接口文档

每个接口包含使用方法、请求参数、返回数据、url示例

目录

[1.1 管理员登录 3](#_Toc2118)

[2.1 获取所有书籍 3](#_Toc27510)

[2.2 根据ID获取书籍 4](#_Toc9118)

[2.3 创建书籍 5](#_Toc12521)

[2.4 更新书籍信息 6](#_Toc5873)

[2.5 删除书籍 7](#_Toc28044)

[2.6 按分类分页查询书籍 8](#_Toc30804)

[2.7 按书名搜索书籍 9](#_Toc29218)

[2.8 上传书籍 10](#_Toc9334)

[2.9 更新书籍费用和价格 11](#_Toc9135)

[2.10 获取阅读次数最高的四本书 12](#_Toc17176)

[2.11. 获取小说封面图片 13](#_Toc11922)

[2.12 获取小说文本内容 14](#_Toc27698)

[3.1 获取用户购买记录 15](#_Toc23446)

[3.2 获取所有购买记录 16](#_Toc23381)

[3.3 更新购买记录 17](#_Toc24136)

[4.1 获取用户VIP购买记录 18](#_Toc19211)

[4.2 获取所有VIP购买记录 19](#_Toc28510)

[4.3 购买VIP 20](#_Toc8257)

[5.1 获取所有用户 21](#_Toc23282)

[5.2 根据ID获取用户 22](#_Toc3587)

[5.3 创建用户 23](#_Toc10706)

[5.4 更新用户信息 24](#_Toc23567)

[5.5 删除用户 25](#_Toc1606)

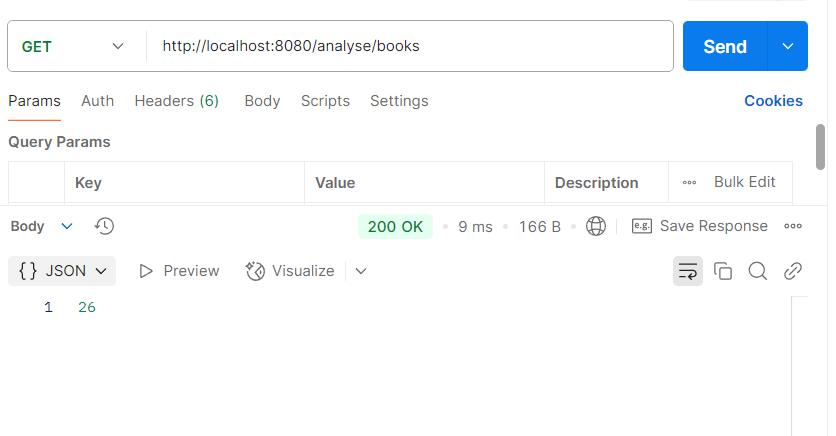
[6.1 记录阅读行为 26](#_Toc3852)

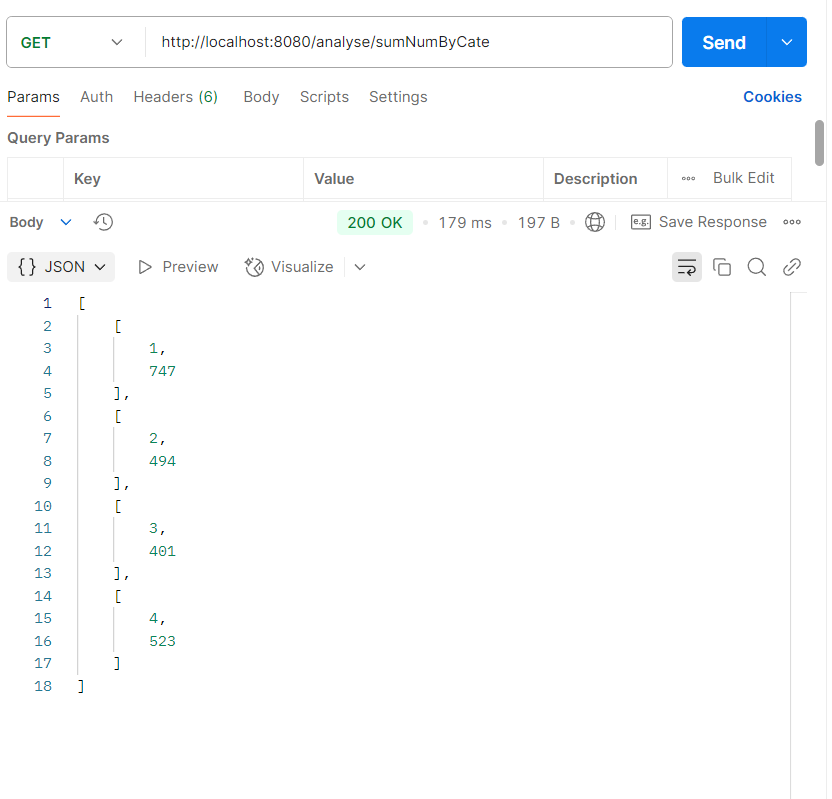
[6.2 查询个人阅读历史 27](#_Toc16363)

[7.1 获取总用户数量 28](#_Toc26471)

[7.2 获取VIP用户数量 29](#_Toc12192)

[7.3 获取书籍总数 29](#_Toc5151)

[7.4 按类别统计书籍数量 30](#_Toc915)

[7.5 按类别统计消费金额 31](#_Toc8467)

### 1. 管理员模块

1.1 管理员登录

接口地址：POST /admin/login

功能描述：管理员登录接口，验证用户名和密码。

请求参数：

username (String): 管理员用户名。

password (String): 管理员密码。

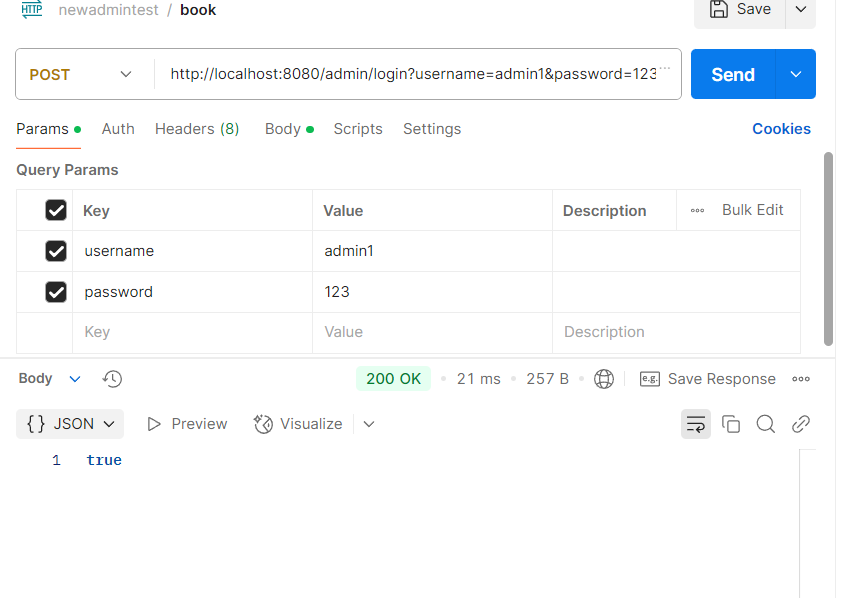
返回数据：

成功：true

失败：false

示例：

POST "http://localhost:8080/admin/login?username=admin1&password=123"



### 2. 书籍模块

2.1 获取所有书籍

接口地址：GET /admin/books

功能描述：获取所有书籍的列表。

请求参数：无

返回数据：书籍列表，每本书包含以下字段：

id (Integer): 书籍ID。

name (String): 书籍名称。

category (Integer): 书籍分类ID。

author (String): 作者。

fee (Integer): 种类。

money (Float): 书籍价格。

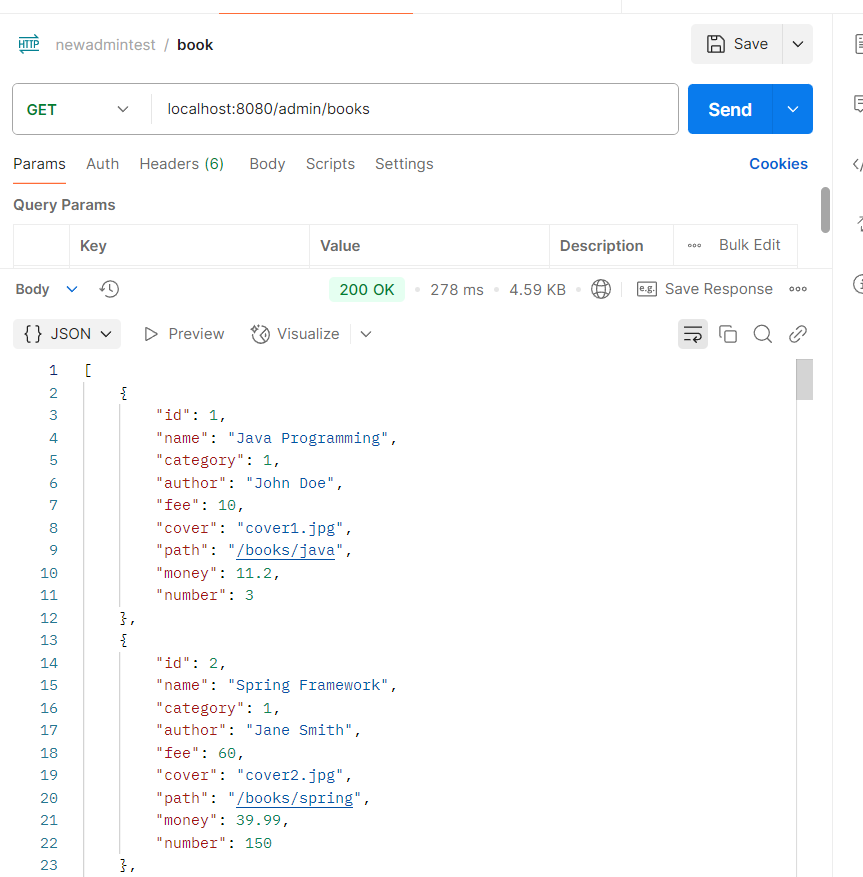
number (Integer): 数量。

path (String): 书籍文件路径。

cover (String): 书籍封面路径。

示例：

GET "http://localhost:8080/admin/books"



