**【一】接口总览表（API 清单）**

**1. 用户认证模块**

| **接口路径** | **方法** | **请求参数** | **响应结构** | **错误码** |
| --- | --- | --- | --- | --- |
| /api/auth/login | POST | username(字符串,必填,用户名)<br>password(字符串,必填,密码) | code(整数,状态码)<br>data(对象,包含token,user,role)<br>message(字符串,提示信息) | 401: 用户名或密码错误<br>429: 登录失败次数过多,请稍后再试 |
| /api/auth/register | POST | username(字符串,必填,用户名)<br>password(字符串,必填,密码)<br>email(字符串,必填,邮箱)<br>name(字符串,选填,姓名) | code(整数,状态码)<br>data(对象,注册结果)<br>message(字符串,提示信息) | 400: 请求参数错误<br>409: 用户名已存在 |
| /api/auth/logout | POST | 无 | code(整数,状态码)<br>message(字符串,提示信息) | 401: 未授权 |
| /api/auth/user | GET | 无 | code(整数,状态码)<br>data(对象,用户信息)<br>message(字符串,提示信息) | 401: 未授权 |
| /api/auth/profile | PUT | name(字符串,选填,姓名)<br>email(字符串,选填,邮箱)<br>phone(字符串,选填,电话)<br>address(字符串,选填,地址) | code(整数,状态码)<br>data(对象,更新后的用户信息)<br>message(字符串,提示信息) | 400: 请求参数错误<br>401: 未授权 |
| /api/auth/password | PUT | currentPassword(字符串,必填,当前密码)<br>newPassword(字符串,必填,新密码) | code(整数,状态码)<br>message(字符串,提示信息) | 400: 当前密码错误<br>401: 未授权 |

**2. 图书管理模块**

| **接口路径** | **方法** | **请求参数** | **响应结构** | **错误码** |
| --- | --- | --- | --- | --- |
| /api/books | GET | page(整数,选填,页码,默认1)<br>limit(整数,选填,每页条数,默认10)<br>search(字符串,选填,搜索关键词)<br>category(字符串,选填,分类)<br>sortBy(字符串,选填,排序字段)<br>inStock(布尔值,选填,是否有库存) | code(整数,状态码)<br>data(对象,包含books数组和total总数)<br>message(字符串,提示信息) | 无特殊错误码 |
| /api/books/{id} | GET | id(整数,必填,图书ID,路径参数) | code(整数,状态码)<br>data(对象,图书详情)<br>message(字符串,提示信息) | 404: 图书不存在 |
| /api/books/search | GET | query(字符串,必填,搜索关键词) | code(整数,状态码)<br>data(对象,包含books数组)<br>message(字符串,提示信息) | 无特殊错误码 |
| /api/books/categories | GET | 无 | code(整数,状态码)<br>data(数组,分类列表)<br>message(字符串,提示信息) | 无特殊错误码 |
| /api/admin/books | POST | title(字符串,必填,书名)<br>author(字符串,必填,作者)<br>category(字符串,必填,分类)<br>description(字符串,必填,简介)<br>stock(整数,必填,库存)<br>isbn(字符串,选填,ISBN号)<br>publisher(字符串,选填,出版社)<br>publishDate(日期,选填,出版日期)<br>pages(整数,选填,页数)<br>cover(字符串,选填,封面URL) | code(整数,状态码)<br>data(对象,创建的图书)<br>message(字符串,提示信息) | 400: 请求参数错误<br>401: 未授权<br>403: 权限不足 |
| /api/admin/books/{id} | PUT | id(整数,必填,图书ID,路径参数)<br>title(字符串,选填,书名)<br>author(字符串,选填,作者)<br>category(字符串,选填,分类)<br>description(字符串,选填,简介)<br>stock(整数,选填,库存)<br>isbn(字符串,选填,ISBN号)<br>publisher(字符串,选填,出版社)<br>publishDate(日期,选填,出版日期)<br>pages(整数,选填,页数)<br>cover(字符串,选填,封面URL) | code(整数,状态码)<br>data(对象,更新后的图书)<br>message(字符串,提示信息) | 400: 请求参数错误<br>401: 未授权<br>403: 权限不足<br>404: 图书不存在 |
| /api/admin/books/{id} | DELETE | id(整数,必填,图书ID,路径参数) | code(整数,状态码)<br>message(字符串,提示信息) | 401: 未授权<br>403: 权限不足<br>404: 图书不存在 |

