番茄商城体系结构设计文档

文档作者

主要编写者：孙琳嘉，邹涛, 杨怀宇，代辂

其他编写者：

文档修改历史

修改人员	日期	修改原因	版本号
孙琳嘉	2025/4/15	创建文档、6.1	1
孙琳嘉	2025/4/17	完成部分后端内容、1、 2 、5.3、5.4 、5.5、6.1	1.1
邹涛	2025/4/22	完成后端内容5.3，5.4，5.5，6.1	1.2
孙琳嘉	2025/4/23	完善后端部分，添加order部分	1.3
邹涛	2025/4/23	增加后端接口内容	1.4
孙琳嘉	2025/4/24	增加order部分内容	1.5
杨怀宇	2025/4/28	完成逻辑视角、组合视角	1.6
邹涛	2025/5/26	完善后端接口规范部分	1.7
邹涛	2025/5/27	修改并完善后端接口规范部分	1.8
孙琳嘉	2025/5/27	添加后端接口规范、业务需求表等	1.9

1. 引言

1.1 编制目的

本报告详细完成对 **番茄商城** 的概要设计，达到指导详细设计和开发的目的，同时实现和测试人员及用户的沟通。

本报告面向开发人员、测试人员及最终用户而编写，是了解系统的导航。

1.2 参考资料

1. [IEEE 1471-2000]
2. 《软件工程与计算（卷二）：软件开发的技术基础》刘钦、丁二玉著

2. 产品概述

2.1 前景和挑战

近年来，随着人们对知识的追求和文化消费的不断升级，实体书本市场展现出强劲的复苏与增长潜力。番茄读书的线上商城业务将围绕顾客、商家和管理员三大核心服务对象展开。通过精准的市场定位与优质的服务体验，番茄读书的线上商城有望成为实体书本销售领域的一颗璀璨明珠，为读者和出版界搭建起一座高效、便捷的文化交流桥梁。

2.2 业务需求

- **用户管理功能完善**：实现用户可以顺利登录、注册，能够查看和修改个人信息，确保用户信息的准确性和安全性，为用户提供个性化的购书体验。
- **商品信息管理高效**：用户能够方便地获取仓库中所有货物信息，以及指定货物的详细信息，便于用户快速找到心仪的书籍；管理员可以高效地更新、创建、删除产品信息，并能精准地调整指定产品的库存数额，确保商品信息的实时性和准确性。
- **交易流程顺畅**：用户可以顺利提交交易结算单，根据结算单中的物品自动生成支付单，并能自由修改支付单据的状态（支付、取消），确保交易流程的便捷性和灵活性。
- **广告展示优化**：用户能够完整地获取全部广告信息，通过优化广告展示，提升用户体验的同时，增加平台的商业价值。

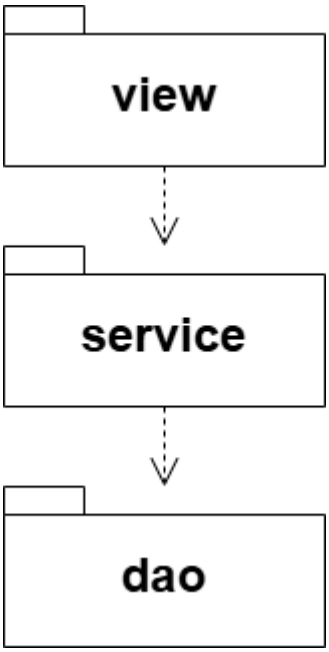
用户角色	功能模块	关键能力
顾客/管理员	用户模块	创建用户、获取和修改用户信息
管理员	商品模块	创建商品、管理商品信息、库存等
	广告模块	创建广告、管理广告等
	订单模块	查看所有订单、上传订单物流信息等
	优惠券模块	创建优惠卷、管理优惠卷
顾客	商品模块	查看商品信息、能够搜索商品、查看商品排行、分类等、筛选商品
	购物模块	添加商品至购物车
	订单模块	选择购物车商品、使用优惠券、填写地址下单、确认收货
	地址模块	创建地址
	收藏模块	收藏喜欢的书籍

	浏览记录	查看浏览记录
	评论模块	评论某书籍、回复他人评论、查看消息
	优惠券模块	领取、使用优惠券

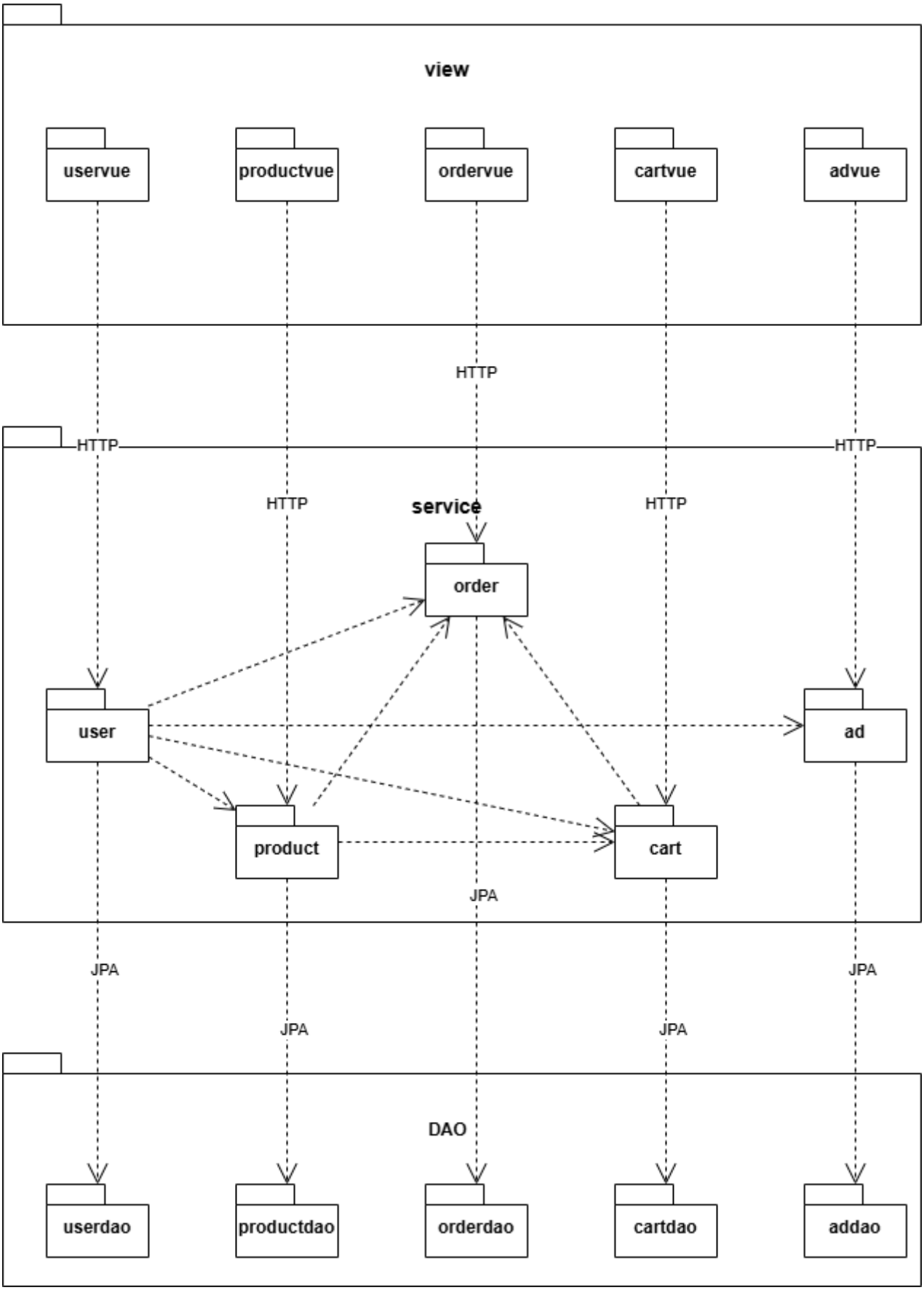
3. 逻辑视角

在番茄商城系统中，选择了**分层体系结构风格**，将系统分为4层（展示层、控制层（controller层）、业务逻辑层和数据层），这四层能够很好地示意整个高层抽象。其中展示层包含GUI页面的实现，控制层包括对展示层请求的分配，业务逻辑层包含系统具体业务逻辑处理的实现，数据层负责数据的持久化和访问。分层体系结构的逻辑视图和逻辑设计方案如下图所示。