2.2 根据ID获取书籍

接口地址：GET /admin/books/{id}

功能描述：根据书籍ID获取书籍详情。

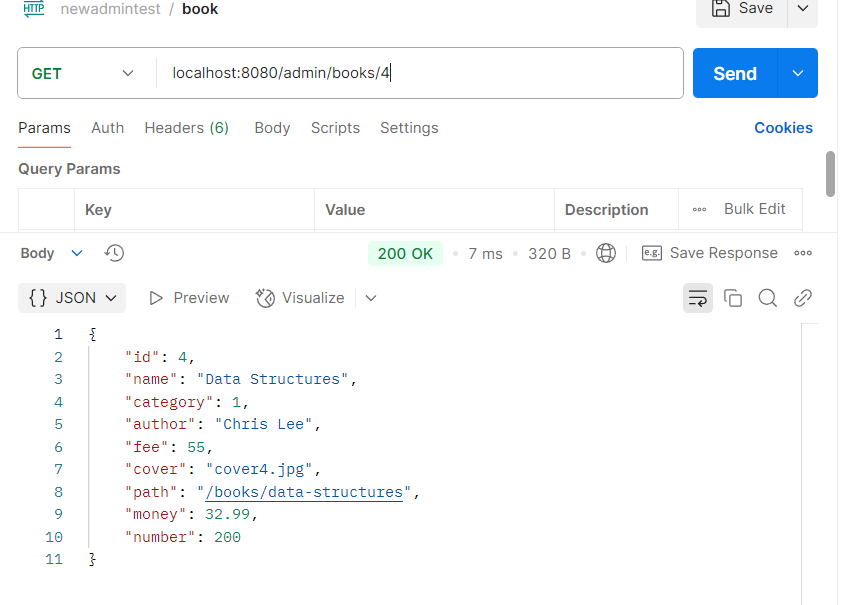
请求参数：

id (Integer): 书籍ID。

返回数据：书籍详情，字段同上。

示例：

GET "http://localhost:8080/admin/books/1"



2.3 创建书籍

接口地址：POST /admin/books

功能描述：创建新书籍。

请求参数：JSON 格式，包含以下字段：

name (String): 书籍名称。

category (Integer): 分类ID。

author (String): 作者。

fee (Integer): 种类。

money (Float): 书籍价格。

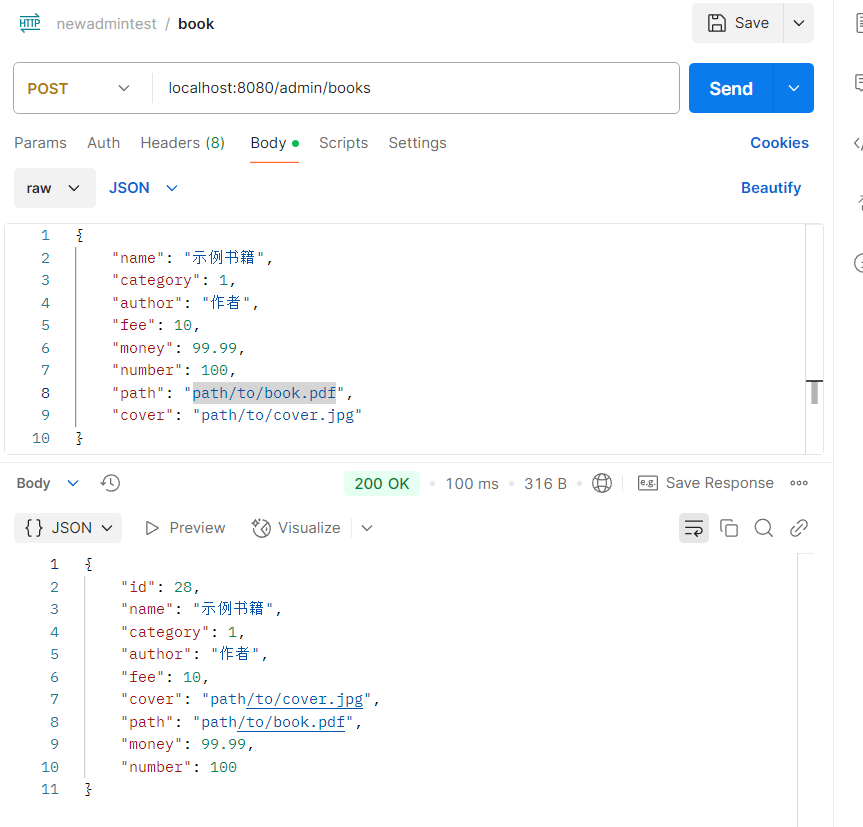
number (Integer): 数量。

path (String): 书籍文件路径。

cover (String): 书籍封面路径。

返回数据：创建的书籍对象。

示例：

POST"http://localhost:8080/admin/books" 

2.4 更新书籍信息

接口地址：PUT /admin/books/write/{id}

功能描述：更新书籍信息。

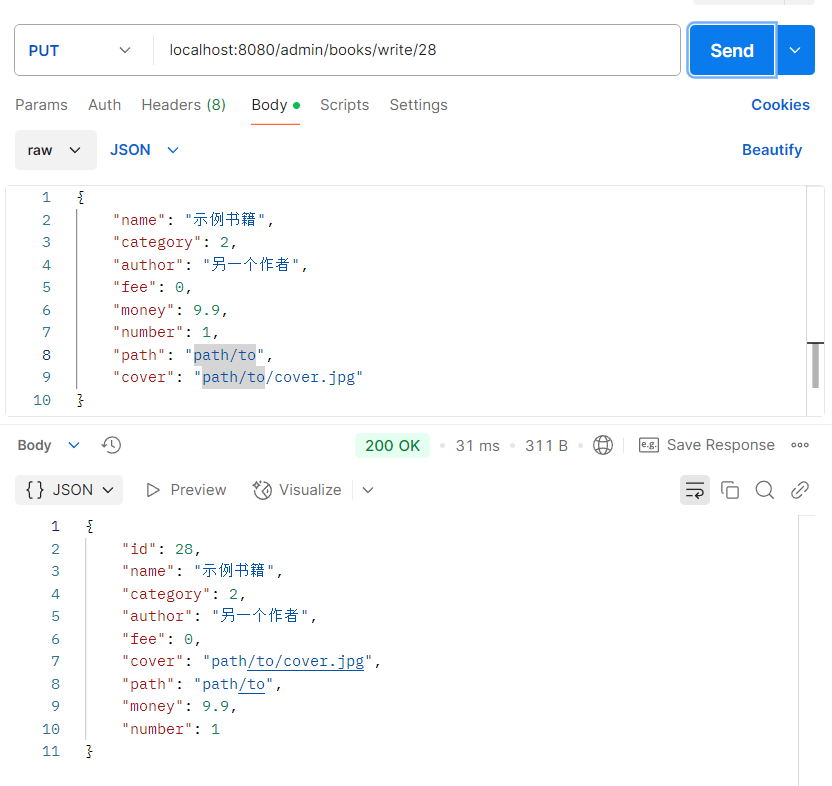
请求参数：

id (Integer): 书籍ID。

JSON 格式，包含需要更新的字段。

返回数据：更新后的书籍对象。

示例：

PUT"http://localhost:8080/admin/books/write/28" 

2.5 删除书籍

接口地址：DELETE /admin/books/{id}

功能描述：根据书籍ID删除书籍。

请求参数：

id (Integer): 书籍ID。

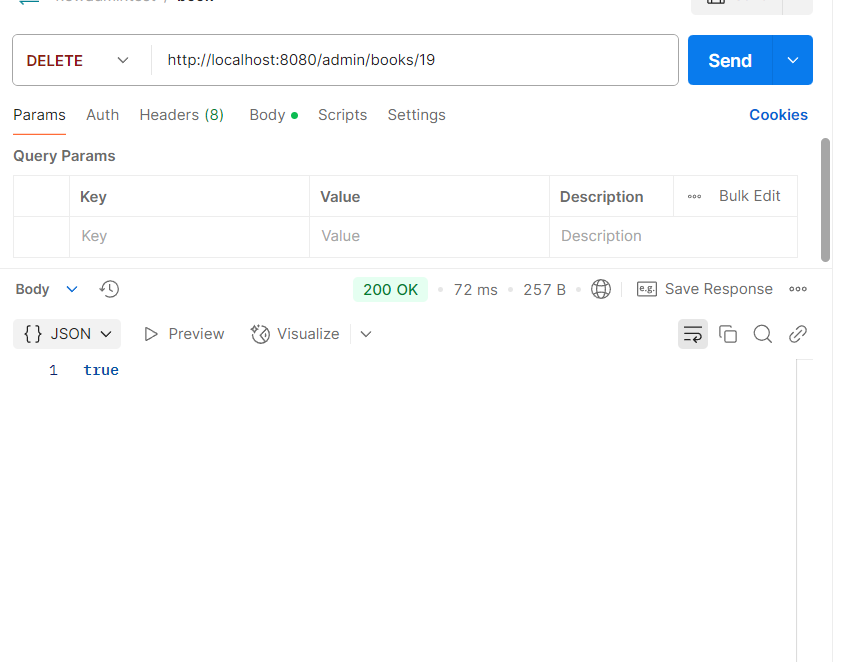
返回数据：

成功：true。

失败：false。

示例：

DELETE "http://localhost:8080/admin/books/19"



