基于 SpringBoot+Vue 的智慧农业助销平台的设计与实现*

刘民辉 林宁**

南宁学院信息工程学院，南宁，530200

摘 要 随着互联网发展与乡村振兴战略实施，农产品电商已成为连接城乡、促进农民增收的重要途径。传统销售模式存在渠道单一、信息不对称、物流成本高等问题，制约农业现代化发展。本文设计并实现一套适配小农户经营特点、整合交易撮合与服务支撑的智慧农业助销平台，采用 Spring Boot+Vue.js 前后端分离架构，结合 MySQL 数据库完成数据持久化，集成支付宝沙箱支付、WebSocket 实时通信等技术，实现用户权限管理、商品发布、在线交易、订单跟踪、在线客服、数据统计与智慧农业资讯等核心功能。系统测试表明，平台功能完整、运行稳定、性能达标，可有效解决农产品“卖难”问题，助力农业数字化转型与乡村产业振兴。

关键字 智慧农业；电商平台；Spring Boot；Vue.js；农产品销售

Design and Implementation of an Intelligent Agricultural Assistance Sales Platform Based on SpringBoot and Vue

Liu Minhui Lin Ning**
College of Information Engineering
Nanning University
Nanning 530200 China

Abstract—With the development of the Internet and the implementation of the rural revitalization strategy, agricultural product e-commerce has become an important way to connect urban and rural areas and increase farmers' income. However, traditional sales models have problems such as single channels, information asymmetry, and high logistics costs, which restrict the modernization of agriculture. This paper designs and implements an intelligent agricultural assistance sales platform suitable for small-scale farmers, integrating transaction matching and service support. The platform adopts the Spring Boot+Vue.js front-end and back-end separation architecture, uses MySQL database for data persistence, and integrates Alipay sandbox payment, WebSocket real-time communication and other technologies. It realizes core functions such as user authority management, product release, online transaction, order tracking, online customer service, data statistics and intelligent agricultural information. System tests show that the platform has complete functions, stable operation and qualified performance, which can effectively solve the problem of "difficult sales" of agricultural products and help the digital transformation of agriculture and the revitalization of rural industries.

Keywords—Smart Agriculture; E-commerce Platform; Spring Boot; Vue.js; Agricultural Product Sales

1 引 言

在国家乡村振兴、数字中国建设的时代背景下，智慧农业成为破解“三农”问题、推动农业提质增效最重要路径。我国的大部分农户是小规模的农户，销售渠道传统、市场信息滞后、流通链条长、产品损耗率高；消费者则对农产品安全、溯源与品质等方面有较高要求。传统电商无法满足小农户经营的需求，急需轻量化、高可用、全流程服务的数字化助销平台来帮助他们实现减免和降低风险[1]。

本文基于 Spring Boot 和 Vue.js 开发智慧农业助销平台，前端、后台分离，用户、商家和管理员三类角色。实现对于商品发布、交易支付、订单管理、客服沟通、数据统计、农业资讯等全流程的支持，以轻量化开发，高并发适配，安全可靠为目标，解决产销信息不对称、流通成本高、交易不透明等痛点问题，为农产品上行、乡村振兴提供技术支撑[2]。同时借鉴可持续乡村服务理念，优化平台服务体系[3]。

2 相关技术与工具介绍

2.1 Spring Boot 框架介绍

*基金资助：南宁学院2025年横向课题（2025HX100）资助。
**通讯作者：林宁，教授，linning@unn.edu.cn

Spring Boot 是基于 Spring 生态体系而搭建的轻量、快速开发框架。其主要目标在于简化传统 Spring 项目繁琐的配置流程，降低开发门槛，提高项目搭建与部署效率；自动配置后端环境可随时调整适合项目依赖需求，无需手工编写大量 XML 配置文件；设计丰富的 Starter 场景启动器方便一键集成各类第三方组件，极大地节约了依赖管理内容；支持 Tomcat、Jetty 等 Web 服务器模式进行打包运营，无需额外添加外部容器。本平台采用 Spring Boot 3.4.10 作为核心后端开发框架，在此基础上引入 Spring Security 安全框架实现细粒度的用户权限控制和接口安全校验，结合 JWT 技术实现无状态身份认证和会话管理，并与 MyBatis Plus 持久层框架相结合，完成高效的数据访问和数据库操作，为后端业务逻辑更稳定的运行提供保障。Spring Boot 具有轻量、易部署以及开发效率高等优点，被广泛应用于各种行业信息化系统中[4]。