3.1 参照体系结构风格的包图表达逻辑视角



3.2 软件体系结构逻辑设计方案



4. 组合视角

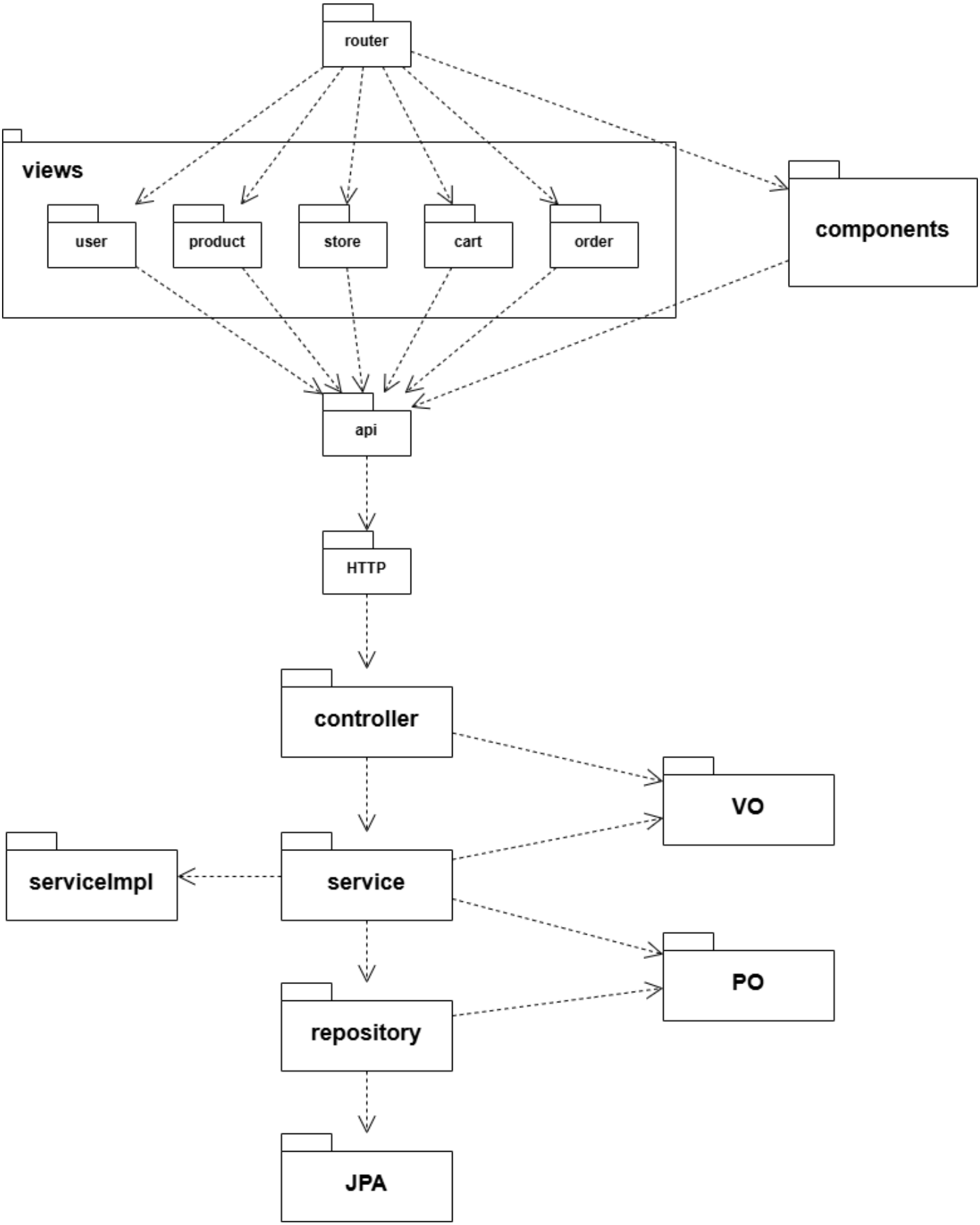
4.1 开发包图

表1 开发包设计

开发(物理)包	依赖的其他开发包
controller	service, vo
service	repository, vo
serviceImpl	repository, po, vo
repository	po
po	
vo	
enums	po, vo, service
util	controller,service,repository, vo, serviceImpl
configure	TomatoMallApplication, CorsFilter, LoginInterceptor, MyWebMvcConfig, SecurityConfig
SecurityConfig	
exception	controller, service, GlobalExceptionHandler, TomatoMallException
views	components, api
components	views
utils	api, views
assets	views, components
AdvertisementService	repository, po, vo
CartService	repository, po, vo
ImageService	repository, po, vo
OrderExpirationService	repository, po, vo
OrderService	repository, po, vo
ProductService	repository, po, vo
UserService	repository, po, vo
ads	views
pay	views
product	views
user	views
CorsFilter	configure
LoginInterceptor	configure
MyWebMvcConfig	configure
SecurityConfig	configure
AdvertisementRepository	po
CartItemRepository	po
CartRepository	po
OrderItemRepository	po
OrderRepository	po
ProductRepository	po
StockpileRepository	po

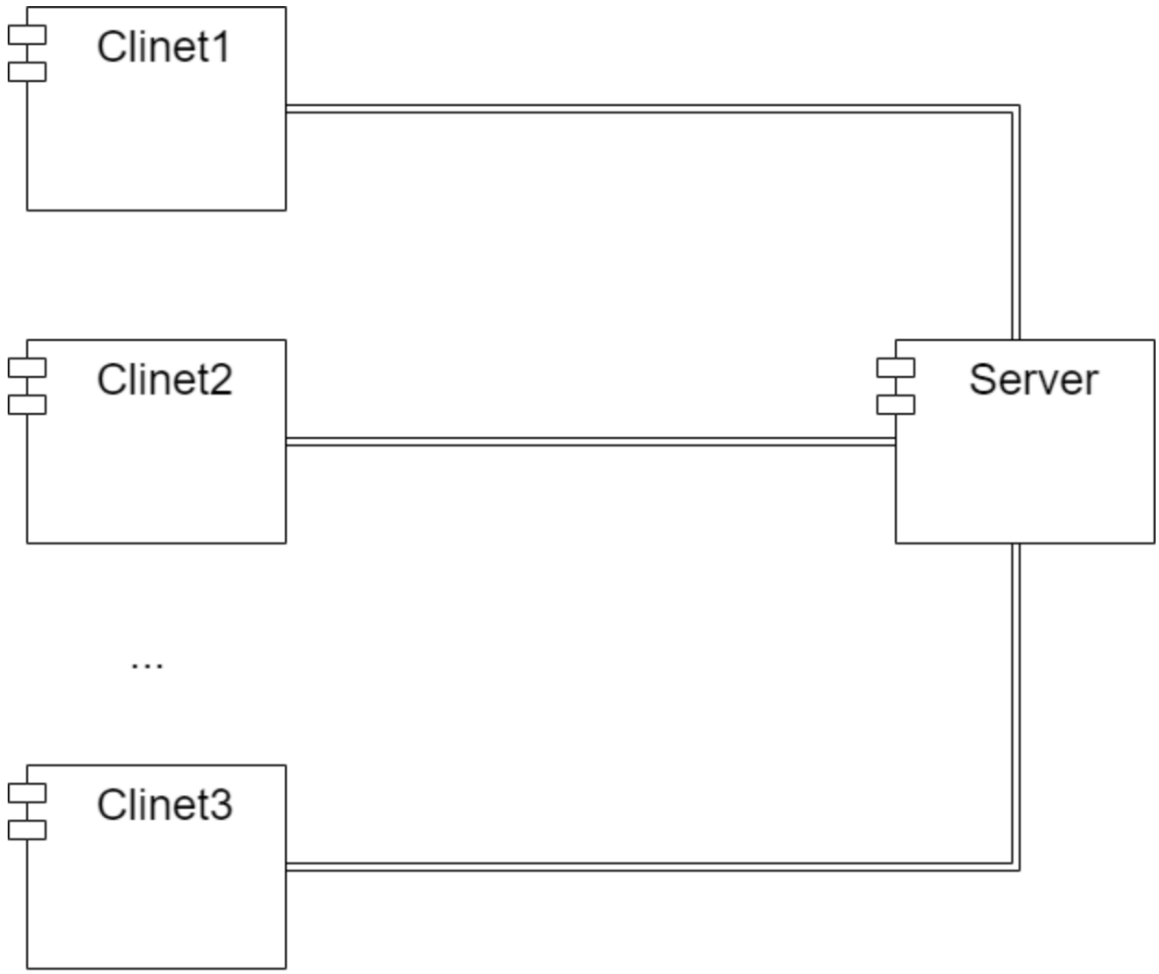
StockpileRepository	po
UserRepository	po
OssUtil	util
SecurityUtil	util
TokenUtil	util