2.6 按分类查询书籍

接口地址：GET /admin/books/{category}/id/books

功能描述：按分类查询书籍。

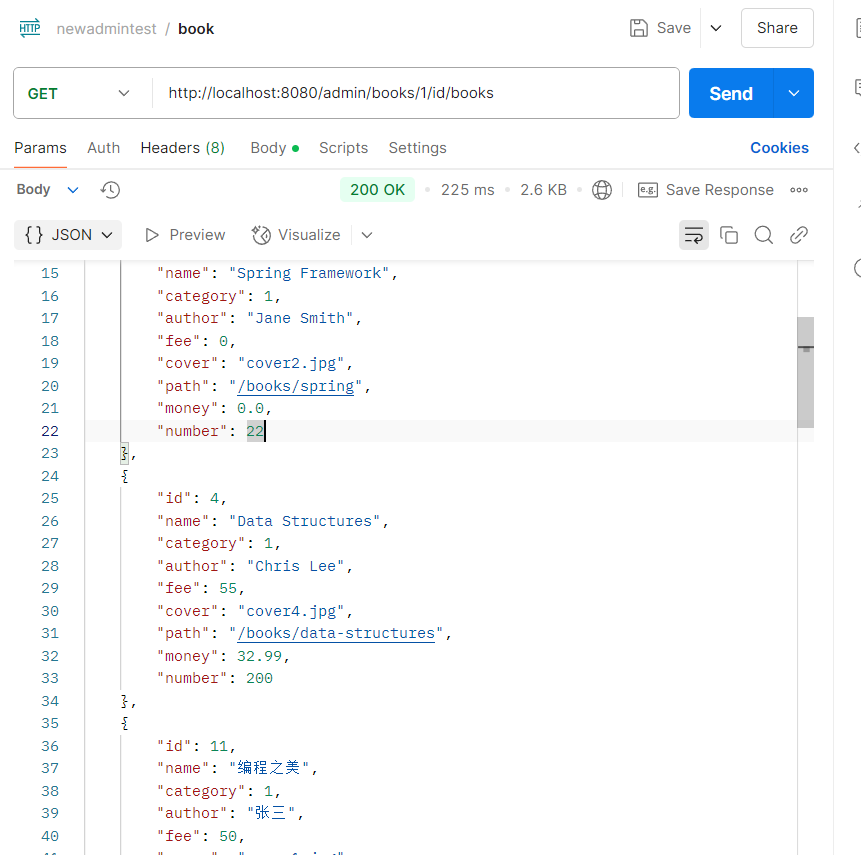
请求参数：

category (Integer): 分类ID。

返回数据：分页结果，包含书籍列表和分页信息。

示例：

GET "http://localhost:8080/admin/books/1/id/books"



2.7 按书名搜索书籍

接口地址：GET /admin/books/search/{bookName}

功能描述：根据书名搜索书籍。

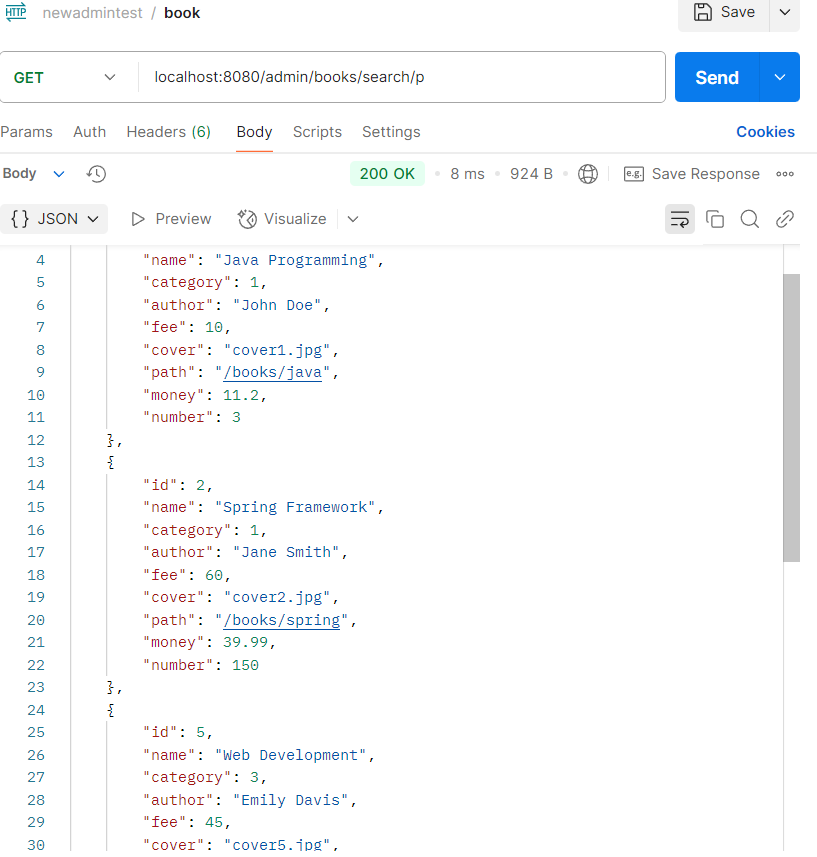
请求参数：

bookName (String): 书名关键字。

返回数据：匹配的书籍列表，字段同上。

示例：

GET "http://localhost:8080/admin/books/search/p"



2.8 上传书籍

接口地址：POST /admin/books/addbook

功能描述：上传书籍文件和封面，并创建书籍记录。

请求参数：

file (MultipartFile): 书籍文件。

coverFile (MultipartFile): 书籍封面文件。

name (String): 书籍名称。

category (Integer): 分类ID。

author (String): 作者。

fee (Integer): 种类。

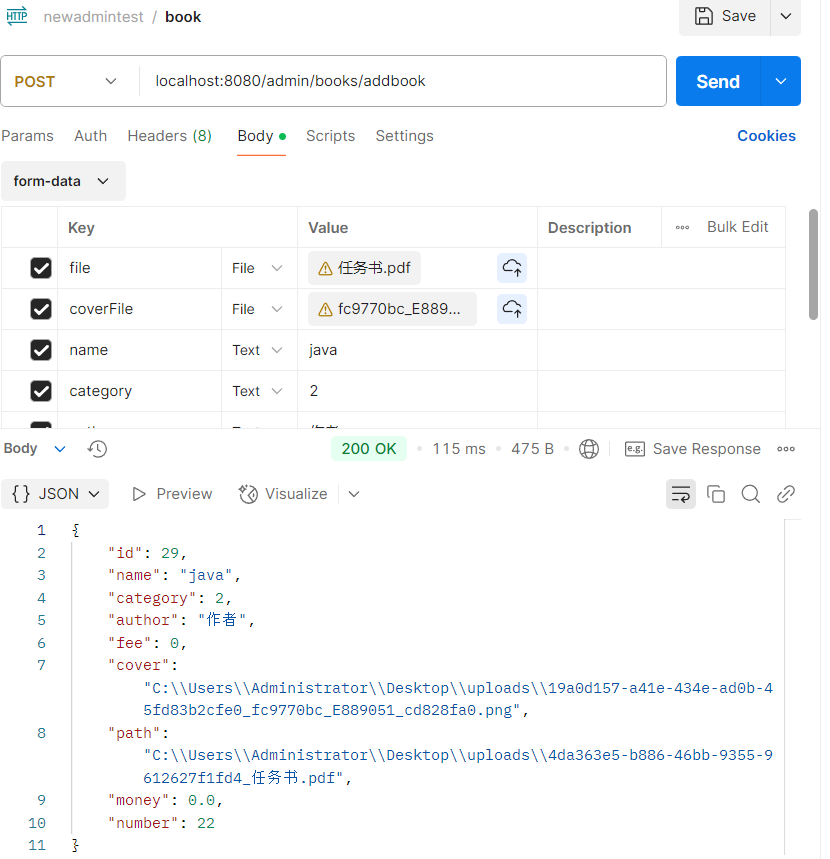
money (Float): 书籍价格。

number (Integer): 数量。

返回数据：true。

示例：

POST "http://localhost:8080/admin/books/addbook"