2.2 Vue.js 前端框架介绍

Vue.js 是一款轻量级、易上手的渐进式前端开发框架，具有组件化开发、响应式数据绑定、路由动态管理、虚拟 DOM 渲染等优点，能够快速搭建交互友好的单页应用（SPA）。通过设计组件可以将页面功能进行解耦和复用，同时实现了响应式机制的自动追踪并及时更新页面视图，使得用户在操作过程中体验更佳，提升了前端开发效率。本平台的前端主要采用了 Vue 3 为核心框架，并且与 Element UI 企业级前端组件库相结合，可迅速搭建出界面美观，布局规范，交流顺畅的页面；此外通过 Axios 异步请求工具实现前后端数据稳定的交互方式，完成接口调用、参数传递、数据渲染等工作，给用户带来流畅、便捷的操作感受。因为 Vue 框架兼容性强、组件化好，往往需要与 Spring Boot 配合才能搭建前后端分离系统，满足各种 Web 管理平台、服务系统的开发需求[5]。

2.3 MySQL 与 MyBatis Plus 介绍

MySQL 是开源关系型数据库，支持高并发、易扩展，适合存储用户、商品、订单等结构化数据。MyBatis Plus 是 MyBatis 增强工具，简化 CRUD 操作，支持分页、条件查询与性能优化，提升数据库开发效率[6]。该组合搭配稳定、开源免费，是中小型数字化服务系统的主流数据层解决方案[7]。

3 系统设计

3.1 系统架构设计

本平台采用分层架构设计，确保系统的模块化、高效性和可扩展性。系统分为表现层、控制层、服务层、持久层和数据库层，每层有清晰的职责，且能够有效支持功能的扩展和维护。

表现层负责用户界面的呈现与交互，采用 Vue 3 框架实现响应式组件化开发，前端页面通过组件管理和数据绑定，提高了开发效率和用户体验[8]。Vue 3 的响应式机制使得页面能够及时响应用户操作，并高效地更新页面内容。前端与后端的通信通过 Axios 实现，前端通过 HTTP 请求与后端进行数据交换。

控制层则负责接收并处理来自前端的请求，使用 Spring MVC 进行请求分发。控制层将请求转发到具体的服务层进行业务处理，并将结果返回给前端。为了实现接口的高效管理和调用，控制层采用了 RESTful API 风格的接口设计，确保了系统接口的清晰和一致性。

服务层是系统的核心，处理具体的业务逻辑。在服务层，系统会根据前端传来的数据进行相应的计算和处理，然后调用持久层与数据库进行数据操作。服务层的设计确保了业务规则的封装和逻辑清晰，使得系统的扩展更加简便。

持久层基于 MyBatis-Plus 操作 MySQL 数据库，负责系统中的数据持久化工作。MyBatis-Plus 是 MyBatis 的增强工具，能够简化数据库操作，提升了开发效率和代码的简洁性。持久层实现了对数据库中各类实体（如用户、商品、订单、购物车、通知等）的增删查改功能，确保了数据与业务逻辑的良好协作。

数据库层则是系统的数据存储和管理部分。平台使用 MySQL 数据库来存储所有平台数据，包括用户信息、商品信息、订单记录、购物车数据、通知内容等。数据库表设计合理，并通过外键关系将不同数据表连接起来，确保了数据的完整性和一致性。为了应对高并发和海量数据，系统在数据库层设计了合理的索引和查询策略，保证了数据存取的高效性[6]。

该分层架构通过职责分离、松耦合设计，显著提升系统的可维护性、可扩展性与稳定性，使系统在功能扩展、日常运维及高并发访问场景下均能保持高效运行。

3.2 系统功能模块设计

系统功能以用户为核心，围绕农产品电商交易与农业信息服务构建模块化服务。用户模块支持注册、登录、个人资料修改、收货地址管理、收藏查看等功能，为用户提供统一的身份认证与个人账户管理服务；商品模块支持商家发布农产品图文信息、上传商品图片、编辑信息、上下架商品与库存管理，同时面向用户提供商品展示、分类浏览、详情查看与收藏功能，为平台交易提供基础内容支撑。购物车与订单模块构成平台核心交易机制，支持商品加购、数量调整、订单创建、支付、状态跟踪与售后处理，保障交易流程完整顺畅。