**3. 借阅管理模块**

| **接口路径** | **方法** | **请求参数** | **响应结构** | **错误码** |
| --- | --- | --- | --- | --- |
| /api/borrowing/borrow | POST | bookId(整数,必填,图书ID) | code(整数,状态码)<br>data(对象,借阅记录)<br>message(字符串,提示信息) | 400: 图书库存不足<br>401: 未授权<br>404: 图书不存在 |
| /api/borrowing/return/{recordId} | POST | recordId(整数,必填,借阅记录ID,路径参数) | code(整数,状态码)<br>data(对象,更新后的借阅记录)<br>message(字符串,提示信息) | 401: 未授权<br>403: 无权操作此借阅记录<br>404: 借阅记录不存在<br>409: 此书已归还 |
| /api/borrowing/user | GET | page(整数,选填,页码,默认1)<br>limit(整数,选填,每页条数,默认10) | code(整数,状态码)<br>data(对象,包含records数组和total总数)<br>message(字符串,提示信息) | 401: 未授权 |
| /api/admin/borrowing | GET | page(整数,选填,页码,默认1)<br>limit(整数,选填,每页条数,默认10)<br>status(字符串,选填,状态:已归还/未归还)<br>username(字符串,选填,用户名)<br>startDate(日期,选填,开始日期)<br>endDate(日期,选填,结束日期) | code(整数,状态码)<br>data(对象,包含records数组和total总数)<br>message(字符串,提示信息) | 401: 未授权<br>403: 权限不足 |
| /api/admin/borrowing/return/{recordId} | POST | recordId(整数,必填,借阅记录ID,路径参数) | code(整数,状态码)<br>data(对象,更新后的借阅记录)<br>message(字符串,提示信息) | 401: 未授权<br>403: 权限不足<br>404: 借阅记录不存在<br>409: 此书已归还 |
| /api/admin/borrowing/stats | GET | 无 | code(整数,状态码)<br>data(对象,借阅统计数据)<br>message(字符串,提示信息) | 401: 未授权<br>403: 权限不足 |

**4. 用户管理模块（管理员）**

| **接口路径** | **方法** | **请求参数** | **响应结构** | **错误码** |
| --- | --- | --- | --- | --- |
| /api/admin/users | GET | page(整数,选填,页码,默认1)<br>limit(整数,选填,每页条数,默认10)<br>query(字符串,选填,搜索关键词)<br>role(字符串,选填,角色过滤) | code(整数,状态码)<br>data(对象,包含users数组和total总数)<br>message(字符串,提示信息) | 401: 未授权<br>403: 权限不足 |
| /api/admin/users/{id} | GET | id(整数,必填,用户ID,路径参数) | code(整数,状态码)<br>data(对象,用户详情)<br>message(字符串,提示信息) | 401: 未授权<br>403: 权限不足<br>404: 用户不存在 |
| /api/admin/users | POST | username(字符串,必填,用户名)<br>password(字符串,必填,密码)<br>email(字符串,必填,邮箱)<br>name(字符串,选填,姓名)<br>phone(字符串,选填,电话)<br>role(字符串,必填,角色)<br>status(字符串,必填,状态) | code(整数,状态码)<br>data(对象,创建的用户)<br>message(字符串,提示信息) | 400: 请求参数错误<br>401: 未授权<br>403: 权限不足<br>409: 用户名已存在 |
| /api/admin/users/{id} | PUT | id(整数,必填,用户ID,路径参数)<br>email(字符串,选填,邮箱)<br>name(字符串,选填,姓名)<br>phone(字符串,选填,电话)<br>role(字符串,选填,角色)<br>status(字符串,选填,状态) | code(整数,状态码)<br>data(对象,更新后的用户)<br>message(字符串,提示信息) | 400: 请求参数错误<br>401: 未授权<br>403: 权限不足<br>404: 用户不存在 |
| /api/admin/users/{id} | DELETE | id(整数,必填,用户ID,路径参数) | code(整数,状态码)<br>message(字符串,提示信息) | 401: 未授权<br>403: 权限不足<br>404: 用户不存在<br>409: 无法删除管理员账户 |
| /api/admin/users/{id}/status | PUT | id(整数,必填,用户ID,路径参数)<br>status(字符串,必填,状态:active/inactive) | code(整数,状态码)<br>data(对象,更新后的用户)<br>message(字符串,提示信息) | 400: 请求参数错误<br>401: 未授权<br>403: 权限不足<br>404: 用户不存在 |