2.9 更新书籍费用和价格

接口地址：PUT /admin/books/charge/{id}

功能描述：更新书籍的租借费用、价格和库存数量。

请求参数：

id (Integer): 书籍ID。

fee (Integer): 新类型。

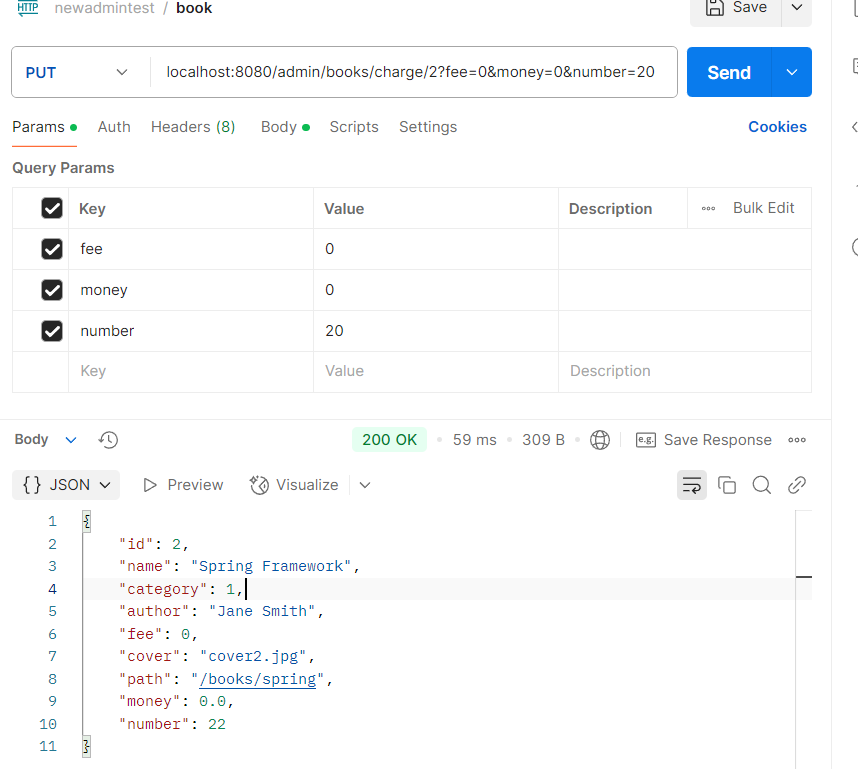
money (Float): 新的价格。

number (Integer): 新的库存数量。

返回数据：更新后的书籍对象。

示例：

PUT "localhost:8080/admin/books/charge/2?fee=0&money=0&number=20"



2.10 获取阅读次数最高的四本书

接口地址：GET /admin/books/top4

功能描述：根据书籍的阅读次数（number 字段），返回阅读次数最高的四本书作为推荐书籍。

请求参数：无

返回值：

成功：返回一个包含最多四本书的列表，每本书包含以下信息：

bookId (Integer): 书籍ID。

title (String): 书籍标题。

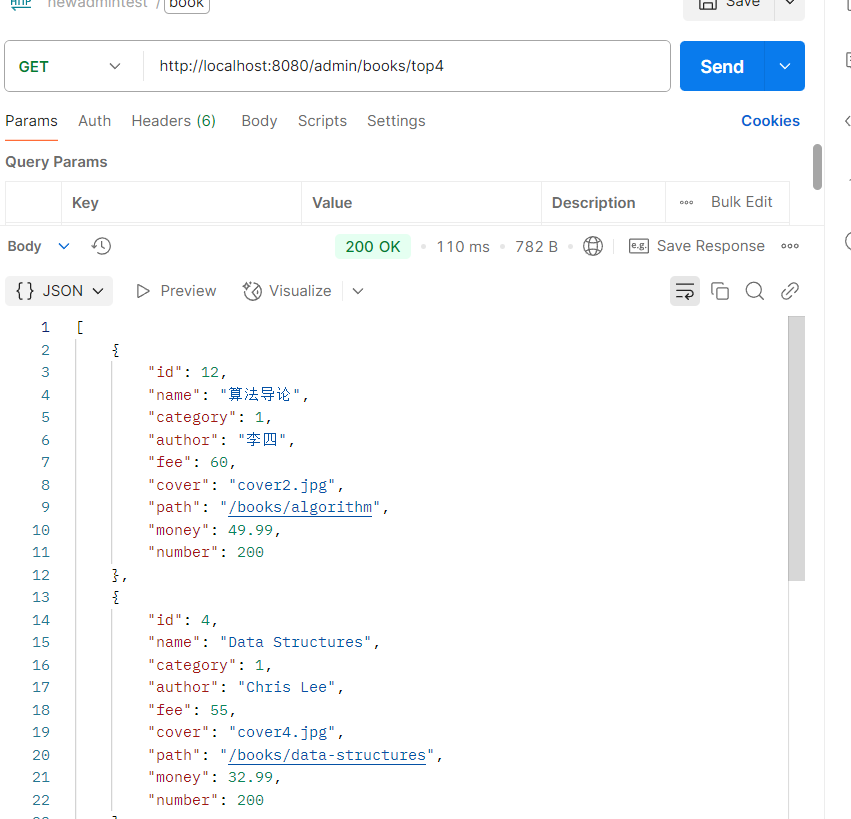
number (Integer): 书籍的阅读次数。

如果书籍数量不足四本，返回实际数量的书籍列表。

示例：

GET”http://localhost:8080/admin/books/top4”

返回示例：



2.11. 获取小说封面图片

接口地址：/cover/{id}

请求方式：GET

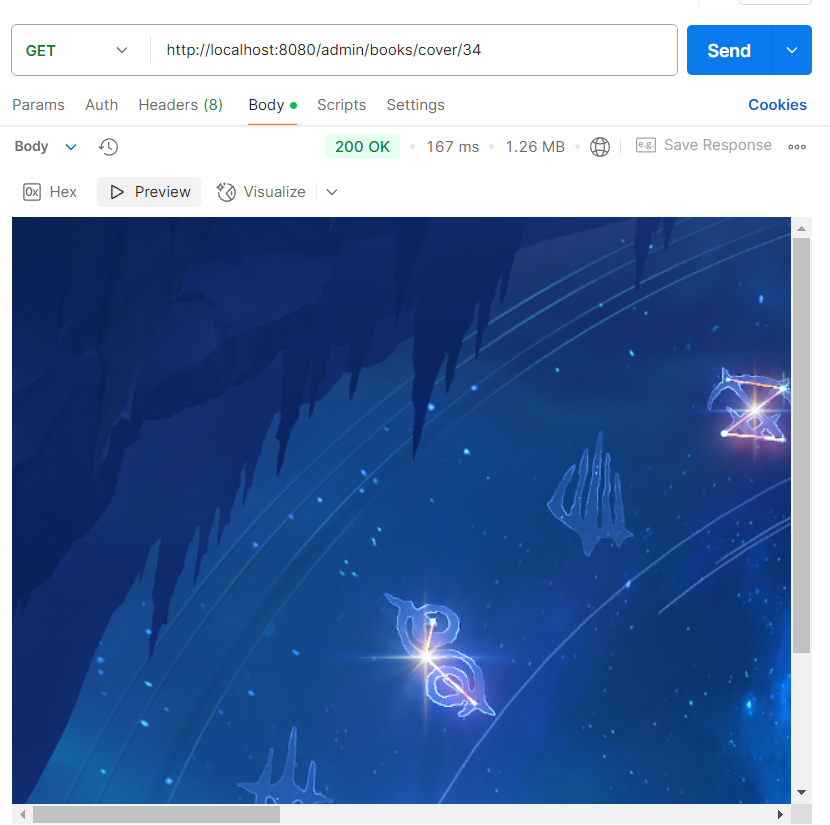
功能描述：根据小说ID获取小说封面图片。

路径参数：

id：小说的ID（Integer类型，必填）。

示例：

GET”http://localhost:8080/admin/books/cover/34”



2.12 获取小说文本内容

接口地址：/{id}/content

请求方式：GET

功能描述：根据小说ID获取小说的文本内容。

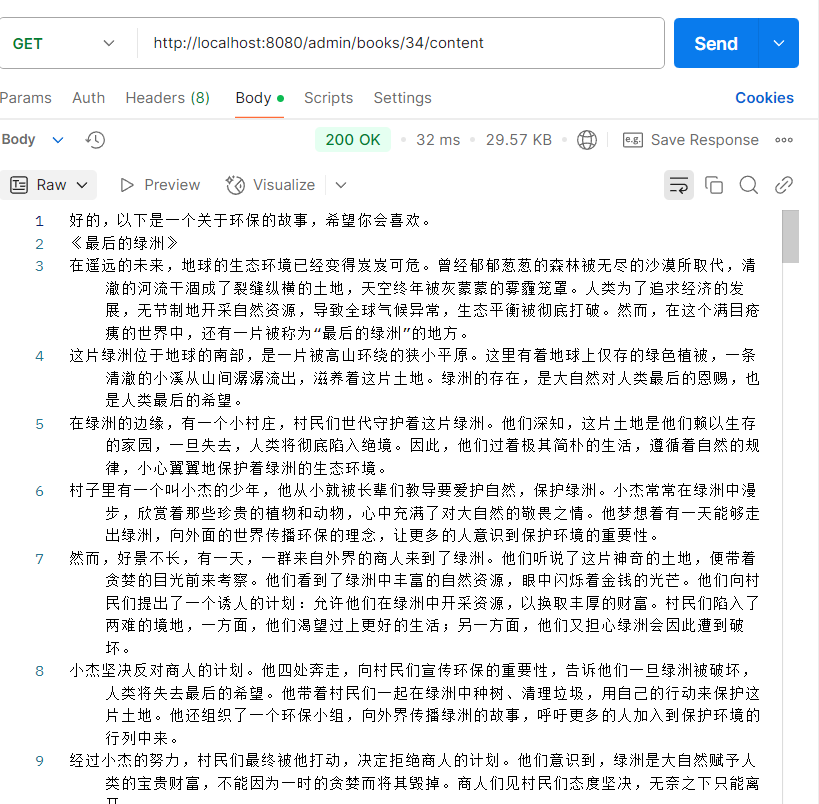
路径参数：

id：小说的ID（Integer类型，必填）。

示例：

GET “http://localhost:8080/admin/books/34/content”