消息模块、客服模块与搜索模块共同提升平台交互效率与使用体验，消息模块对订单状态变更、商品互动、客服回复等行为提供实时通知提醒；客服模块支持用户与商家、管理员之间的在线沟通，满足售前咨询与售后反馈需求；搜索模块支持关键字模糊匹配，可快速查找商品、分类与农业资讯，大幅提升信息获取效率。管理员通过后台管理模块可全面监控用户与商家行为，审核商品信息、处理违规内容、管理分类与公告，并通过数据统计图表观测用户活跃度、订单量、交易额等关键指标，实时掌握平台整体运行状态，保障平台稳定、规范、安全运行。本系统主要包括前台用户子系统、商家管理子系统和管理员后台子系统三大核心模块，系统功能结构如图 3-1 所示：

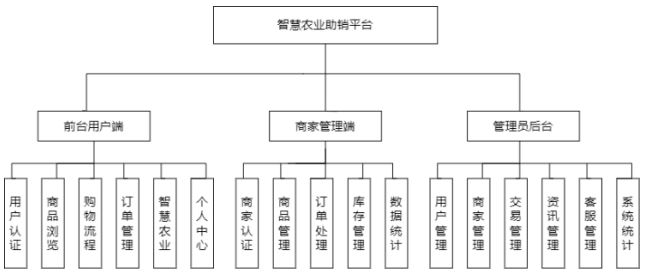

图 3-1 系统功能结构图

3.3 系统 E-R 图

系统总体 E-R 图如图 3-2 所示，涵盖了用户、商品、订单、订单明细、购物车、商品分类、评价、消息等核心业务实体及相互关联，清晰呈现数据库设计的逻辑结构与实体间的业务约束关系：

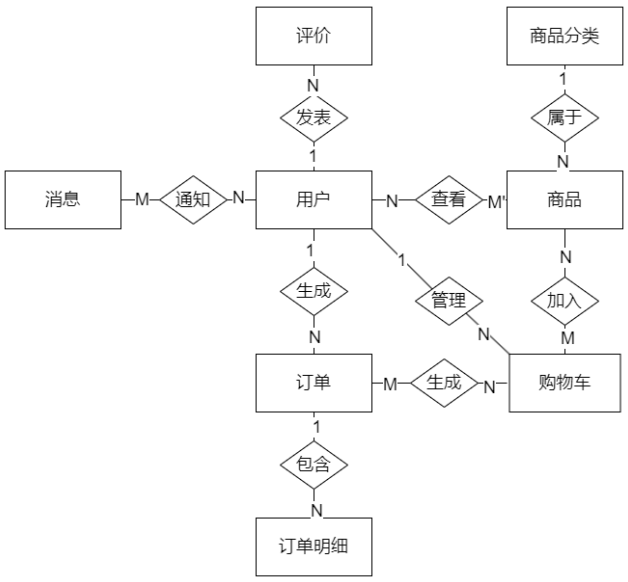

图 3-2 系统总体 E-R 图

3.4 数据库设计

系统采用 MySQL 8.0.33 数据库进行数据存储，通过主键、外键和索引进行关联，确保数据的完整性和查询效率。并根据业务需求设计了多张规范化的数据表，用于支撑平台的全功能运行。主要的数据库表包括：

（1）用户表：

用户表的说明如表 3-1 所示。

表 3-1 用户表

字段名称	类型	长度	空值	说明
id	bigint		否	用户 ID
username	varchar	50	否	用户名
password	varchar	255	否	密码
name	varchar	50	否	姓名
role	varchar	20	否	角色
email	varchar	100	是	邮箱
status	tinyint		否	状态
created_at	datetime		否	创建时间

（2）商品表：

商品表的说明如表 3-2 所示。

表 3-2 商品表

字段名称	类型	长度	空值	说明
id	bigint		否	商品 ID
name	varchar	100	否	商品名称
category_id	bigint		否	分类 ID
price	decimal	10,2	否	价格
stock	int		否	库存
sales	int		否	销量
description	text		是	商品描述
image	varchar	500	是	商品图片