**🔸【二】模块功能结构说明**

**1. 登录注册模块**

| **功能点** | **调用接口** | **触发条件** | **调用顺序** |
| --- | --- | --- | --- |
| 用户登录 | /api/auth/login | 用户在登录页填写表单点击登录按钮 | 1. 登录验证<br>2. 成功后获取用户信息 |
| 用户注册 | /api/auth/register | 用户在注册页填写表单点击注册按钮 | 1. 注册<br>2. 成功后跳转到登录页 |
| 退出登录 | /api/auth/logout | 用户点击退出登录按钮 | 1. 请求退出<br>2. 清除本地存储的token |
| 获取个人信息 | /api/auth/user | 1. 用户登录后自动请求<br>2. 访问个人中心页面时 | 要求用户已登录 |
| 修改个人信息 | /api/auth/profile | 用户在个人中心页修改信息后提交 | 要求用户已登录 |
| 修改密码 | /api/auth/password | 用户在个人中心页修改密码后提交 | 要求用户已登录 |

**2. 图书管理模块**

| **功能点** | **调用接口** | **触发条件** | **调用顺序** |
| --- | --- | --- | --- |
| 获取图书列表 | /api/books | 1. 访问图书列表页<br>2. 应用筛选条件<br>3. 切换分页 | 可能与分类接口同时调用 |
| 获取图书详情 | /api/books/{id} | 访问图书详情页 | 无特殊顺序要求 |
| 搜索图书 | /api/books/search 或 /api/books?search=xxx | 用户在搜索框输入并提交 | 无特殊顺序要求 |
| 获取图书分类 | /api/books/categories | 1. 访问图书列表页<br>2. 加载分类筛选器 | 通常与获取图书列表同时调用 |
| 添加图书(管理员) | /api/admin/books | 管理员在图书管理页点击添加图书并填写表单 | 要求管理员已登录 |
| 编辑图书(管理员) | /api/admin/books/{id} (PUT) | 管理员在图书管理页点击编辑图书并提交修改 | 要求管理员已登录 |
| 删除图书(管理员) | /api/admin/books/{id} (DELETE) | 管理员在图书管理页点击删除图书并确认 | 要求管理员已登录 |

**3. 借阅管理模块**

| **功能点** | **调用接口** | **触发条件** | **调用顺序** |
| --- | --- | --- | --- |
| 借阅图书 | /api/borrowing/borrow | 用户在图书详情页点击借阅按钮 | 要求用户已登录 |
| 归还图书 | /api/borrowing/return/{recordId} | 用户在借阅记录页点击归还按钮 | 要求用户已登录 |
| 查看个人借阅记录 | /api/borrowing/user | 1. 访问个人借阅记录页<br>2. 切换分页 | 要求用户已登录 |
| 管理员查看所有借阅记录 | /api/admin/borrowing | 1. 管理员访问借阅记录管理页<br>2. 应用筛选条件<br>3. 切换分页 | 要求管理员已登录 |
| 管理员标记归还 | /api/admin/borrowing/return/{recordId} | 管理员在借阅记录管理页点击标记已归还 | 要求管理员已登录 |
| 获取借阅统计数据 | /api/admin/borrowing/stats | 管理员访问管理仪表盘页面 | 要求管理员已登录 |