返回小说ID为34的文本内容。



### 3. 购买书籍模块

3.1 获取用户购买记录

接口地址：GET /admin/buybooks/{userId}

功能描述：根据用户ID获取购买记录。

请求参数：

userId (Integer): 用户ID。

返回数据：购买记录列表，每条记录包含以下字段：

id (Integer): 记录ID。

book (Book): 书籍信息。

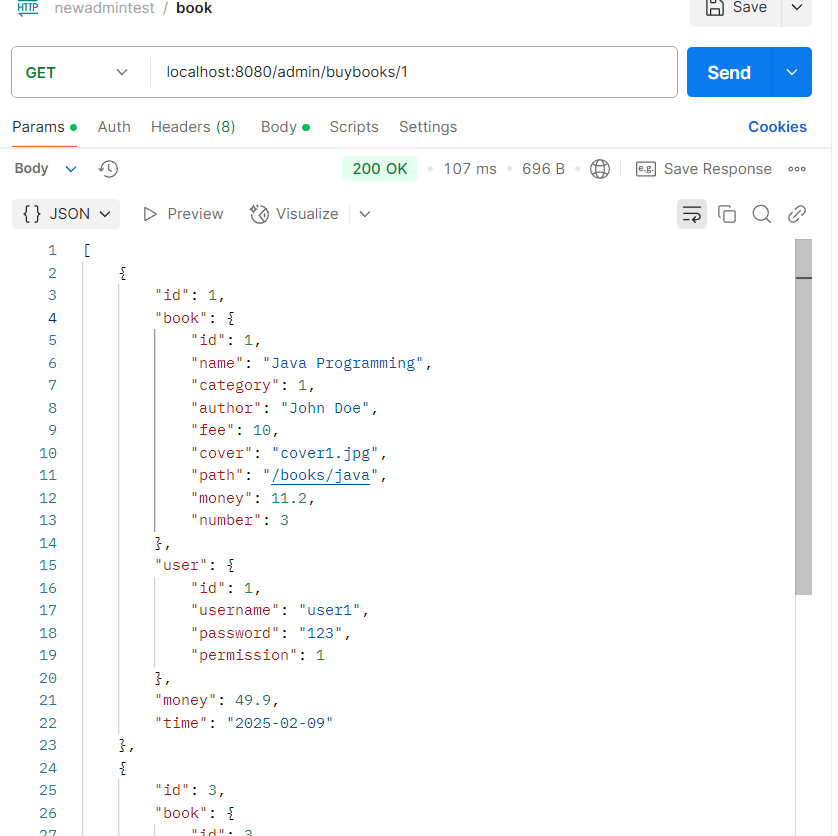
user (User): 用户信息。

money (Float): 购买金额。

time (Date): 购买时间。

示例：

GET "http://localhost:8080/admin/buybooks/1"



3.2 获取所有购买记录

接口地址：GET /admin/buybooks

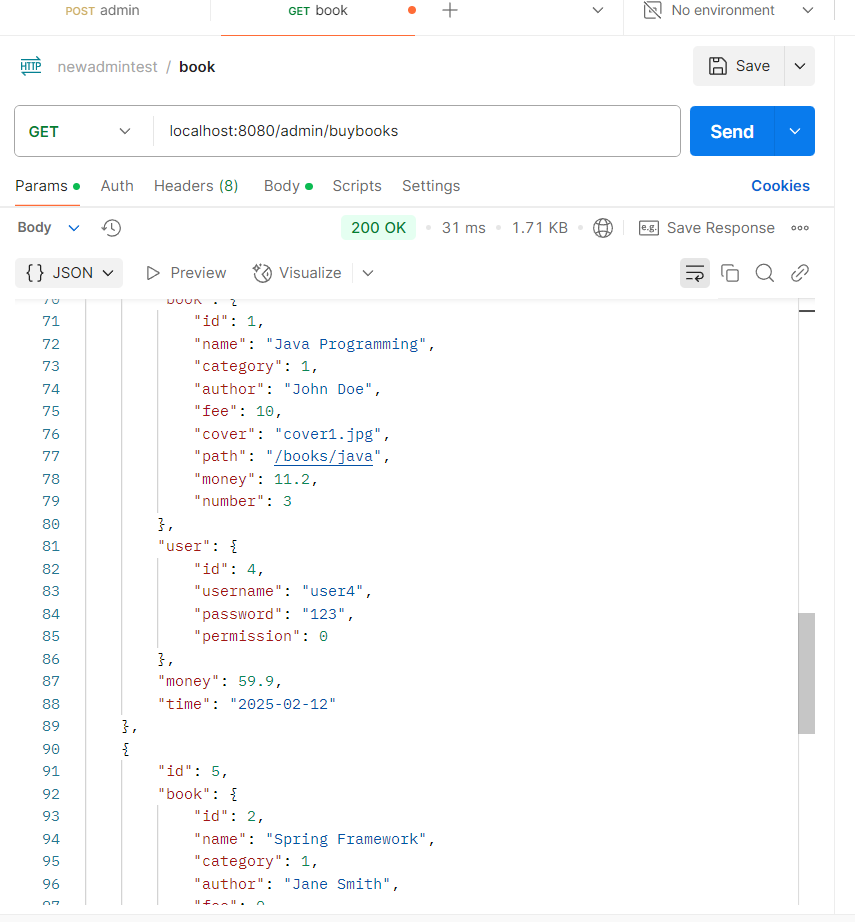
功能描述：获取所有购买记录。

请求参数：无

返回数据：购买记录列表，字段同上。

示例：

GET "http://localhost:8080/admin/buybooks"



3.3 更新购买记录

接口地址：PUT /admin/buybooks/update

功能描述：更新购买记录的金额和时间。

请求参数：

userId (Integer): 用户ID。

bookId(Integer):书籍ID

money (Float): 新的购买金额。

time (Date): 新的购买时间。

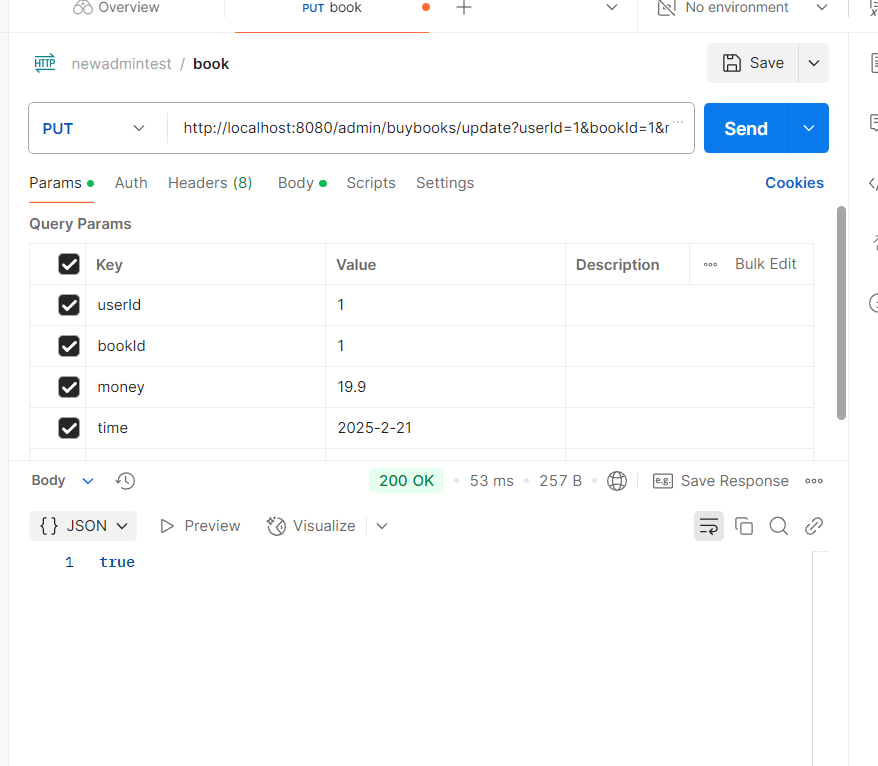
返回数据：

成功：true。

失败：false。

示例：

PUT "http://localhost:8080/admin/buybooks/update?userId=1&bookId=1&money=19.9&time=2025-2-21"



### 4. 购买VIP模块

4.1 获取用户VIP购买记录

接口地址：GET /admin/buyvip/{userId}

功能描述：根据用户ID获取VIP购买记录。

请求参数：

userId (Integer): 用户ID。

返回数据：VIP购买记录列表，每条记录包含以下字段：

id (Integer): 记录ID。

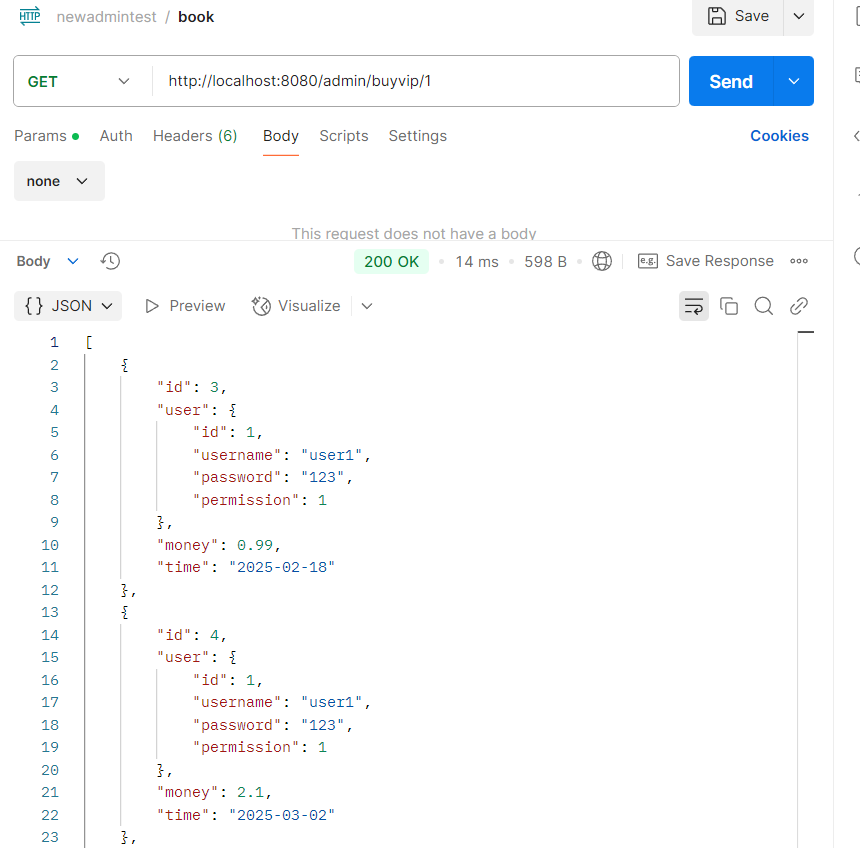
user (User): 用户信息。

money (Float): 购买金额。

time (Date): 购买时间。

示例：

GET "http://localhost:8080/admin/buyvip/1"