开发包图



4.2 运行时进程

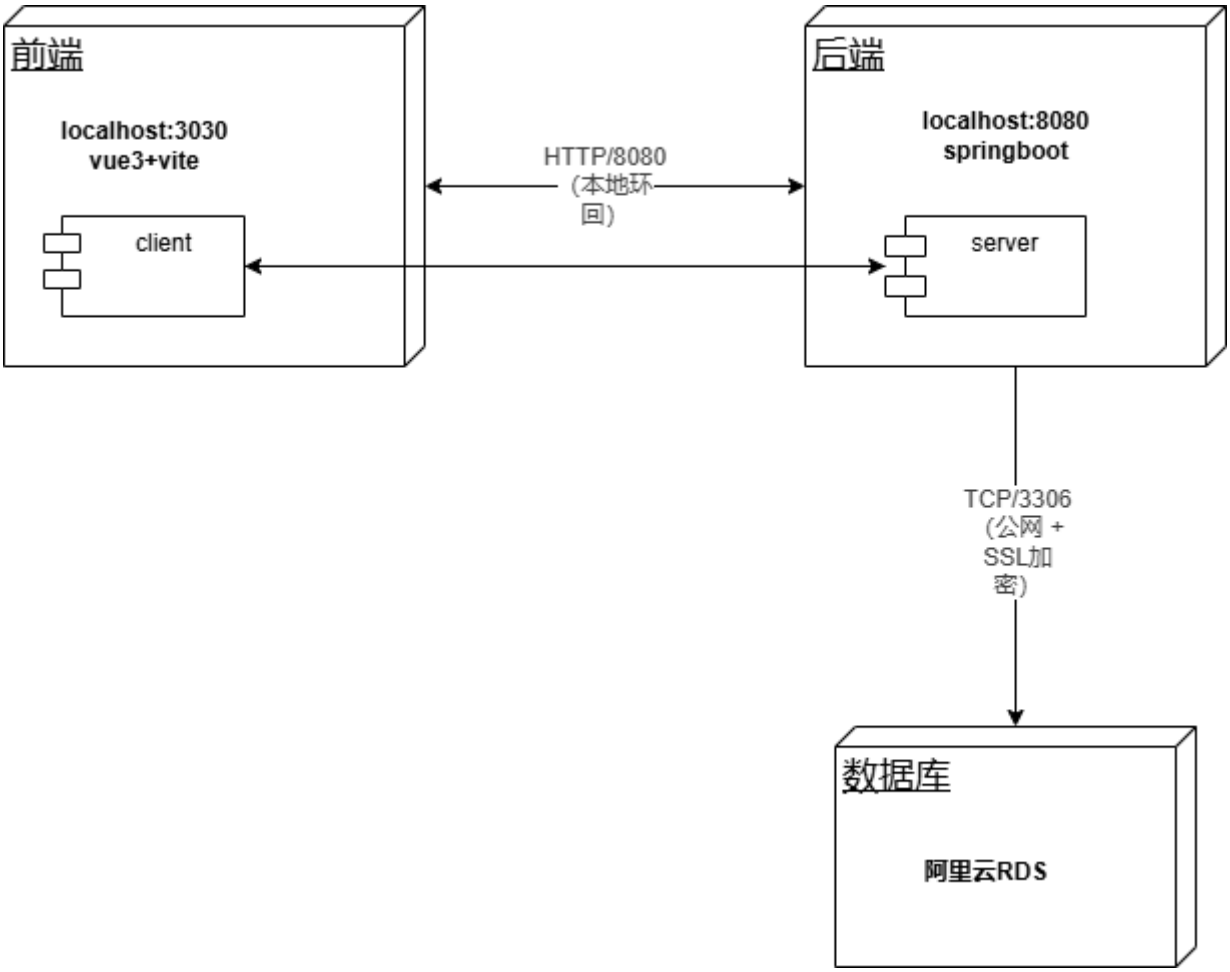
目前未将项目部署至服务器，前后端都在用户本地运行，数据库在云端



4.3 物理部署

项目采用混合部署架构：前端（vue3，localhost:3030）与后端（Spring Boot，localhost:8080）部署于本地开发机，通过局域网通信；阿里云RDS MySQL提供云端数据服务，后端通过公网SSL加密连接3306端口。本地防火墙开放3030/8080端口，云数据库安全组仅允许后端IP访问，实现内外网隔离，支撑开发测试环境运行。

部署图



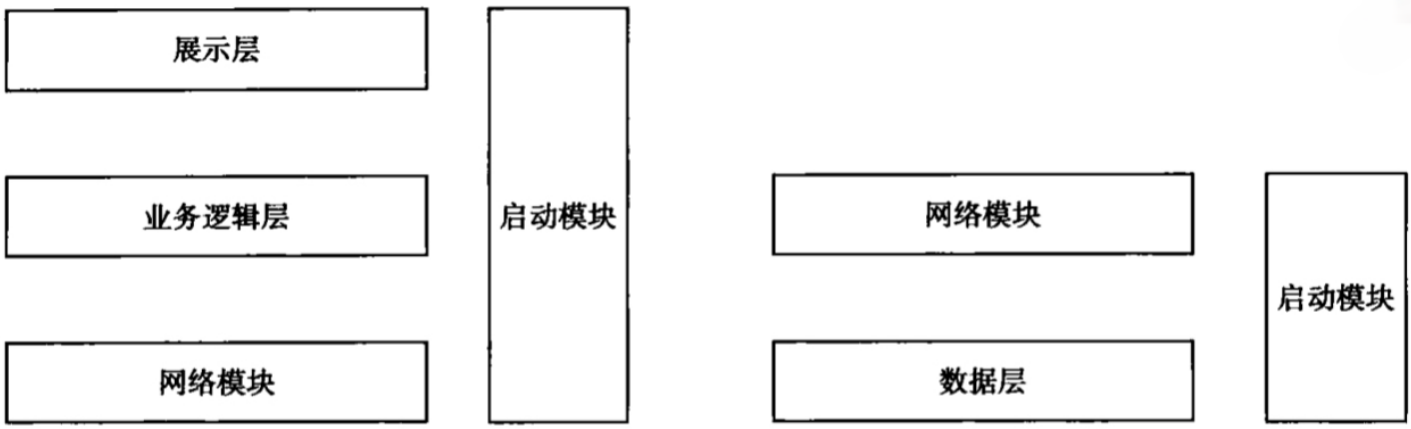
5. 接口视角

5.1 模块的职责

层	职责
启动模块	后端：负责初始化 Spring Boot 应用，配置必要的启动项（如数据库连接、服务器端口等）并 启动后端服务。 前端：负责初始化前端项目的基本配置（如 Vue、路由和状态管理），并启动用户界面。
展示层	负责渲染用户界面，显示不同页面和组件，如商品展示页、购物车页面、用户登录页面 等。
业务逻辑层	处理页面中的业务逻辑，管理状态，如计算购物车总价、表单验证、用户交互等，向后 端发送请求并处理返回数据。
数据层	负责与数据库进行交互，执行数据的持久化操作，提供数据访问接口供业务逻辑层使用，如增、删、改、查操作
网络模块	后端：接收和处理客户端请求，进行路由分发、数据格式转换、跨域支持、安全验证以及错误 处理，确保前后端的顺畅通信。

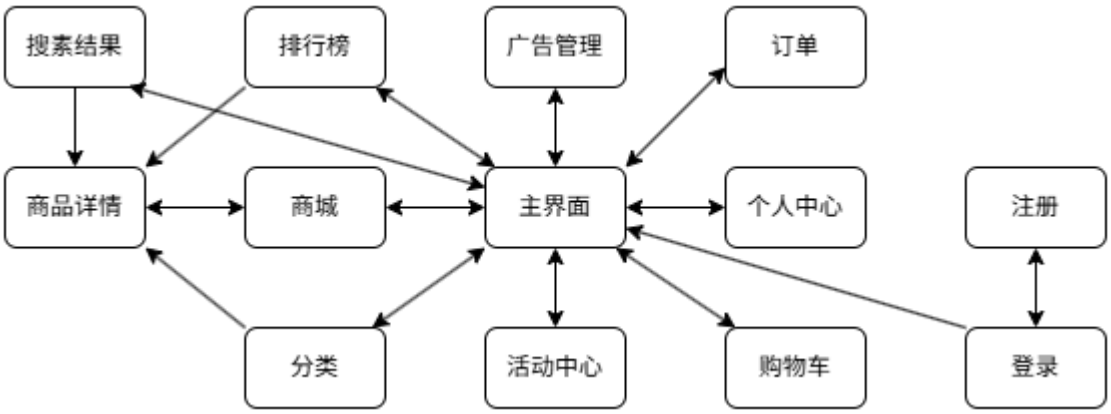
	前端：基于 Axios 向后端发送 HTTP 请求，处理请求参数、响应结果的格式化、错误处理等， 实现前后端的通信。

5.2 模块视图



5.3 用户界面

主要界面及其关系如下



5.4 接口层的分解

5.4.1 职责

模块	职责
account	负责用户界面
cart	负责购物车界面
product	负责商品界面
advertisement	负责广告界面

order	负责订单界面
address	负责地址界面
BrowseHistory	负责浏览记录界面
category	负责书籍分类界面
collection	负责收藏界面
comment	负责评论部分
coupon	负责优惠券界面
rank	负责商品排行
search	负责搜索部分
upload	负责图片上传部分

5.4.2 接口规范

5.4.2.1 /api/accounts接口规范

提供的服务（供接口）		
createUser	语法	public Response<String> createUser(@RequestBody AccountVO accountVO);
	前置条件	accountVO中的信息不能为空且与AccountVO匹配
	后置条件	返回注册成功
updateUser	语法	public AccountVO updateAccount(AccountVO account);
	前置条件	accountVO中的信息不能为空，与AccountVO匹配，已登录
	后置条件	返回是否更新成功
login	语法	public Response<String> login(@RequestBody AccountLoginRequest accountLoginRequest)
	前置条件	accountLoginRequest信息不为空且与AccountLoginRequest匹配
	后置条件	返回token
getUser	语法	public Response<AccountVO> getUser(@PathVariable String username,@RequestHeader("token") String token)

	前置条件	username不为空，且已登录
	后置条件	返回用户信息
getUserName	语法	public Response<AccountVO> getUserName(@PathVariable String id)
	前置条件	id不为空，且已登录
	后置条件	返回用户信息
需要的服务（需接口）		
服务名		服务
AccountService.create(AccountVO accountVO)		创建用户
AccountService.login(AccountLoginRequest accountLoginRequest);		用户登录
AccountService.updateInformation(AccountVO accountVO,String token);		修改用户信息
AccountService.getInformation(String username,String token);		获取用户信息
AccountService.getUsername(String id);		获取id对应的用户信息