（3）订单表：

订单表的说明如表 3-3 所示。

表 3-3 订单表

字段名称	类型	长度	空值	说明
id	bigint		否	订单ID
user_id	bigint		否	用户ID
order_no	varchar	50	否	订单编号
total_price	decimal	10,2	否	总价
status	varchar	20	否	订单状态
address	varchar	500	否	收货地址
phone	varchar	20	否	联系电话
created_at	datetime		否	创建时间

（4）购物车表

购物车表的说明如表 3-4 所示。

表 3-4 购物车表

字段名称	类型	长度	空值	说明
id	bigint		否	购物车ID
user_id	bigint		否	用户ID
product_id	bigint		否	商品ID
quantity	int		否	商品数量
selected	tinyint		否	选中状态
price	datetime		否	创建时间

（5）评价表

评价表的说明如表 3-5 所示。

表 3-5 评价表

字段名称	类型	长度	空值	说明
id	bigint		否	评价ID
user_id	bigint		否	用户ID
product_id	bigint		否	商品ID
content	text		是	评价内容
rating	int	20	否	评分
images	varchar	100	是	评价图片
created_at	datetime		否	创建时间

4 核心功能实现

4.1 用户模块实现

用户模块作为平台的身份入口与权限基础，承担用户注册、身份认证、会话管理、信息维护等核心职责，是保障系统安全、支撑多角色协同运行的关键模块。模块基于 Spring Security + JWT 实现无状态身份认证，结合 MyBatis Plus 完成用户数据的持久化与唯一性校验，通过统一响应封装、异常捕获、敏感字段脱敏等机制提升健壮性，可支撑用户、商家、管理员三类角色的全生命周期账号管理。

注册流程采用用户名唯一性校验机制，通过 LambdaQueryWrapper 构建查询条件，在用户数据入库前完成数据库层面的重复检查，避免脏数据写入；同时自动初始化账号状态与创建时间戳，保证用户数据规范性。登录流程基于用户名 / 密码比对完成身份核验，认证通过后调用 JWT 工具类生成携带用户 ID 与用户名的令牌，实现无状态会话管理；响应时主动清空密码字段，避免敏感信息泄露，符合接口安全设计规范。模块全程使用统一返回体 Result 封装结果，通过日志记录异常现场，便于问题定位与系统运维，为后续商品交易、订单操作、权限拦截提供可靠的身份支撑。

```
@Override
public Result register(User user) {
  try {
    if (this.lambdaQuery().eq(User::getUsername, user.getUserna
me()).exists()) {
      return Result.err(501, "用户名已存在");
    }
    user.setStatus(1); // 1 表示正常状态
    user.setCreatedAt(new Timestamp(System.currentTimeMillis
()));
    this.save(user);
    return Result.ok("注册成功");
    } catch (Exception e) {
    log.error("注册失败: username={}", user.getUsername(), e);
    return Result.err(500, "注册失败: " + e.getMessage());
  }
}

@Override
public Result login(String username, String password) {
  try {
    User user = this.lambdaQuery()
      .eq(User::getUsername, username)
      .eq(User::getPassword, password)
      .one();
    if (user != null) {
      String token = JwtTokenUtils.generateToken(user.getId(), u
ser.getUsername());
      user.setToken(token);
      user.setPassword(null); // 密码不返回
      return Result.ok("登录成功").data(user);
    }
    return Result.err(401, "用户名或密码错误");
  } catch (Exception e) {
    log.error("登录失败: username={}", username, e);
    throw new RuntimeException("登录失败", e);
  }
}
```

4.2 商品管理模块实现

商品管理模块是平台交易体系的核心数据支撑模块，主要实现农产品信息的发布、编辑、删除、查询、分类展示及状态管控等功能。模块基于 MyBatis Plus 实现商品数据的高效 CRUD 操作，通过条件构造器、分页插件、多维度查询等技术满足前端多样化的数据获取需求；同时支持商品状态管理、分类归属、详情信息维护，为用户浏览、搜索、选购农产品提供标准化数据接口。