**4. 用户管理模块（管理员）**

| **功能点** | **调用接口** | **触发条件** | **调用顺序** |
| --- | --- | --- | --- |
| 获取用户列表 | /api/admin/users | 1. 管理员访问用户管理页<br>2. 应用筛选条件<br>3. 切换分页 | 要求管理员已登录 |
| 获取用户详情 | /api/admin/users/{id} | 管理员点击查看用户详情 | 要求管理员已登录 |
| 添加用户 | /api/admin/users (POST) | 管理员在用户管理页点击添加用户并填写表单 | 要求管理员已登录 |
| 编辑用户 | /api/admin/users/{id} (PUT) | 管理员在用户管理页点击编辑用户并提交修改 | 要求管理员已登录 |
| 删除用户 | /api/admin/users/{id} (DELETE) | 管理员在用户管理页点击删除用户并确认 | 要求管理员已登录 |
| 更新用户状态 | /api/admin/users/{id}/status | 管理员在用户管理页切换用户状态开关 | 要求管理员已登录 |

**🔸【三】认证与状态管理机制**

**登录认证流程**

1. **认证机制**：本系统使用基于JWT (JSON Web Token) 的认证机制
2. **Token存储**：
   * 登录成功后，后端生成的token存储在浏览器的localStorage中
   * 键名：token
3. **Token携带**：
   * 使用Axios拦截器自动为每个请求添加Authorization请求头
   * 格式：Bearer {token}，例如：Authorization: Bearer eyJhbGciOiJIUzI1NiI...
4. **拦截器实现**：

JavaScript

// API请求拦截器添加认证头

api.interceptors.request.use(

config => {

const token = localStorage.getItem('token')

if (token) {

config.headers['Authorization'] = `Bearer ${token}`

}

return config

},

error => {

console.error('Request error:', error)

return Promise.reject(error)

}

)

1. **错误处理**：
   * 使用Axios响应拦截器处理认证相关错误
   * 401 (未授权)：清除本地存储的token和用户信息，重定向到登录页面
   * 403 (禁止访问)：显示权限不足提示
   * 其他错误：显示相应的错误提示

JavaScript

// API响应拦截器处理错误

api.interceptors.response.use(

response => { /\* 省略正常响应处理 \*/ },

error => {

if (error.response && error.response.status === 401) {

// Token失效或未登录，清除用户信息并跳转登录页

localStorage.removeItem('token')

localStorage.removeItem('user')

window.location.href = '/login'

}

return Promise.reject(error)

}

)

1. **登录与注销流程**：
   * **登录**：提交用户名密码 → 后端验证并返回token → 前端保存token到localStorage → 获取用户信息
   * **注销**：调用注销接口 → 后端使token失效 → 前端清除localStorage中的token和用户信息

**🔸【四】分页与筛选**

**分页参数说明**

1. **通用分页参数**：
   * page：当前页码，整数，从1开始
   * limit：每页条数，整数
2. **分页响应格式**：

JSON

{

"code": 200,

"data": {

"records": [...], // 当前页的数据记录

"total": 100, // 总记录数

"page": 1, // 当前页码

"limit": 10 // 每页条数

},

"message": "成功"

}

**筛选参数说明**

1. **图书筛选参数**：
   * search：搜索关键词，匹配书名、作者、描述
   * category：图书分类
   * sortBy：排序字段，格式如 field:order，例如 title:asc
   * inStock：是否有库存，布尔值
2. **借阅记录筛选参数**：
   * status：借阅状态，已归还或未归还
   * username：用户名
   * startDate和endDate：日期范围，格式：yyyy-MM-dd
3. **用户筛选参数**：
   * query：搜索关键词，匹配用户名、姓名、邮箱
   * role：用户角色，如 admin或user

**示例请求**

1. **分页请求示例**：

Code

GET /api/books?page=2&limit=20

1. **综合筛选请求示例**：

Code

GET /api/books?page=1&limit=10&search=Java&category=技术&sortBy=title:asc&inStock=true

1. **日期范围筛选示例**：

Code

GET /api/admin/borrowing?startDate=2025-01-01&endDate=2025-06-01&status=已归还

**🔸【五】Vuex 中维护的重要状态**

**用户模块 (user)**

JavaScript

// 用户模块状态结构

state: {

isLoggedIn: Boolean, // 是否已登录

user: Object, // 用户信息

token: String, // JWT令牌

role: String // 用户角色（admin/user）

}

**重要状态字段**：

* isLoggedIn：用户登录状态，后端需要验证token有效性
* token：身份令牌，后端解析验证以确认用户身份和权限
* role：用户角色，用于前端权限控制，后端也需要验证
* user：用户详细信息，包含用户名、姓名、邮箱等