5.4.2.2 /api/cart接口规范

提供的服务（供接口）		
addCartItem	语法	public Response<CartItemVO> addCartItem(@RequestBody CartItemRequest request)
	前置条件	request信息不为空且与CartItemRequest匹配
	后置条件	返回创建后的购物车商品信息
deleteCartItem	语法	public Response<String> deleteCartItem(@PathVariable String cartItemId)
	前置条件	cartItemId不能为空
	后置条件	返回是否删除成功

updateCartItemQuantity	语法	public Response<String> updateCartItemQuantity(@RequestBody UpdateCartQuantity request)
	前置条件	request信息不为空且与UpdateCartQuantity匹配
	后置条件	返回是否修改成功
getAllCartItemOfTheUser	语法	public Response<CartData> getAllCartItemOfTheUser()
	前置条件	无（已登录）
	后置条件	返回用户购物车所有商品、总商品数和总金额
checkout	语法	public Response<OrderVO> checkout(@RequestBody CheckoutRequest request)
	前置条件	request不为空，且与CheckoutRequest匹配
	后置条件	返回订单信息
需要的服务（需接口）		
服务名		服务
CartItemService.addBookToCart(String productId, Integer quantity);		添加商品到购物车
CartItemService.deleteCartItem(String cartItemId);		删除购物车商品
CartItemService.updateQuantity(String productId, Integer quantity);		修改购物车商品数量
CartItemService.getAll();		获取购物车内所有商品
CartItemService.checkout(List<String> cartItemIds,String paymentMethod);		选择商品付款，锁定库存，若30分钟未付款则取消锁定

5.4.2.3 /api/products接口规范

提供的服务(供接口)		
getAllBook	语法	public Response<List<ProductVO>> getAllBook()
	前置条件	无(已登录)

	后置条件	返回所有商品的信息(不包括商品库存)
getBookById	语法	public Response<ProductVO> getBookById(@PathVariable String id)
	前置条件	商品 id 不为空,且已登录
	后置条件	返回对应商品信息,如果不存在对应商品返回信息为空
updateBookInfo	语法	public Response<String> updateBookInfo(@RequestBody UpdateBookRequest bookRequest)
	前置条件	bookReuquest中的id字段不能为空,其他字段任意
	后置条件	返回是否更新成功
addBook	语法	public Response<ProductVO> addBook(@RequestBody AddBookRequest bookRequest)
	前置条件	bookRequest中的title,price,rate字段不能为空,其他字段任意
	后置条件	返回创建商品的基本信息
deleteBook	语法	public Response<String> deleteBook(@PathVariable String id)
	前置条件	商品的id不能为空,且已登录
	后置条件	返回是否删除成功
updateBookStock	语法	public Response<String> updateBookStock(@PathVariable String productId, @RequestBody Map<String, Integer> requestBody)
	前置条件	进行库存调整的商品id不能为空,请求体中的amout字段不能为空
	后置条件	返回商品的库存是否更改成功
getStockInfoById	语法	public Response<StockpileVO> getStockInfoById(@PathVariable String productId)
	前置条件	进行库存查询的商品id不能为空
	后置条件	返回对应商品的库存信息
getSalesByID	语法	public Response<Integer> getSalesByID(@PathVariable String productId)

	前置条件	对应商品存在
	后置条件	返回对应商品的销量
deleteCategory	语法	public Response<String> deleteCategory(@RequestParam("productId") String productId)
	前置条件	1.对应商品存在 2.管理员进行请求
	后置条件	返回商品分类删除是否成功
updateCategory	语法	public Response<String> updateCategory(@RequestParam("productId") String productId, @RequestParam("categoryId") String categoryId)
	前置条件	对应商品以及分类存在
	后置条件	返回商品分类更新是否成功
getProductCategory	语法	public Response<String> getProductCategory(@RequestParam("productId") String productId)
	前置条件	对应商品存在
	后置条件	得到对应商品的分类信息

需要的服务(需接口)

服务名	服务
productService.getAllBook()	得到所有商品信息
productService.getBookById(id)	查找特定id商品
productService.updateBookInfo(bookRequest)	更新对应商品信息
productService.addBook(bookRequest)	添加商品
productService.deleteBook(id)	删除指定商品
	更新指定商品库存

productService.updateBookStock(productId, amount)	
productService.getStockInfoById(productId)	查找指定商品库存
productService.getSalesById(Integer.*parseInt*(productId))	得到商品销量
productService.deleteCategory(Integer.*parseInt*(productId))	删除对应商品分类信息
productService.updateCategory(Integer.*parseInt*(productId) ,Integer.*parseInt*(categoryId))	更新商品的分类信息
productService.getProductCategory(Integer.*parseInt*(productId))	得到商品所属的分类

5.4.2.4 /api/advertisements接口规范

提供的服务(供接口)		
getAllAdvertisements	语法	public Response<List<AdvertisementVO>> getAllAdvertisements()
	前置条件	无(已登录)
	后置条件	返回所有的广告信息
updateAdvertisement	语法	public Response<String> updateAdvertisement(@RequestBody AdvertisementVO advertisementVO)
	前置条件	advertisementVO中的id以及productId字段非空,其他字段任意
	后置条件	返回是否成功更新的信息
createAdvertisement	语法	public Response<AdvertisementVO> createAdvertisement(@RequestBody AdvertisementVO advertisementVO)
	前置条件	title,content,imgUrl,productId四个字段皆为必填,其他字段任意
	后置条件	返回创建的商品信息
deleteAdvertisement	语法	public Response<String> deleteAdvertisement(@PathVariable String id)
	前置条件	要删除的广告id且已登录

	后置条件	返回是否成功删除的信息
需要的服务(需接口)		
服务名		服务
advertisementsService.getAllAdvertisements()		得到所有广告信息
advertisementsService.updateAdvertisement(advertisementVO)		更新指定id广告的信息
advertisementsService.createAdvertisement(advertisementVO)		创建新的广告
advertisementsService.deleteAdvertisement(id)		删除指定id的广告

5.4.2.5 /api/orders接口规范

提供的服务（供接口）		
getPending	语法	public Response<List<OrderVO>> getPending();
	前置条件	
	后置条件	返回当前用户未支付的订单
getSuccess	语法	public Response<List<OrderVO>> getSuccess();
	前置条件	
	后置条件	返回当前用户已支付成功的订单
getAll	语法	public Response<List<OrderVO>> getAll()
	前置条件	
	后置条件	返回用户所有订单
pay	语法	public ResponseEntity<Map<String, Object>> pay(@PathVariable String orderId)
	前置条件	订单有效
	后置条件	返回支付宝表单
handleAlipayNotify	语法	

		public void handleAlipayNotify(HttpServletRequest request, HttpServletResponse response)
	前置条件	用户在支付宝支付成功
	后置条件	返回success
returnUrl	语法	public String returnUrl(HttpServletRequest request, HttpServletResponse response)
	前置条件	用户在支付宝支付成功
	后置条件	
getStatus	语法	public Response<String> getStatus(@PathVariable String orderId)
	前置条件	
	后置条件	返回订单状态
getDetail	语法	public Response<CartData> getDetail(@PathVariable String orderId)
	前置条件	
	后置条件	返回订单详情
需要的服务（需接口）		
服务名		服务
AlipayClient.pageExecute		调用支付宝SDK生成表单
OrderService.getPending()		获取用户未支付订单
OrderService.getSuccess()		获取用户支付成功的订单
OrderService.getAll()		获取用户全部订单

5.4.2.6 /api/address接口规范

提供的服务(供接口)		
getAddressById	语法	public Response<List<String>> getAddressById(@PathVariable String id)
	前置条件	1.用户的id不为空,且用户存在

	后置条件	返回属于该id的所有收货地址
addAddress	语法	public Response<String> addAddress(@RequestBody AddAddressRequest addAddressRequest)
	前置条件	addAddressRequest中的accountId和address不能为空,其他字段任意
	后置条件	返回是否新增地址成功
getDefaultAddress	语法	public Response<AddressVO> getDefaultAddress(@RequestParam("userId") String userId)
	前置条件	1.用户存在
	后置条件	返回用户的默认地址
deleteAddress	语法	public Response<String> deleteAddress(@RequestParam("userId") String userId, @RequestParam("addressId") String addressId)
	前置条件	1.用户存在 2.地址存在
	后置条件	删除用户对应的收货地址,返回是否删除成功
updateAddress	语法	public Response<String> updateAddress(@RequestParam("addressId") String addressId, @RequestParam("address") String address, @RequestParam("telephone") String telephone, @RequestParam("name") String name)
	前置条件	1.地址存在,其他字段任意
	后置条件	更新对应收货地址的内容,返回是否更新成功
setDefaultAddress	语法	public Response<String> setDefaultAddress(@RequestParam("userId") String userId, @RequestParam("addressId") String addressId)
	前置条件	1.用户存在 2.收货地址存在

	后置条件	返回默认地址是否设置成功
需要的服务(需接口)		
服务名		服务
addressService.getAddressById(Integar id)		查找id所属的所有收货地址
addressService.addAddress(Integar id,String address)		为ID为id的用户新增收货地址
addressService.getDefaultAddress(Integer.*parseInt*(userId))		得到用户所有的默认地址
addressService.setDefaultAddress(Integer.*parseInt*(userId), Integer.*parseInt*(addressId))		将对应的收货地址设为默认地址
addressService.updateAddress(Integer.*parseInt*(addressId), address, telephone, name)		更新对应收货地址信息
addressService.deleteAddress(Integer.*parseInt*(userId), Integer.*parseInt*(addressId))		删除对应的收货地址