在商品信息管理流程中，系统通过唯一性校验、字段完整性校验保证商品数据规范；采用分页查询技术降低单页数据加载压力，提升前端渲染效率与用户体验；商品信息修改与删除操作基于商品主键 ID 进行精准定位，确保数据操作的安全性与准确性。模块通过统一响应结果封装、异常捕获与日志记录，保证接口稳定性与可维护性，为购物车、订单等上游业务模块提供可靠的数据来源。

```
@Override
public Result createProduct(Product product) {
  try {
    if (StringUtils.isBlank(product.getName())) {
        return Result.err(501, "商品名称不能为空");
    }
    if (product.getPrice() == null || product.getPrice().compareTo
(BigDecimal.ZERO) < 0) {
        return Result.err(501, "商品价格必须大于等于 0");
    }
    if (product.getStock() == null || product.getStock() < 0) {
        return Result.err(501, "商品库存不能为负数");
    }
    if (product.getCategoryId() == null) {
        return Result.err(501, "商品分类不能为空");
    }
    product.setSalesCount(0);
    product.setStatus(1); // 1 表示上架状态
    product.setCreatedAt(new Timestamp(System.currentTimeMil
lis()));
    product.setUpdatedAt(new Timestamp(System.currentTimeMi
llis()));
    boolean success = this.save(product);
    return success ? Result.ok("商品发布成功") : Result.err(501,
 "商品发布失败");
  } catch (Exception e) {
    log.error("发布商品失败", e);
    throw new RuntimeException("发布商品失败：" + e.getMes
sage());
  }
}

@Override
public Result updateProduct(Product product) {
  try {
    Product existingProduct = this.getById(product.getId());
    if (existingProduct == null) {
        return Result.err(404, "商品不存在");
    }
    product.setUpdatedAt(new Timestamp(System.currentTimeMi
llis()));
    boolean success = this.updateById(product);
    return success ? Result.ok("商品更新成功") : Result.err(501,
 "商品更新失败");
  } catch (Exception e) {
    log.error("更新商品失败", e);
    throw new RuntimeException("更新商品失败：" + e.getMes
sage());
  }
}
```

4.3 订单模块实现

订单模块是平台交易闭环的核心模块，实现订单创建、订单支付、订单查询、订单删除等核心业务功能。模块基于 MyBatis Plus 实现订单数据的增删改查操作，通过状态字段管理订单生命周期，关联商品信息实现完整订单展示，同时在下单成功后自动清空对应用户购物车数据，保证交易流程连贯性。

订单创建为核心流程：系统接收前端传入的订单信息，自动生成创建时间戳并初始化订单状态为 “待支付”，完成订单数据持久化后，清空当前用户已结算的购物车数据，实现购物车与订单的数据联动。订单支付功能通过修改订单状态为 “已完成” 模拟支付流程，完成交易状态闭环。订单查询根据用户 ID 筛选对应订单数据，并关联查询商品详细信息，为前端展示订单与商品内容提供支撑。模块通过统一异常捕获、日志记录与标准响应体封装，保证接口运行稳定与数据安全。

```
@Transactional
@Override
public Result createOrder(Order order) {
  try {
    // 检查商品库存
    Product product = productService.getById(order.getProductId
());
    if (product == null) {
        return Result.err(404, "商品不存在");
    }
    if (product.getStock() < order.getQuantity()) {
        return Result.err(501, "商品库存不足");
    }

    // 扣减库存
    product.setStock(product.getStock() - order.getQuantity());
    productService.updateById(product);

    // 计算总价
    order.setTotalPrice(product.getPrice().multiply(new BigDecim
al(order.getQuantity())));
    order.setPrice(product.getPrice());
    order.setStatus(1); // 1 表示待支付
    order.setCreatedAt(new Timestamp(System.currentTimeMillis
()));
    order.setUpdatedAt(new Timestamp(System.currentTimeMilli
s()));

    boolean success = this.save(order);
    if (success) {
        // 增加商品销量
        product.setSalesCount(product.getSalesCount() + order.get
Quantity());
        productService.updateById(product);
    }
    return success ? Result.ok("订单创建成功") : Result.err(501,
 "订单创建失败");
  } catch (Exception e) {
    log.error("创建订单失败", e);
    throw new RuntimeException("创建订单失败：" + e.getMes
sage());
  }
```

```
}

@Override
public Result updateOrderStatus(Long orderId, Integer status) {
  try {
    Order order = this.getById(orderId);
    if (order == null) {
       return Result.err(404, "订单不存在");
    }
    order.setStatus(status);
    order.setUpdatedAt(new Timestamp(System.currentTimeMillis()));
    boolean success = this.updateById(order);
    return success ? Result.ok("订单状态更新成功") : Result.err(501, "订单状态更新失败");
  } catch (Exception e) {
    log.error("更新订单状态失败", e);
    throw new RuntimeException("更新订单状态失败：" + e.getMessage());
  }
}
```