**状态获取方式**：

* 登录成功后从/api/auth/login响应中获取token和基本用户角色
* 详细用户信息通过/api/auth/user接口获取

**图书模块 (books)**

JavaScript

// 图书模块状态结构

state: {

books: Array, // 图书列表

categories: Array, // 图书分类

loading: Boolean, // 加载状态

currentBook: Object, // 当前查看的图书

totalBooks: Number // 图书总数

}

**重要状态字段**：

* books：图书列表数据，从/api/books获取
* categories：分类列表，从/api/books/categories获取
* currentBook：当前查看的图书详情，从/api/books/{id}获取

**借阅模块 (borrowing)**

JavaScript

// 借阅模块状态结构

state: {

borrowRecords: Array, // 借阅记录列表

loading: Boolean, // 加载状态

stats: Object, // 借阅统计数据

totalRecords: Number // 记录总数

}

**重要状态字段**：

* borrowRecords：借阅记录列表，普通用户从/api/borrowing/user获取，管理员从/api/admin/borrowing获取
* stats：借阅统计数据，管理员从/api/admin/borrowing/stats获取

**管理员模块 (admin)**

JavaScript

// 管理员模块状态结构

state: {

users: Array, // 用户列表

loading: Boolean, // 加载状态

totalUsers: Number // 用户总数

}

**重要状态字段**：

* users：用户列表，从/api/admin/users获取
* totalUsers：用户总数，从用户列表接口的响应中获取

**后端验证要求**

1. **Token验证**：
   * 除登录注册外的所有接口都需要验证token
   * 验证token的有效性和是否过期
   * 无效或过期token返回401错误
2. **权限验证**：
   * 对于/api/admin/前缀的接口，后端需要验证用户是否具有管理员权限
   * 权限不足返回403错误
3. **数据一致性验证**：
   * 用户只能查看自己的借阅记录
   * 管理员可以查看所有借阅记录
   * 特定操作需要验证记录归属权
   * 用户只能归还自己借阅的图书
   * 管理员处理借阅时需验证图书和借阅记录的存在性
   * 借阅图书时需验证库存充足
   * **业务逻辑验证**：
   * 借阅图书时需检查用户是否有未归还的同本图书
   * 借阅图书时需检查用户未归还图书总数是否达到上限（如5本）
   * 归还图书时需验证借阅状态是否为"未归还"
   * 新增或修改图书信息时进行数据合法性验证
   * **前后端状态同步**：
   * 每次用户操作后，确保返回最新状态以更新前端Vuex
   * 管理员操作后，可能需要通知相关用户（如借阅被取消等情况）
   * **状态持久化**
   * **持久化方式**：
   * 用户登录状态和token存储在localStorage中
   * 用户注销时清除localStorage中的相关信息
   * **需要持久化的关键信息**：
   * token：用户身份验证令牌
   * user：基本用户信息
   * role：用户角色
   * **🔸【六】数据结构与对象模型**
   * 为了确保前后端数据交互的一致性，这里定义主要数据对象的结构：
   * **1. 用户对象 (User)**
   * JSON
   * {
   * "id": 1, // 用户ID
   * "username": "zhangsan", // 用户名
   * "name": "张三", // 真实姓名
   * "email": "zhangsan@example.com", // 邮箱
   * "phone": "13800138000", // 电话
   * "address": "北京市海淀区", // 地址
   * "role": "user", // 角色：user/admin
   * "status": "active", // 状态：active/inactive
   * "createdAt": "2025-01-15T08:30:00Z" // 创建时间
   * }
   * **2. 图书对象 (Book)**
   * JSON
   * {
   * "id": 101, // 图书ID
   * "title": "Java编程思想", // 书名
   * "author": "埃克尔", // 作者
   * "category": "技术", // 分类
   * "description": "Java经典教程...", // 描述
   * "isbn": "9787111213826", // ISBN
   * "publisher": "机械工业出版社", // 出版社
   * "publishDate": "2007-06-01", // 出版日期
   * "pages": 880, // 页数
   * "cover": "/assets/covers/101.jpg", // 封面图片URL
   * "stock": 5, // 库存数量
   * "createdAt": "2025-01-15T08:30:00Z", // 创建时间