5.4.2.7 /api/upload接口规范

提供的服务(供接口)		
uploadStoreImage	语法	public Response<String> uploadStoreImage(MultipartFile image)
	前置条件	image不能为空
	后置条件	返回能够访问storeImage的url
uploadUserImage	语法	public Response<String> uploadUserImage(MultipartFile image)
	前置条件	image不能为空
	后置条件	返回能够访问UserImage的url
uploadBookImage	语法	public Response<String> uploadBookImage(MultipartFile image)
	前置条件	Image不能为空

	后置条件	返回能够访问BookImage的url
需要的服务(需接口)		
服务名		服务
uploadService.uploadStoreImage(image)		上传商店图片到阿里云
uploadService.uploadUserImage(image)		上传用户头像到阿里云
uploadService.uploadBookImage(image)		上传商品图片到阿里云

5.4.2.8 /api/category接口规范

提供的服务(供接口)		
getAllCategory	语法	public Response<List<CategoryData>> getAllCategory()
	前置条件	无(已登录)
	后置条件	返回分类规则里面的所有分类
getProductsByCategoryId	语法	public Response<List<ProductVO>> getProductsByCategoryId(@PathVariable String category_id)
	前置条件	1.商品分类存在
	后置条件	返回对应分类里面的所有商品
需要的服务(需接口)		
服务名		服务
categoryService.getAllCategory()		给出所有的分类
categoryService.getProductsByCategoryId(category_id)		根据分类id获取商品

5.4.2.9 /api/browser接口规范

提供的服务(供接口)		
	语法	public Response<String> createBrowserHistory(@RequestParam("userId") String userId,

createBrowserHistory		@RequestParam("productId") String productId)
	前置条件	1.用户存在 2.商品存在
	后置条件	返回浏览历史是否创建成功
getBrowserHistory	语法	public Response<List<BrowserHistoryData>> getBrowserHistory(@RequestParam("userId") String userId)
	前置条件	1.用户存在
	后置条件	返回用户所有的浏览历史
deleteBrowserHistory	语法	public Response<String> deleteBrowserHistory(@RequestParam("userId") String userId, @RequestParam("productId") String productId)
	前置条件	1.用户存在 2.商品存在
	后置条件	返回浏览历史是否删除成功
需要的服务(需接口)		
服务名		服务
browserHistoryService.createBrowserHistory(Integer.*parseInt*(userId),Integer.*parseInt*(productId))		创建浏览历史
browserHistoryService.getBrowserHistory(Integer.*parseInt*(userId))		得到用户浏览历史
browserHistoryService.deleteBrowserHistory(Integer.*parseInt*(userId),Integer.*parseInt*(productId))		删除对应浏览历史

5.4.2.10 /api/collection接口规范

提供的服务(供接口)		
createCollection	语法	public Response<String> createCollection(@RequestParam("userId") String userId,

		@RequestParam("productId") String productId)
	前置条件	1.用户存在 2.商品存在
	后置条件	返回创建收藏是否成功
getAllCollection	语法	public Response<List<ProductVO>> getAllCollection(@RequestParam("userId") String userId)
	前置条件	1.用户存在
	后置条件	返回用户所有收藏商品
deleteCollection	语法	public Response<String> deleteCollection(@RequestParam("userId") String userId, @RequestParam("productId") String productId)
	前置条件	1.用户存在 2.商品存在
	后置条件	返回收藏商品是否删除成功
需要的服务(需接口)		
服务名		服务
collectionService.createCollection(Integer.*parseInt*(userId) ,Integer.*parseInt*(productId))		创建对应的收藏商品
collectionService.getAllCollection(Integer.*parseInt*(userId))		得到用户所有的收藏商品
collectionService.deleteCollection(Integer.*parseInt*(userId), Integer.*parseInt*(productId))		删除用户对应的收藏商品

5.4.2.11 /api/coupon接口规范

提供的服务(供接口)		
	语法	

createCoupon		public Response<String> createCoupon(@RequestBody CreateCouponRequest couponRequest)
	前置条件	1.请求体中各字段不为空 2.请求者为管理员
	后置条件	返回优惠券创建是否成功
deleteCoupon	语法	public Response<String> deleteCoupon(@RequestParam("couponId") String couponId)
	前置条件	1.优惠券存在 2.请求者为管理员
	后置条件	返回优惠券删除是否成功
需要的服务(需接口)		
服务名		服务
couponService.createCoupon(newCoupon)		管理员创建优惠券
couponService.deleteCoupon(Integer.*parseInt*(couponId))		管理员删除优惠券

5.4.2.12 /api/ranking接口规范

提供的服务(供接口)		
rankingBySales	语法	public Response<List<RankingData>> rankingBySales()
	前置条件	无
	后置条件	返回销量排行榜相关数据
rankingByRate	语法	public Response<List<ProductVO>> rankingByRate()
	前置条件	无
	后置条件	返回评分排行榜商品列表
需要的服务(需接口)		
服务名		服务

rankingService.rankingBySales()	得到销量排行榜数据
rankingService.rankingByRate()	得到评分排行榜数据

5.4.2.13 /api/search接口规范

提供的服务(供接口)		
search	语法	public Response<List<ProductVO>> search(@RequestParam("keyword") String keyword)
	前置条件	1.keyword不为空
	后置条件	返回与keyword相关得商品列表
searchWithFilter	语法	public Response<List<ProductVO>> searchWithFilter(@RequestParam("keyword") String keyword, @RequestParam(value = "maxPrice",required = false) BigDecimal maxPrice, @RequestParam(value = "minPrice",required = false) BigDecimal minPrice, @RequestParam(value = "maxRate",required = false) Double maxRate, @RequestParam(value = "minRate",required = false) Double minRate)
	前置条件	1.keyword不为空,其他任意
	后置条件	返回符合条件得商品列表
需要的服务(需接口)		
服务名		服务
searchService.search(keyword)		搜索相关得商品
searchService.searchWithFilter(keyword, maxPrice, minPrice, maxRate, minRate)		对搜索结果进行对应条件得过滤

5.4.2.14 /api/usercoupon接口规范

提供的服务(供接口)		
getAllCoupon	语法	public Response<CouponData> getAllCoupon(@PathVariable String user_id)
	前置条件	1.用户存在 2.由用户本人发起
	后置条件	返回用户自己得优惠券信息
getCouponCanUse	语法	public Response<CouponData> getCouponCanUse(@RequestParam("userId") String userId, @RequestParam("price") String price)
	前置条件	1.用户必须存在 2.price>=0
	后置条件	返回符合条件得优惠券信息
getCoupon	语法	public Response<String> getCoupon(@RequestParam("userId") String userId, @RequestParam("couponId") String couponId)
	前置条件	1.用户存在 2.优惠券存在
	后置条件	返回用户是否成功领取优惠券信息
需要的服务(需接口)		
服务名		服务
userCouponService.getAllCoupon(Integer.*parseInt*(user_id))		得到用户当前拥有以及可领取优惠券
userCouponService.getCouponCanUse(Integer.*parseInt*(userId), new BigDecimal(price))		筛选用户可以使用得优惠券
userCouponService.getCoupon(Integer.*parseInt*(userId), Integer.*parseInt*(couponId))		用户领取优惠券

5.4.2.15 /api/comments接口规范

提供的服务（供接口）		
createComment	语法	public Response<String> createComment(@RequestBody CommentVO commentVO)
	前置条件	无
	后置条件	返回create success
deleteComment	语法	public Response<String> deleteComment(@PathVariable String commentId)
	前置条件	无
	后置条件	返回delete success
getComment	语法	public Response<CommentVO> getComment(@PathVariable String commentId)
	前置条件	commentId不为空
	后置条件	返回评论信息
getCommentOfProduct	语法	public Response<List<CommentVO>> getCommentOfProduct(@PathVariable String productId)
	前置条件	无
	后置条件	返回商品的所有评论
getCommentOfUser	语法	public Response<List<CommentVO>> getCommentOfUser(@PathVariable String userId)
	前置条件	无
	后置条件	返回用户得所有评论
getReply	语法	public Response<List<CommentVO>> getReply(@PathVariable String commentId)
	前置条件	无
	后置条件	返回评论的所有回复
需要的服务（需接口）		
服务名		服务

CommentService.createComment(Integer userId ,Integer productId, String comment, Integer repliedCommentId)	创建评论
CommentService.deleteComment(Integer commentId)	删除评论
CommentService.getComment(Integer commentId);	获取评论
CommentService.getCommentOfProduct(Integer productId);	获取商品评论
CommentService.getCommentOfUser(Integer userId);	获取用户的所有评论
CommentService.getReplyComment(Integer commentId);	获取某评论的所有回复

5.5 业务逻辑层的分解

业务逻辑层模块主要包含多个根据控制层请求转发来调用的模块，并返回控制层请求的需要返回给展示层的数据的对象，例如，对象负责和账户有关的业务逻辑，比如增加、查询、修改账户等。