4.2 获取所有VIP购买记录

接口地址：GET /admin/buyvip

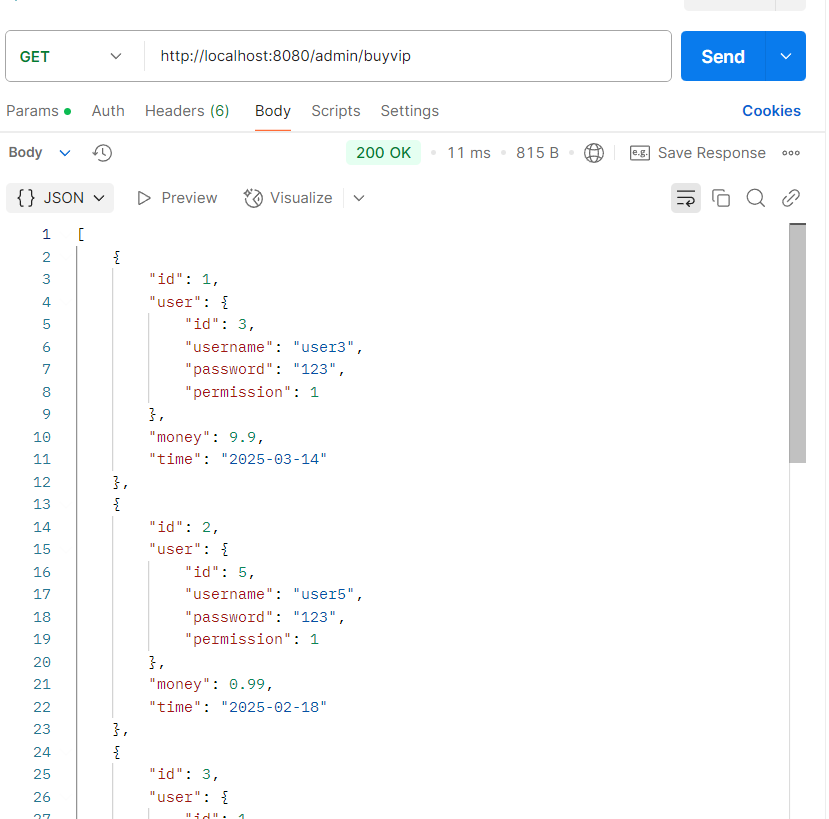
功能描述：获取所有VIP购买记录。

请求参数：无

返回数据：VIP购买记录列表，字段同上。

示例：

GET "http://localhost:8080/admin/buyvip"



4.3 购买VIP

接口地址：POST /admin/buyvip/buy

功能描述：用户购买VIP服务。

请求参数：

userId (Integer): 用户ID。

money (Float): 购买金额。

time (Date): 购买时间。

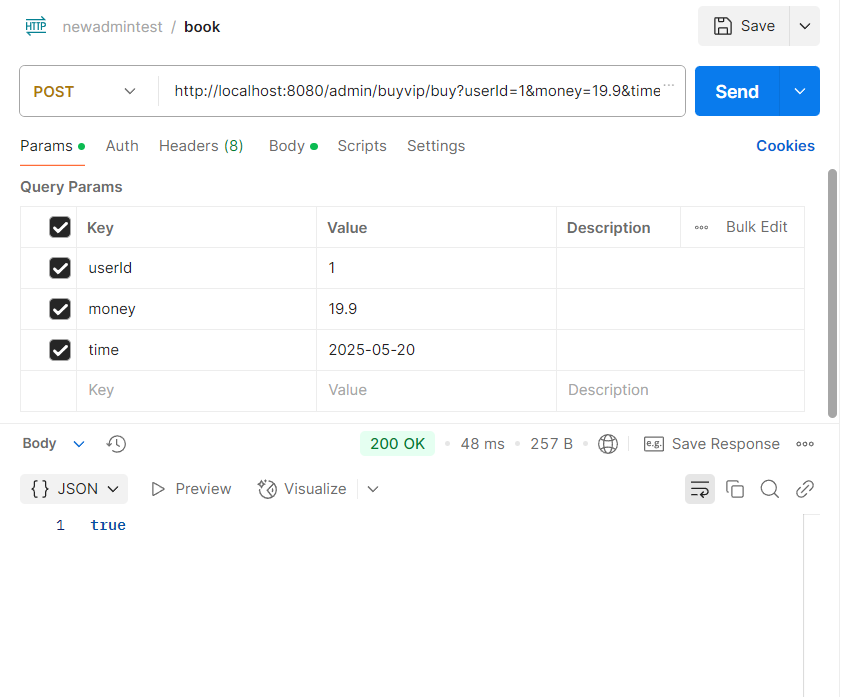
返回数据：

成功：true。

失败：false。

示例：

POST "http://localhost:8080/admin/buyvip/buy?userId=1&money=19.9&time=2025-05-20"



### 5. 用户模块

5.1 获取所有用户

接口地址：GET /admin/users

功能描述：获取所有用户列表。

请求参数：无

返回数据：用户列表，每条记录包含以下字段：

id (Integer): 用户ID。

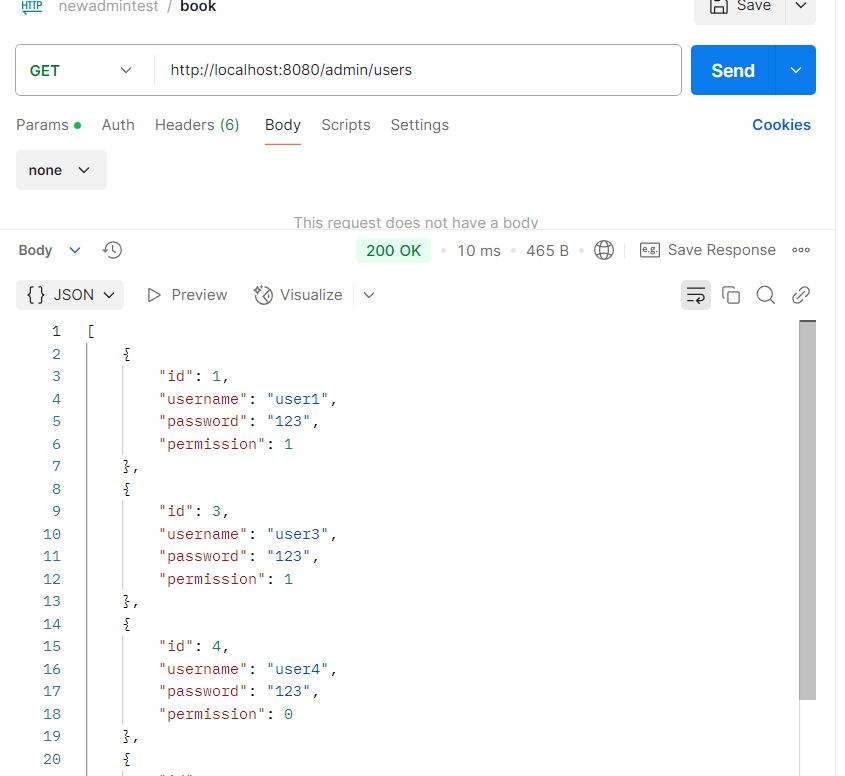
username (String): 用户名。

password (String): 密码。

permission (Integer): 权限（0=普通用户，1=会员）。

示例：

GET "http://localhost:8080/admin/users"



5.2 根据ID获取用户

接口地址：GET /admin/users/{id}

功能描述：根据用户ID获取用户详情。

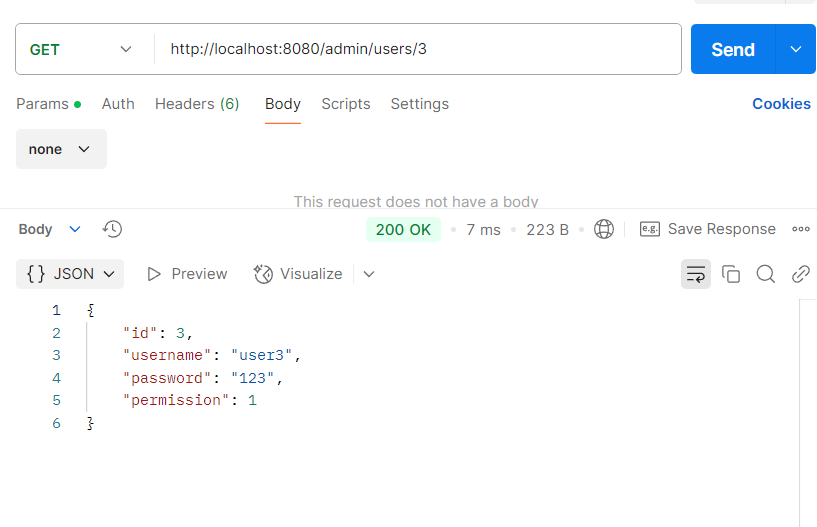
请求参数：

id (Integer): 用户ID。

返回数据：用户详情，字段同上。

示例：

GET "http://localhost:8080/admin/users/1"



