 200 OK với List<Sach>

**UserAPI – Quản lý người dùng**

**Base URL: /api/user**

**Mô tả:** Quản lý thông tin người dùng, phân quyền, chỉnh sửa và xác minh người dùng.

**1. Xóa người dùng theo ID**

**URL:** DELETE /{id}

**Quyền truy cập:** Admin

**Mô tả:** Xóa người dùng khỏi hệ thống dựa vào ID.

**Response:** 200 OK nếu xóa thành công

**2. Thêm người dùng (qua Admin)**

**URL:** POST /

**Quyền truy cập**: Admin

**Request Body:** NguoiDung

**Mô tả:** Admin thêm người dùng mới vào hệ thống.

**Response:**

200 OK + Thông báo "đã thêm thành công người dùng"

500 INTERNAL\_SERVER\_ERROR nếu lỗi khi thêm

**3. Lấy danh sách tất cả người dùng**

**URL:** GET /

**Quyền truy cập:** Admin

**Mô tả:** Trả về danh sách toàn bộ người dùng hiện có trong hệ thống.

**Response:**

200 OK + List<NguoiDung>

500 INTERNAL\_SERVER\_ERROR nếu danh sách trống

4**. Cập nhật thông tin người dùng**

**URL:** PUT /{id}

**Quyền truy cập:** Admin hoặc user (tùy bạn xử lý trong service)

**Request Body:** UpdateUser

**Mô tả:** Cập nhật thông tin người dùng theo ID.

**Response:**

200 OK + Optional<NguoiDung> nếu cập nhật thành công

404 NOT\_FOUND nếu không tồn tại ID

**5. Lấy thông tin người dùng từ JWT token**

**URL:** GET /gettokenuser

**Header:** Authorization: Bearer <token>

**Quyền truy cập:** Người dùng đã đăng nhập

**Mô tả:** Lấy thông tin người dùng từ token đã đăng nhập.

**Response:**

200 OK + NguoiDungDTO nếu thành công

404 NOT\_FOUND nếu không tìm thấy user

500 INTERNAL\_SERVER\_ERROR nếu lỗi server