5.5.1 职责

模块	职责
AccountService	负责和账号有关的业务逻辑
AdvertisementService	负责和广告有关的业务逻辑
CartItemService	负责和购物车与支付有关的业务逻辑
ProductService	负责和商品管理有关的业务逻辑
UploadService	负责上传图片业务逻辑
AddressService	负责用户收货地址的管理
CategoryService	负责商品分类的管理
OrderService	负责查看订单的逻辑
BrowserHistoryService	负责用户浏览历史的逻辑
CollectionService	负责用户收藏商品的逻辑

CouponService	负责管理员管理优惠券逻辑
RankingService	负责商品排行榜逻辑
SearchService	负责商品搜索逻辑
UserCouponService	负责用户优惠券逻辑
CommentService	负责评论逻辑

5.5.2 接口规范

5.5.2.1 AccountService接口规范

提供的服务（供接口）		
create	语法	public String create(AccountVO accountVO)
	前置条件	accountVO中的信息不为空且与AccountVO匹配
	后置条件	返回创建成功
updateInformation	语法	public Boolean updateInformation(AccountVO accountVO,String token)
	前置条件	accountVO和token中的信息不为空，accountVO合规；已登录，token合规
	后置条件	返回是否更新成功
login	语法	public String login(AccountLoginRequest accountLoginRequest)
	前置条件	accountLoginRequest信息不为空且与AccountLoginRequest匹配
	后置条件	返回token
getInformation	语法	public AccountVO getInformation(String username,String token)
	前置条件	username不为空，且已登录
	后置条件	返回用户信息
getUsername	语法	public AccountVO getUsername(String id)
	前置条件	id不为空

	后置条件	返回用户信息
需要的服务（需接口）		
服务名		服务
AccountRepository.findByUsername(String username);		通过用户名找到账号

5.5.2.2 CartItemService接口规范

提供的服务（供接口）		
addBookToCart	语法	public CartItemVO addBookToCart(String productId, Integer quantity)
	前置条件	无
	后置条件	返回创建后的购物车商品信息
deleteCartItem	语法	public String deleteCartItem(String cartItemId)
	前置条件	无
	后置条件	返回是否删除成功
updateQuantity	语法	public String updateQuantity(String cartItemId,Integer quantity)
	前置条件	无
	后置条件	返回修改成功或数量不足
getAll	语法	public CartData getAll()
	前置条件	无
	后置条件	返回用户购物车所有商品、总商品数和总金额
checkout	语法	public OrderVO checkout(List<String> cartItemIds,String paymentMethod,Integer addressId ,Integer couponId);
	前置条件	无
	后置条件	返回订单信息
需要的服务（需接口）		

服务名	服务
CartItemRepository.findByUserIdAndOrderIsNull(Integer userId)	通过用户id找到购物车商品列表
CartItemRepository.findByCartItemId(Integer cartItemId)	通过购物车商品id找到商品
CartItemRepository.findByProductIdAndUserId(Integer productId, Integer userId)	通过对应用户和产品id找到购物车商品
CartItemRepository.setOrderForCartItems(@Param("order") Order order, @Param("cartIds") List<Integer> cartIds)	为购物车商品设置对应订单
CartItemRepository.findCartItemsToClear(@Param("now") LocalDateTime now)	找到订单已失效的购物车商品
CartItemRepository.findByOrder_OrderId(Integer orderId)	通过对应订单的订单编号找到对应订单
CartItemRepository.clearOrdersForCartItems(@Param("cartItems") List<CartItem> cartItems)	为对应订单已失效的购物车商品清除对应订单
ProductRepository.findById(Integer productId)	找到对应商品编号的商品
StockpileRepository.findByProductId(Integer productId)	找到商品编号对应商品的库存
StockpileRepository.updateStockpile(@Param("id") Integer id, @Param("amount") int amount, @Param("frozen") int frozen);	更新商品库存
OrderRepository.findByOrderId(Integer orderId)	找到订单编号对应订单
OrderRepository.findTimeoutOrder(LocalDateTime now)	找到超时未支付的订单

5.5.2.3 ProductService接口规范

提供的服务(供接口)		
getAllBook	语法	public List<ProductVO> getAllBook()
	前置条件	无
	后置条件	返回所有商品的信息(不包括商品库存)

getBookById	语法	public ProductVO getBookById(String id)
	前置条件	商品 id 不为空,且已登录
	后置条件	返回对应商品信息,如果不存在对应商品返回信息为空
updateBookInfo	语法	public String updateBookInfo(UpdateBookRequest bookRequest)
	前置条件	bookReuquest中的id字段不能为空,其他字段任意
	后置条件	返回是否更新成功
addBook	语法	public ProductVO addBook(AddBookRequest bookRequest)
	前置条件	bookRequest中的title,price,rate字段不能为空,其他字段任意
	后置条件	返回创建商品的基本信息
deleteBook	语法	public String deleteBook(String id)
	前置条件	商品的id不能为空,且已登录
	后置条件	返回是否删除成功
updateBookStock	语法	public String updateBookStock(String productId,Integer amount)
	前置条件	进行库存调整的商品id不能为空,请求体中的amout字段不能为空
	后置条件	返回商品的库存是否更改成功
getStockInfoById	语法	public StockpileVO getStockInfoById(String productId)
	前置条件	进行库存查询的商品id不能为空
	后置条件	返回对应商品的库存信息
getSalesByID	语法	public Integer getSalesById(Integer productId)
	前置条件	对应商品存在
	后置条件	返回对应商品的销量
deleteCategory	语法	public String deleteCategory(Integer productId)
	前置条件	1.对应商品存在 2.管理员进行请求
	后置条件	返回商品分类删除是否成功

updateCategory	语法	public String updateCategory(Integer productId, Integer categoryId)
	前置条件	对应商品以及分类存在
	后置条件	返回商品分类更新是否成功
getProductCategory	语法	public String getProductCategory(Integer productId)
	前置条件	对应商品存在
	后置条件	得到对应商品的分类信息
需要的服务(需接口)		
ProductRepository.findById(Integer productId);		通过商品id查找对应的商品
stockpileRepository.findByProductId(Integer productId);		通过商品id查找对应商品的库存
stockpileRepository.deleteByProductId(Integer productId);		通过商品id删除对应的商品库存信息
specificationsRepository.deleteByProductId(Integar productId);		通过商品id删除对应商品的规格说明信息
specificationsRepository.findByProductId(Integar productId)		通过商品id查找对应商品的规格说明信息
categoryRepository.findById(Integar productId);		通过商品id查找商品对应的分类

5.5.2.4 AdvertisementService接口规范

提供的服务(供接口)		
getAllAdvertisements	语法	public List<AdvertisementVO> getAllAdvertisements()
	前置条件	无
	后置条件	返回所有广告的信息
updateAdvertisement	语法	public String updateAdvertisement(AdvertisementVO advertisementVO)

	前置条件	无
	后置条件	返回广告信息是否更新成功
createAdvertisement	语法	public AdvertisementVO createAdvertisement(AdvertisementVO advertisementVO)
	前置条件	无
	后置条件	返回创建的广告信息
deleteAdvertisement	语法	public String deleteAdvertisement(String id)
	前置条件	无
	后置条件	返回对应id的广告信息是否删除
需要的服务(需接口)		
服务名		服务
advertisementsRepository.findAll();		查找所有广告信息
advertisementsRepository.findById(Integer id);		根据id查找对应的广告
advertisementsRepository.deleteById(Integer id);		根据id删除对应的广告
productRepository.findById(Integer id);		根据id查找对应的商品信息

5.5.2.5 OrderService接口规范

提供的服务（供接口）		
getPending	语法	public List<OrderVO> getPending()
	前置条件	
	后置条件	返回用户所有未支付订单
getSuccess	语法	public List<OrderVO> getSuccess()
	前置条件	
	后置条件	返回用户所有已支付订单
getAll	语法	public List<OrderVO> getAll()

	前置条件	
	后置条件	返回用户所有订单
getStatus	语法	public String getStatus(String orderId)
	前置条件	
	后置条件	返回订单状态
getDetail	语法	public CartData getDetail(String orderId)
	前置条件	
	后置条件	返回订单详情
需要的服务（需接口）		
服务名		服务
OrderRepository.findByUserIdAndStatus		通过用户id和订单状态获取订单
OrderRepository.findByUserId		通过用户id获取订单

5.5.2.6 UploadService接口规范

提供的服务（供接口）		
uploadStoreImage	语法	public String uploadStoreImage(MultipartFile image)
	前置条件	image不能为空
	后置条件	返回能够访问storeImage的url
uploadUserImage	语法	public String uploadUserImage(MultipartFile image)
	前置条件	image不能为空
	后置条件	返回能够访问UserImage的url
uploadBookImage	语法	public String uploadBookImage(MultipartFile image)
	前置条件	image不能为空
	后置条件	返回能够访问BookImage的url
需要的服务（需接口）		
服务名		服务

OssUtil.upload(String objectName, byte[] content, String mimeType)	上传图片到阿里云

5.5.2.7 CategoryService接口规范

提供的服务(供接口)		
getAllCategory	语法	public List<CategoryData> getAllCategory()
	前置条件	无(已登录)
	后置条件	返回分类规则里面的所有分类
getProductsByCategoryId	语法	public List<ProductVO> getProductsByCategoryId(String category_id)
	前置条件	1.商品分类id存在
	后置条件	返回对应分类里面的所有商品
需要的服务(需接口)		
服务名		服务
ProductRepository.findByCategoryId(Integer category_id)		查找对应分类所有商品
CategoryRepository.findAll()		查找当前所有分类