5.3 创建用户

接口地址：POST /admin/users

功能描述：创建新用户。

请求参数：JSON 格式，包含以下字段：

username (String): 用户名。

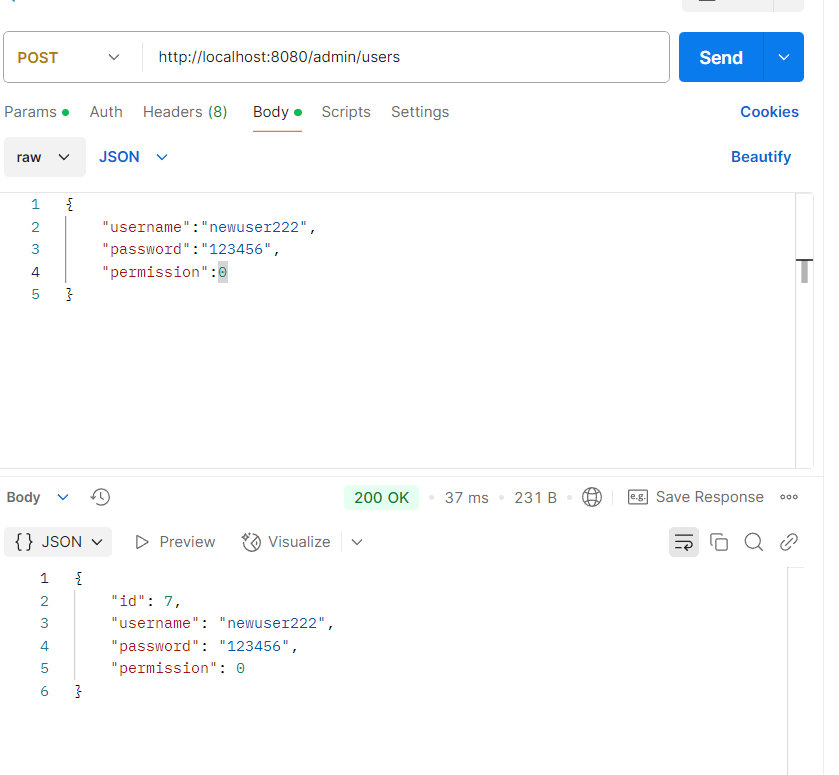
password (String): 密码。

permission (Integer): 权限。

返回数据：创建的用户对象。

示例：

POST "http://localhost:8080/admin/users"



5.4 更新用户信息

接口地址：PUT /admin/users/{id}

功能描述：更新用户信息。

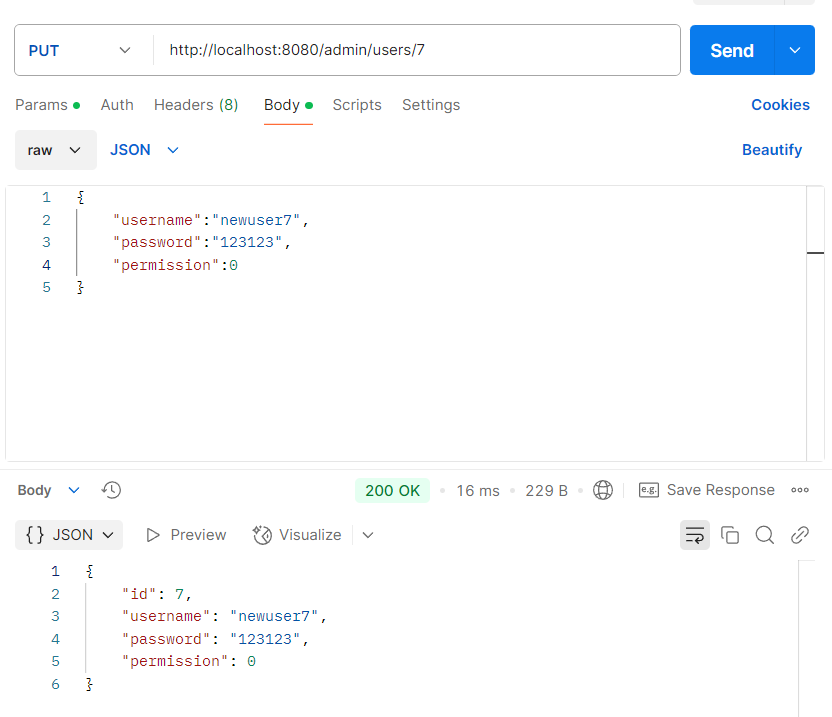
请求参数：

id (Integer): 用户ID。

JSON 格式，包含需要更新的字段。

返回数据：更新后的用户对象。

示例：

PUT "http://localhost:8080/admin/users/7"

5.5 删除用户

接口地址：DELETE /admin/users/{id}

功能描述：根据用户ID删除用户。

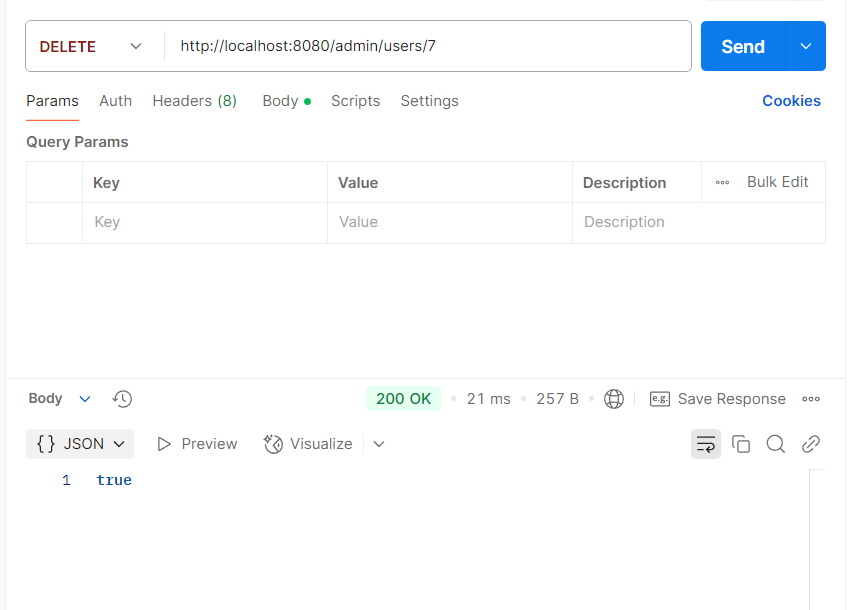
请求参数：

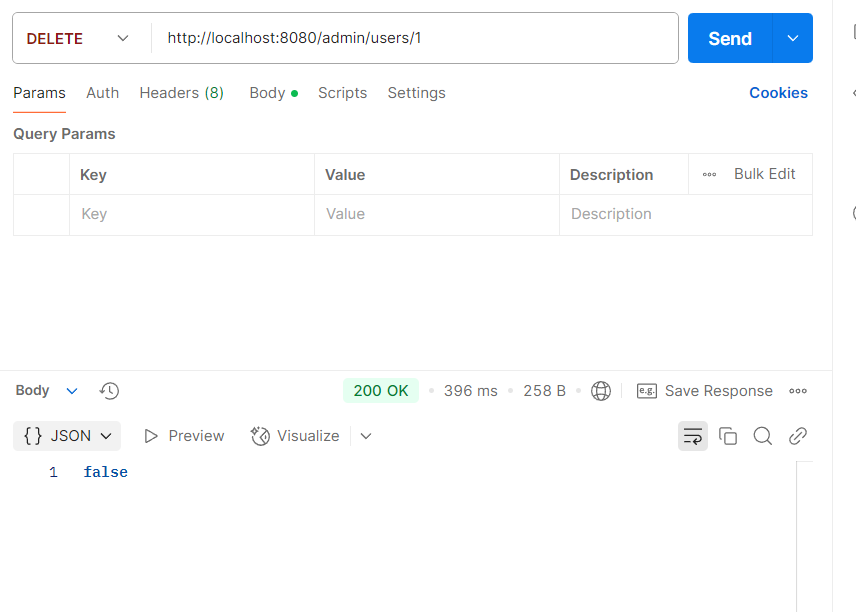
id (Integer): 用户ID。

返回数据：true 和 false

示例：

DELETE "http://localhost:8080/admin/users/7"