   * "updatedAt": "2025-05-20T10:15:00Z" // 更新时间
   * }
   * **3. 借阅记录对象 (BorrowRecord)**
   * JSON
   * {
   * "id": 1001, // 借阅记录ID
   * "userId": 1, // 用户ID
   * "username": "zhangsan", // 用户名（冗余，方便管理员查看）
   * "bookId": 101, // 图书ID
   * "bookTitle": "Java编程思想", // 书名（冗余，防止图书信息变更）
   * "borrowDate": "2025-06-01", // 借阅日期
   * "dueDate": "2025-07-01", // 应还日期
   * "returnDate": null, // 归还日期，未归还为null
   * "status": "未归还", // 状态：已归还/未归还
   * "createdAt": "2025-06-01T09:30:00Z" // 创建时间
   * }
   * **4. 统计数据对象 (Stats)**
   * JSON
   * {
   * "totalBooks": 120, // 图书总数
   * "availableBooks": 98, // 可借阅图书数
   * "totalUsers": 56, // 用户总数
   * "activeUsers": 42, // 活跃用户数
   * "totalBorrowings": 230, // 总借阅次数
   * "currentBorrowings": 22, // 当前借出数量
   * "overdueBorrowings": 3, // 逾期未还数量
   * "recentBorrowings": [ // 最近借阅记录
   * {
   * "date": "2025-06-08",
   * "count": 5
   * },
   * {
   * "date": "2025-06-07",
   * "count": 3
   * },
   * // 更多日期...
   * ],
   * "popularBooks": [ // 热门图书
   * {
   * "id": 101,
   * "title": "Java编程思想",
   * "borrowCount": 15
   * },
   * // 更多图书...
   * ],
   * "categoryDistribution": [ // 分类分布
   * {
   * "category": "技术",
   * "count": 45
   * },
   * {
   * "category": "文学",
   * "count": 38
   * },
   * // 更多分类...
   * ]
   * }
   * **🔸【七】响应格式规范**
   * 为确保前后端交互的一致性，后端API应遵循以下响应格式:
   * **1. 成功响应**
   * JSON
   * {
   * "code": 200, // 状态码：200表示成功
   * "data": { // 数据对象，根据接口不同而变化
   * // 接口返回的具体数据
   * },
   * "message": "操作成功" // 成功提示信息
   * }
   * **2. 分页数据响应**
   * JSON
   * {
   * "code": 200,
   * "data": {
   * "records": [ // 数据数组
   * // 数据项列表
   * ],
   * "total": 100, // 总记录数
   * "page": 1, // 当前页码
   * "limit": 10 // 每页条数
   * },
   * "message": "查询成功"
   * }
   * **3. 错误响应**
   * JSON
   * {
   * "code": 400, // HTTP错误码
   * "message": "请求参数错误", // 错误提示信息
   * "data": null // 错误情况下通常为null
   * }
   * **4. 常用HTTP状态码**
   * **200**: 请求成功
   * **400**: 请求参数错误
   * **401**: 未授权（未登录或token无效）
   * **403**: 禁止访问（权限不足）
   * **404**: 资源不存在
   * **409**: 冲突（如用户名已存在）
   * **429**: 请求过于频繁（如登录失败次数过多）
   * **500**: 服务器内部错误
   * **🔸【八】安全考虑**
   * 前端项目中已实现的安全机制，后端需要配合实现：
   * **密码安全**：
   * 密码应在后端使用BCrypt或PBKDF2等算法加密存储
   * 禁止在响应中返回密码信息，即使是加密后的密码
   * **登录保护**：
   * 实现登录失败次数限制，超过阈值（如5次）后暂时锁定账户或IP
   * 提供密码重置功能
   * **CSRF保护**：
   * 使用基于令牌的认证机制（JWT）

**权限控制**：

* + 实现细粒度的权限检查，确保用户只能访问和修改自己的资源
  + 管理员操作需要特权验证
  + **数据验证**：
  + 所有用户输入必须经过后端验证，不能只依赖前端验证
  + 特别注意特殊字符处理，防止XSS和SQL注入
  + **日志记录**：
  + 记录关键操作日志，特别是管理员操作
  + 记录异常登录尝试
  + 记录借阅和归还操作