5.5.2.8 AddressService接口规范

提供的服务(供接口)		
getAddressById	语法	public List<AddressVO> getAddressById(Integer userId)
	前置条件	1.用户的id不为空,且用户存在
	后置条件	返回属于该id的所有收货地址
addAddress	语法	public String addAddress(Integer userId,String address,String telephone,String name)
	前置条件	addAddressRequest中的accountId和address不能为空,其他字段任意
	后置条件	返回是否新增地址成功

getDefaultAddress	语法	public AddressVO getDefaultAddress(Integer userId)
	前置条件	1.用户存在
	后置条件	返回用户的默认地址
deleteAddress	语法	public String deleteAddress(Integer userId,Integer addressId)
	前置条件	1.用户存在 2.地址存在
	后置条件	删除用户对应的收货地址,返回是否删除成功
updateAddress	语法	public String updateAddress(Integer addressId,String address,String telephone,String name)
	前置条件	1.地址存在,其他字段任意
	后置条件	更新对应收货地址的内容,返回是否更新成功
setDefaultAddress	语法	public String setDefaultAddress(Integer userId, Integer addressId)
	前置条件	1.用户存在 2.收货地址存在
	后置条件	返回默认地址是否设置成功
需要的服务(需接口)		
服务名		服务
AddressRepository.findByAccountId(Integer id)		查找对应的收货地址
AccountRepository.findById(Integer account_id)		查找对应id的用户
AddressRepository.findByAccountIdAndDeletedFalse(userId)		查找用户当前有效的收货地址

提供的服务（供接口）		
getAddressById	语法	public List<String> getAddressById(String id)
	前置条件	id不能为空

	后置条件	返回属于id的收货地址
addAddress	语法	public String addAddress(String account_id, String address)
	前置条件	account_id和address不能为空
	后置条件	返回添加收货地址是否成功
需要的服务（需接口）		
服务名		服务
addressRepository.findByAccountId(Integer id)		查找对应的收货地址
accountRepository.findById(Integer account_id)		查找对应id的用户

5.5.2.9 BrowserHistoryService接口规范

提供的服务(供接口)		
createBrowserHistory	语法	public String createBrowserHistory(Integer userId, Integer productId)
	前置条件	1.用户存在 2.商品存在
	后置条件	返回浏览历史是否创建成功
getBrowserHistory	语法	public List<BrowserHistoryData> getBrowserHistory(Integer userId)
	前置条件	1.用户存在
	后置条件	返回用户所有的浏览历史
deleteBrowserHistory	语法	public String deleteBrowserHistory(Integer userId, Integer productId)
	前置条件	1.用户存在 2.商品存在
	后置条件	返回浏览历史是否删除成功
需要的服务(需接口)		

服务名	服务
BrowserHistoryRepository.findByUserIdAndProductId(Integar userId,Integar productId)	根据用户和商品id找到对应浏览历史记录
BrowserHistoryRepository.findTop30ByUserIdAndDeletedFalseOrderByViewedAtDesc(Integar userId)	找到用户最近30条未删除的;浏览记录,以创建的时间降序输出

5.5.2.10 CollectionService接口规范

提供的服务(供接口)		
createCollection	语法	public String createCollection(Integer userId, Integer productId)
	前置条件	1.用户存在 2.商品存在
	后置条件	返回创建收藏是否成功
getAllCollection	语法	public List<ProductVO> getAllCollection(Integer userId)
	前置条件	1.用户存在
	后置条件	返回用户所有收藏商品
deleteCollection	语法	public String deleteCollection(Integer userId, Integer productId)
	前置条件	1.用户存在 2.商品存在
	后置条件	返回收藏商品是否删除成功
需要的服务(需接口)		
服务名		服务
CollectionRepository.findByUserIdAndProductId(Integar userId, Integar productId)		查找对应用户和商品的收藏信息
CollectionRepository.findByUserId(Integar userId)		查找用户所有的收藏商品
CollectionRepository.deleteById(Integar id)		删除对应的收藏信息

5.5.2.11 CouponService接口规范

提供的服务(供接口)		
createCoupon	语法	public String createCoupon(CouponVO couponVO)
	前置条件	1.请求体中各字段不为空
	后置条件	返回优惠券创建是否成功
deleteCoupon	语法	public String deleteCoupon(Integer couponId)
	前置条件	1.优惠券id存在
	后置条件	返回优惠券删除是否成功
需要的服务(需接口)		
服务名		服务
CouponRepository.findById(Integar couponId)		查找对应id的优惠券

5.5.2.12 RankingService接口规范

提供的服务(供接口)		
rankingBySales	语法	public List<RankingData> rankingBySales()
	前置条件	无
	后置条件	返回销量排行榜相关数据
rankingByRate	语法	public List<ProductVO> rankingByRate()
	前置条件	无
	后置条件	返回评分排行榜商品列表
需要的服务(需接口)		
服务名		服务
ProductRepository.findAll()		查找所有商品数据
StockpileRepository.findAll()		查找所有库存数据

5.5.2.13 SearchService接口规范

提供的服务(供接口)		
search	语法	public List<ProductVO> search(String keyword)
	前置条件	1.keyword不为空
	后置条件	返回与keyword相关得商品列表
searchWithFilter	语法	public List<ProductVO> searchWithFilter(String keyword, BigDecimal maxPrice, BigDecimal minPrice, Double maxRate, Double minRate)
	前置条件	1.keyword不为空,其他任意
	后置条件	返回符合条件得商品列表
需要的服务(需接口)		
服务名		服务
ProductRepository.findByTitleContainingIgnoreCase(String keyword);		查找title包含keyword的商品
ProductRepository.findByDescriptionContainingIgnoreCase(String keyword);		查找description包含keyword的商品
SpecificationsRepository.findByValueContainingIgnoreCase(String keyword);		查找specification包含keyword的商品

5.5.2.14 UserCouponService接口规范

提供的服务(供接口)		
getAllCoupon	语法	public CouponData getAllCoupon(Integer user_id)
	前置条件	1.用户id存在
	后置条件	返回用户自己得优惠券信息
getCouponCanUse	语法	public String getCoupon(Integer userId, Integer couponId)
	前置条件	1.用户id存在 2.price>=0
	后置条件	返回符合条件得优惠券信息

getCoupon	语法	public CouponData getCouponCanUse(Integer userId, BigDecimal price)
	前置条件	1.用户id存在 2.优惠券id存在
	后置条件	返回用户是否成功领取优惠券信息
需要的服务(需接口)		
服务名		服务
UserCouponRepository.findByUserId (Integar userId)		查找用户所有优惠券
CouponRepository.findByEndTimeAfterAndRemainingAmountGreaterThanAndDeletedFalse(LocalDateTime now , Integar remaing_amount)		查找当前可领取的所有优惠券

5.5.2.15 CommentService接口规范

提供的服务（供接口）		
createComment	语法	public String createComment(Integer userId ,Integer productId, String comment, Integer repliedCommentId)
	前置条件	userId、productId不为空，repliedCommentId不为空，若不是任何评论的回复则传入0
	后置条件	返回创建成功
deleteComment	语法	public String deleteComment(Integer commentId)
	前置条件	commentId不为空
	后置条件	返回是否更新成功
getComment	语法	public CommentVO getComment(Integer commentId)
	前置条件	无
	后置条件	返回评论信息
	语法	public List<CommentVO> getCommentOfProduct(Integer productId)

getCommentOfProduct	前置条件	无
	后置条件	返回商品的所有评论
getCommentOfUser	语法	public List<CommentVO> getCommentOfUser(Integer userId)
	前置条件	无
	后置条件	返回用户评论
getReplyComment	语法	public List<CommentVO> getReplyComment(Integer commentId)
	前置条件	无
	后置条件	返回评论的所有回复
需要的服务（需接口）		
服务名		服务
CommentRepository.findByUserId(Integer userId)		通过用户id找到所有评论
CommentRepository.findByProductId(Integer productId)		通过商品id找到所有评论
CommentRepository.findByRepliedCommentId(Integer commentId)		通过评论id找到所有回复

5.6 数据层的分解

5.6.1 职责

数据层模块的职责如表所示

模块	职责
AccountRepository	负责和用户账号有关的和数据库的操作
AdvertisementsRepository	负责和广告有关的数据库的操作
CartItemRepository	负责和购物车商品有关的数据库的操作
OrderRepository	负责和订单有关的数据库的操作
ProductRepository	负责商品管理有关的数据库的操作

SpecificationsRepository	负责商品规格说明有关数据库的操作
StockpileRepository	负责商品库存信息有关数据库的操作
AddressRepository	负责用户收货地址有关数据库的操作
CategoryRepository	负责商品分类有关数据库的操作
BrowserHistoryRepository	负责浏览历史有关数据库操作
CollectionRepository	负责收藏商品有关数据库操作
CouponRepository	负责优惠券有关数据库操作
UserCouponRepository	负责用户优惠券有关数据库操作
CommentRepository	负责评论有关的数据库操作

5.6.2 接口规范

5.6.2.1 AccountRepository接口规范

提供的服务（供接口）		
findByUsername	语法	public Account findByUsername(String username);
	前置条件	无
	后置条件	返回username对应的用户
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.2 CartItemRepository接口规范