### 6. 阅读历史服务

6.1 记录阅读行为

接口地址：POST /reading/record

功能描述：记录用户的阅读行为。检查用户和书籍是否存在，若存在则创建新的阅读记录，并更新书籍的阅读次数。

请求参数：

userId (Integer): 用户ID，必填。

bookId (Integer): 书籍ID，必填。

返回值：

成功：返回字符串 true。

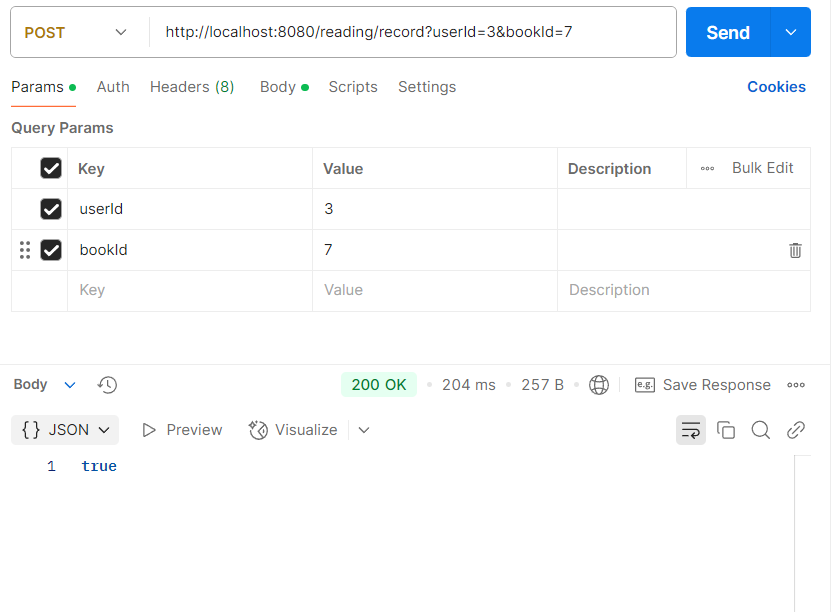
用户不存在：返回 false。

书籍不存在：返回false。

示例：

POST “http://localhost:8080/reading/record?userId=3&bookId=7”

返回示例：



6.2 查询个人阅读历史

接口地址：GET /reading/history/{userId}

功能描述：根据用户ID查询该用户的阅读历史记录。

路径参数：

userId (Integer): 用户ID，必填。

返回值：

成功：返回用户阅读历史记录的列表，包含书籍ID、阅读时间等信息。

用户不存在：返回空列表。

示例：

GET “http://localhost:8080/reading/history/1”

返回示例：

### 7. 数据分析服务

7.1 获取总用户数量

接口地址：GET /analyse/users

功能描述：返回系统中所有用户的总数。

请求参数：无

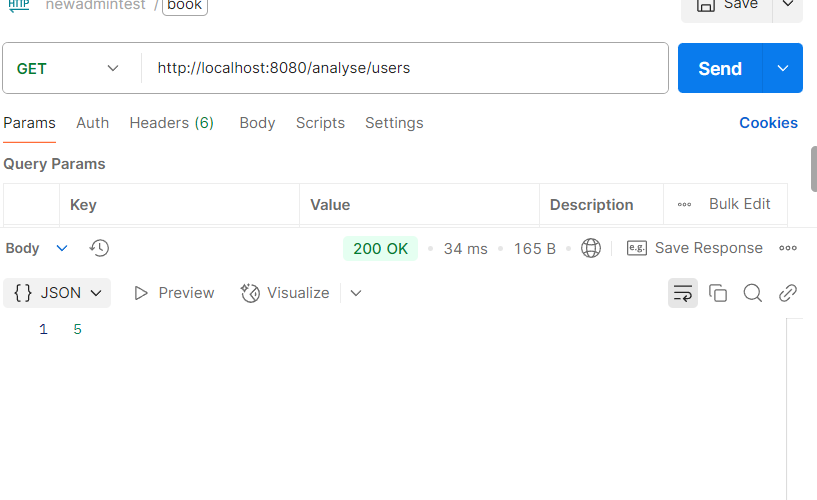
返回值：

成功：返回用户总数（long 类型）。

示例：

GET “http://localhost:8080/analyse/users”

返回示例：



7.2 获取VIP用户数量

接口地址：GET /analyse/vips

功能描述：返回系统中所有VIP用户的总数。

请求参数：无

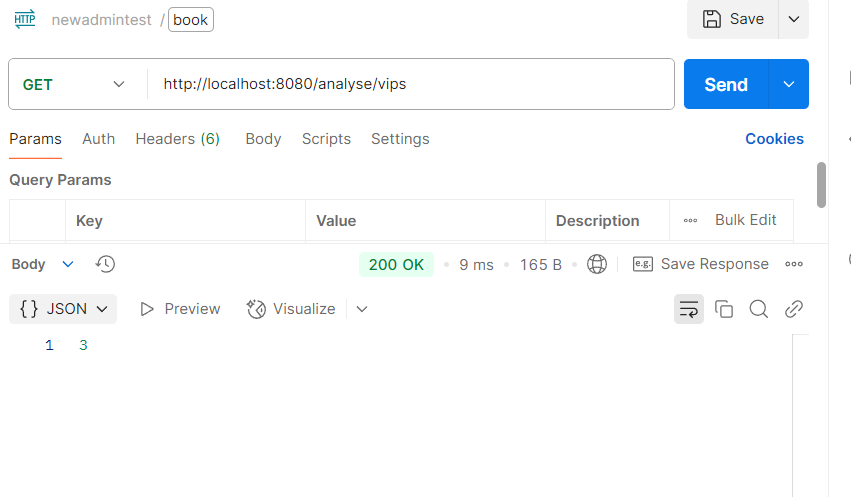
返回值：

成功：返回VIP用户总数（long 类型）。

示例：

GET”http://localhost:8080/analyse/vips”

返回示例：



7.3 获取书籍总数

接口地址：GET /analyse/books

功能描述：返回系统中所有书籍的总数。

请求参数：无

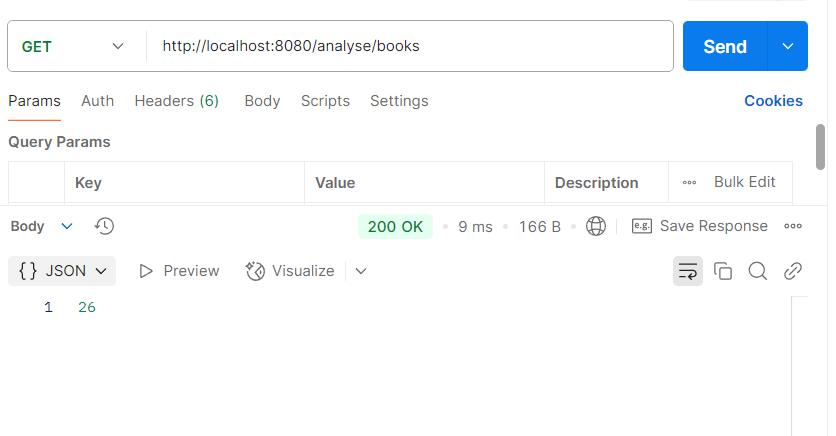
返回值：

成功：返回书籍总数（long 类型）。

示例：

GET “http://localhost:8080/analyse/books”

返回示例：

7.4 按类别统计书籍数量

接口地址：GET /analyse/sumNumByCate

功能描述：按书籍类别统计每个类别的阅读总数。

请求参数：无

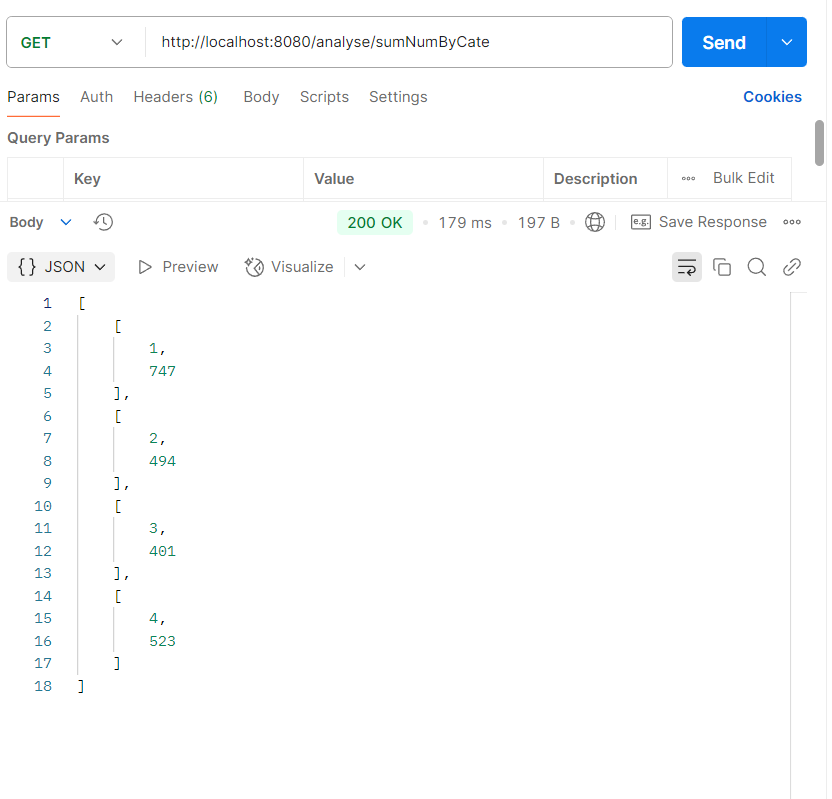
返回值：

成功：返回包含类别和对应数量的列表（List<Object[]>），每个对象包含两个字段：类别名称和数量。

示例：

GET “http://localhost:8080/analyse/sumNumByCate”

返回示例：

7.5 按类别统计消费金额

接口地址：GET /analyse/totalMoneyByCate

功能描述：按书籍类别统计每个类别的消费金额。

请求参数：无

返回值：

成功：返回包含类别和对应消费金额的列表（List<Object[]>），每个对象包含两个字段：类别名称和消费金额。

示例：

GET “http://localhost:8080/analyse/totalMoneyByCate”

返回示例：