4.4 购物车模块实现

购物车模块作为用户选购商品的临时存储与管理单元，承担商品添加、数量修改、条目删除、购物车列表查询等核心功能，是连接商品浏览与订单创建的关键中间环节。模块基于 MyBatis Plus 实现购物车数据的持久化操作，通过用户 ID + 商品 ID 联合校验机制，避免同一用户重复添加同一商品，保证购物车数据唯一性。

在功能实现上，添加商品时系统先校验用户购物车中是否已存在对应商品，存在则直接更新数量，不存在则新增记录；数量修改功能基于购物车记录 ID 进行精准更新，支持动态调整商品购买数量；列表查询功能根据当前登录用户 ID 筛选专属购物车数据，并关联查询商品详细信息，为用户展示完整的商品名称、图片、单价、小计金额等内容。模块通过事务保证数据操作一致性，采用统一返回格式与异常捕获机制，提升接口可靠性与用户操作体验。

```
@Override
public Result addToCart(Cart cart) {
  try {
    // 检查商品是否存在
    Product product = productService.getById(cart.getProductId());
    if (product == null) {
      return Result.err(404, "商品不存在");
    }

    // 检查是否已在购物车中
    Cart existingCart = this.lambdaQuery()
      .eq(Cart::getUserId, cart.getUserId())
      .eq(Cart::getProductId, cart.getProductId())
      .one();

    if (existingCart != null) {
    // 更新数量
       existingCart.setQuantity(existingCart.getQuantity() + cart.getQuantity());
       existingCart.setUpdatedAt(new Timestamp(System.currentTimeMillis()));
       boolean success = this.updateById(existingCart);
       return success ? Result.ok("购物车更新成功") : Result.err(501, "购物车更新失败");
    } else {
       // 新增购物车项
       cart.setCreatedAt(new Timestamp(System.currentTimeMillis()));
       cart.setUpdatedAt(new Timestamp(System.currentTimeMillis()));
       boolean success = this.save(cart);
       return success ? Result.ok("添加到购物车成功") : Result.err(501, "添加到购物车失败");
    }
  } catch (Exception e) {
    log.error("添加到购物车失败", e);
    throw new RuntimeException("添加到购物车失败：" + e.getMessage());
  }
}

@Override
public Result getCartList(Long userId) {
  try {
    List<Cart> cartList = this.lambdaQuery()
       .eq(Cart::getUserId, userId)
       .list();

    // 填充商品信息
    for (Cart cart : cartList) {
      Product product = productService.getById(cart.getProductId());
       if (product != null) {
         cart.setProduct(product);
       }
    }

    return Result.ok("获取购物车列表成功").data(cartList);
  } catch (Exception e) {
    log.error("获取购物车列表失败", e);
    throw new RuntimeException("获取购物车列表失败：" + e.getMessage());
  }
}
```

4.5 评价模块实现

评价模块作为用户对商品使用体验的反馈与分享单元，承担商品评价创建、状态管理、评价删除、评价列表查询等核心功能，是连接商品购买与口碑传播的重要环节。模块基于 MyBatis Plus 实现评价数据的持久化操作，通过用户 ID + 商品 ID 联合校验机制，确保评价数据的有效性和完整性。