提供的服务（供接口）		
findByUserIdAndOrderIsNull	语法	public List<CartItem> findByUserId(Integer userId)
	前置条件	无
	后置条件	返回用户id对应购物车商品列表
	语法	

findByCartItemId		Optional<CartItem> findByCartItemId(Integer cartItemId);
	前置条件	无
	后置条件	返回id对应的购物车商品
findByProductIdAndUserId	语法	Optional<CartItem> findByProductIdAndUserId(Integer productId, Integer userId);
	前置条件	无
	后置条件	返回商品id和用户id对应的商品
setOrderForCartItems	语法	void setOrderForCartItems(@Param("order") Order order, @Param("cartIds") List<Integer> cartIds);
	前置条件	单据状态不能为空
	后置条件	为该订单中的购物车商品设置对应订单
findCartItemsToClear	语法	List<CartItem> findCartItemsToClear(@Param("now") LocalDateTime now);
	前置条件	无
	后置条件	返回订单已过期的购物车商品
findByOrder_OrderId	语法	List<CartItem> findByOrder_OrderId(Integer orderId);
	前置条件	无
	后置条件	返回订单id对应订单的商品列表
clearOrdersForCartItems	语法	void clearOrdersForCartItems(@Param("cartItems") List<CartItem> cartItems);
	前置条件	无
	后置条件	为失效订单中的购物车商品清除对应订单
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.3 OrderRepository接口规范

提供的服务（供接口）		
findByOrderId	语法	public Order findByOrderId(Integer orderId);
	前置条件	无
	后置条件	返回订单id对应的订单
findByUserIdAndStatus	语法	public List<Order> findByUserIdAndStatus(Integer userId, String status);
	前置条件	无
	后置条件	返回该用户该状态订单列表
findByUserId	语法	public List<Order> findByUserId(Integer userId);
	前置条件	无
	后置条件	返回该用户订单列表
findTimeoutOrder	语法	public List<Order> findTimeoutOrder(@Param("now") LocalDateTime now);
	前置条件	无
	后置条件	返回超时订单
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.4 AddressRepository接口规范

提供的服务（供接口）		
findByAccountIdAndDeletedFalse	语法	List<Address> findByAccountIdAndDeletedFalse(Integer accountId);
	前置条件	accountId不为空
	后置条件	返回该用户的所有未删除收货地址
	语法	Boolean existsByAccountId(Integer accountId);

existsByAccountId	前置条件	1.accountId不为空
	后置条件	返回是否存在有记录包含accountId
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.5 CategoryRepository接口规范

需要的服务（需接口）	
服务名	服务
JpaRepository	框架提供的直接操作数据库的服务

5.6.2.6 ProductRepository接口规范

提供的服务（供接口）		
findByTitleContainingIgnoreCase	语法	List<Product> findByTitleContainingIgnoreCase(String keyword);
	前置条件	1.keyword不能为空
	后置条件	返回title属性包含keyword的商品
findByDescriptionContainingIgnoreCase	语法	List<Product> findByDescriptionContainingIgnoreCase(String keyword);
	前置条件	1.keyword不能为空
	后置条件	返回description属性包含keyword的商品
findByCategoryId	语法	List<Product> findByCategoryId(Integer categoryId);
	前置条件	1.catrgoryId存在
	后置条件	返回对应分类的所有商品

需要的服务（需接口）	
服务名	服务
JpaRepository	框架提供的直接操作数据库的服务

5.6.2.7 SpecificationRepository接口规范

提供的服务（供接口）		
findByProductId	语法	List<Specification> findByProductId(Integer productId);
	前置条件	productId不为空
	后置条件	返回对应商品id的所有规格说明
deleteByProductId	语法	void deleteByProductId(Integer productId);
	前置条件	productId不为空
	后置条件	数据库中对应商品的规格说明被删除
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.8 StockpileRepository接口规范

提供的服务（供接口）		
findByProductId	语法	Stockpile findByProductId(Integer productId);
	前置条件	productId不能为空
	后置条件	返回对应商品的库存信息
deleteByProductId	语法	void deleteByProductId(Integer productId);
	前置条件	productId不能为空
	后置条件	数据库中对应商品的库存信息被删除
updateStockpile	语法	void updateStockpile(@Param("id") Integer id, @Param("amount") int amount, @Param("frozen") int frozen);
	前置条件	参数不能为空

	后置条件	更新对应商品的库存信息
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.9 AdvertisementRepository接口规范

提供的服务(供接口)		
deleteByProductId	语法	void deleteByProductId(Integer productId);
	前置条件	productId不为空
	后置条件	对应商品广告被删除
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.10 BrowserHistoryRepository接口规范

提供的服务（供接口）		
findTop30ByUserIdAndDeletedFalseOrderByViewedAtDesc	语法	List<BrowserHistory> findTop30ByUserIdAndDeletedFalseOrderByViewedAtDesc(Integer userId);
	前置条件	1.userId存在
	后置条件	返回用户最近30条浏览记录
findByUserIdAndProductId	语法	BrowserHistory findByUserIdAndProductId(Integer userId,Integer productId);
	前置条件	1.userId存在 2.productId存在

	后置条件	根据用户id和商品id返回对应的浏览记录
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.11 CollectionRepository接口规范

提供的服务（供接口）		
findByUserId	语法	List<Collection> findByUserId(Integer userId);
	前置条件	1.userId存在
	后置条件	返回用户所有的收藏记录
findByUserIdAndProductId	语法	Collection findByUserIdAndProductId(Integer userId,Integer productId);
	前置条件	1.uesrId存在 2.productId存在
	后置条件	返回用户id和商品id对应的收藏记录
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.12 CouponRepository接口规范

提供的服务（供接口）		
findByEndTimeAfterAndRemainingAmountGreaterThanAndDeletedFalse	语法	List<Coupon> findByEndTimeAfterAndRemainingAmountGreaterThanAndDeletedFalse(LocalDateTime now, Integer remainingAmount);
	前置条件	1.now不能为空 2.remaingAmount>=0
	后置条件	返回符合条件的所有优惠券信息

需要的服务（需接口）	
服务名	服务
JpaRepository	框架提供的直接操作数据库的服务

5.6.2.13 UserCouponRepository接口规范

提供的服务（供接口）		
findByCouponId	语法	UserCoupon findByCouponId(Integer couponId);
	前置条件	1.couponId存在
	后置条件	返回优惠券对应的用户优惠券
findByUserId	语法	List<UserCoupon> findByUserId(Integer user_id);
	前置条件	1.user_id存在
	后置条件	返回用户当前拥有所有优惠券
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

5.6.2.14 CommentRepository接口规范

提供的服务（供接口）		
findByUserId	语法	List<Comment> findByUserId(Integer userId);
	前置条件	无
	后置条件	返回用户对应的评论
findByProductId	语法	List<Comment> findByProductId(Integer productId);
	前置条件	无
	后置条件	返回商品对应的评论

findByRepliedCommentId	语法	List<Comment> findByRepliedCommentId(Integer commentId);
	前置条件	无
	后置条件	返回评论对应的回复
需要的服务（需接口）		
服务名		服务
JpaRepository		框架提供的直接操作数据库的服务

6. 信息视角

6.1 描述数据持久化对象(PO)

系统的PO类是对应的相关的实体类，在此做简单的介绍。

- Acount类包含用户的id、用户名、密码、姓名、头像、角色类型、电话、邮箱、地址属性
- Advertisement类包含id，名称，内容
- CartItem类包含id、对应用户、对应商品、数量、对应订单属性
- Order类包含id、对应用户、总金额、支付方式、状态、创建时间，所用优惠券、实际金额、物流信息、是否确认收货属性
- Product类包含id，名称，价格，评分，描述，封面，细节，图片，规格说明，分类
- Specification类包含id，规格项名称，规格项的值，规格对应的商品
- Stockpile类包含id，库存数，购买数，库存对应的商品
- Address类包含id，地址信息，地址所属的用户
- Category类包含id，分类名称，子分类，所属父分类
- Comment类包含id、评论内容、创建时间、发表用户、对应商品、回复的评论属性

示例：持久化对象AccountPO的定义如下

代码块

```
public class Account {

    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Id
    @Column(name = "id")
    private Integer id;

    @Basic
```

```java
    @Column(name="username")
    private String username; //用户名

    @Basic
    @Column(name="password")
    private String password; //password

    @Basic
    @Column(name="name")
    private String name;  //用户姓名

    @Basic
    @Column(name="avatar")
    private String avatar;  //用户头像链接

    @Basic
    @Column(name="role")
    private String role; //用户身份

    @Basic
    @Column(name="telephone")
    private String telephone;//用户手机号

    @Basic
    @Column(name="email")
    private String email;//用户邮箱

    @Basic
    @Column(name="location")
    private String location;//用户所在地

    public AccountVO toVO(){
        AccountVO account=new AccountVO();
        account.setId(this.id);
        account.setUsername(this.username);
        account.setName(this.name);
        account.setAvatar(this.avatar);
        account.setRole(this.role);
        account.setTelephone(this.telephone);
        account.setEmail(this.email);
        account.setLocation(this.location);
        return account;
    }

}
```

6.2 数据库表

数据库中包含Acount表、Advertisement表、CartItem表、Order表、Product表、Specification表、Stockpile表，Address表，Category表、Comment表。