在功能实现上，创建评价时系统先校验用户和商品是否存在，验证评分范围是否合法，处理图片列表的 JSON 序列化存储；状态更新功能基于评价 ID 进行精准更新，支持审核通过/不通过操作；列表查询功能根据不同维度（商品、用户、商家）筛选评价数据，并关联查询用户和商品详细信息，为前端展示完整的评价内容。模块通过统一的异常捕获与日志记录机制，提升接口可靠性与用户操作体验。

```
@param review
@return Result
@throws Exception
public Result<?> createReview(Review review) {
  try {
    // 检查用户是否存在
    User user = userMapper.selectById(review.getUserId());
    if (user == null) {
      return Result.error("-1", "用户不存在");
```

```
    }

    // 检查商品是否存在
    Product product = productMapper.selectById(review.getProductId());
    if (product == null) {
      return Result.error("-1", "商品不存在");
    }

    // 检查评分是否合法
    if (review.getRating() < 1 || review.getRating() > 5) {
      return Result.error("-1", "评分必须在 1-5 之间");
    }

    if (review.getImages() != null && !review.getImages().isEmpty()) {
      if (review.getImages().startsWith("[") && review.getImages().endsWith("]")) {
      } else {
          try {
              List<String> imageList = JSON.parseObject(review.getImages(), new TypeReference<List<String>>() {});
              review.setImages(JSON.toJSONString(imageList));
          } catch (Exception e) {
              // 如果解析失败，记录日志但不抛出异常
              LOGGER.warn("无法解析 images 字段为 JSON 数组: {}", e.getMessage());
          }
      }
    }
    // 情况 2：如果前端使用的是 imageList 字段
    else if (review.getImageList() != null && !review.getImageList().isEmpty()) {
      review.setImages(JSON.toJSONString(review.getImageList()));
    }

    // 创建评价
    int result = reviewMapper.insert(review);
    if (result > 0) {
     // 设置返回给前端的用户信息
      review.setUser(user);
      review.setProduct(product);
      // 设置返回的图片列表
      convertImagesToImageList(review);
      LOGGER.info("创建评价成功，评价 ID：{}", review.getId());
      return Result.success(review);
    }
    return Result.error("-1", "创建评价失败");
  } catch (Exception e) {
    LOGGER.error("创建评价失败：{}", e.getMessage());
    return Result.error("-1", "创建评价失败：" + e.getMessage());
  }
}
```

5 结束语

本文设计并实现了一个基于 Spring Boot+Vue 的智慧农业助销平台，系统实现了用户认证、商品管理、订单交易、在线支付、客服互动、数据统计等核心功能，具有良好的用户体验与完善的后台管理能力。可以有效解决农产品销售渠道单一、产销信息不对称、交易流程繁琐等痛点，具备较强的实用性与落地价值。未来，可以进一步引入 AI 智能推荐技术优化商品匹配效率，提升平台的精准服务与智能化水平；同时，完善移动端适配、打通物联网溯源、接入区块链信任机制也是系统后续优化与扩展的重要方向。

参 考 文 献

[1] 肖丛曦. 协同治理视角下Z县农村电商发展环境的优化路径研究[D]. 中共四川省委党校, 2024. DOI:10.27477/d.cnki.gzssd.2024.000046.

[2] 唐双林. 基于Vue和SpringBoot架构的智能推荐农产品团购销售系统[D]. 重庆三峡学院, 2023. DOI:10.27883/d.cnki.gcqsx.2023.000390.

[3] 刘玮欣. 面向可持续乡村服务的湖南省农产品电商平台设计[D]. 长沙理工大学, 2023. DOI:10.26985/d.cnki.gcsjc.2023.000874.

[4] 黄镓升, 邓舒婷. 基于Spring Boot的南宁旅游APP的设计与实现[C]//全国高等学校计算机教育研究会. 第32届计算机新科技与教育学术会议论文集. 南宁学院信息工程学院;, 2025:187-191. DOI:10.26914/c.cnkihy.2025.012290.

[5] 程哲桥, 魏小迪. 基于Vue+Echarts的高校体测可视化信息系统的研究与实现[C]//全国高等学校计算机教育研究会. 第32届计算机新科技与教育学术会议论文集. 南宁学院信息工程学院;, 2025:275-281. DOI:10.26914/c.cnkihy.2025.012305.

[6] 周贤武. 基于SpringBoot的高并发网购平台系统及其后台管理的设计与实现[D]. 哈尔滨理工大学, 2024. DOI:10.27063/d.cnki.ghlgu.2024.000336.

[7] 叶坤龙, 林宁, 左悦. 基于微信小程序的小区管家服务系统的设计与实现[C]//全国高等学校计算机教育研究会. 第32届计算机新科技与教育学术会议论文集. 南宁学院信息工程学院;南宁学院土木与建筑工程学院;, 2025:304-308. DOI:10.26914/c.cnkihy.2025.012310.

[8] 黄颖. 面向智慧农业的Vue3可视化编辑系统开发[D]. 湖北大学, 2024. DOI:10.27130/d.cnki.ghubu.2024.001819.