系统架构概述

本次实验旨在构建一个功能完备的网络商城系统，采用三层角色分离架构设计，实现用户、商家和管理员的功能隔离。系统采用前后端分离的开发模式，后端基于SpringBoot+MyBatis技术栈，前端使用Vue.js框架，通过RESTful API进行数据交互。

角色设计与功能隔离

用户端包含商品浏览与搜索功能，支持多维度筛选和排序，购物车管理和订单处理系统以及个人信息维护和交易历史查询；商户端包括商品信息管理（CRUD操作），库存管理与销售数据分析，订单处理与物流跟踪等等功能；管理端包括用户与商家资质审核以及平台数据监控与分析，系统参数配置设置功能。各角色界面、API接口和服务层完全独立，通过权限拦截器确保访问隔离，避免越权操作。

后端技术架构

数据持久层采用MySQL数据库的规范化表结构设计，减少数据冗余，并建立了索引优化了查询性能，并通过外键约束保证数据完整性。对于数据映射，该项目采用了MyBatis增强功能简化开发，并使用动态SQL处理复杂的查询条件，并采用二级缓存模式提升性能。

服务层的架构采用了模型（Model）层、仓库（Repository）层、服务（Service）层、控制（Controller）层的四层架构模式，其中模型层包含实体类（DO）和数据传输对象（DTO），仓库层包括数据访问接口和MyBatis映射文件，服务层通过先定义接口再实现接口的方式实现了业务逻辑，控制层是RESTful API的端点，用于处理HTTP请求。在架构的基础上，该项目完成了自定义注解实现了AOP切面编程，并且使用公共构件统一了异常处理机制，将参数验证与数据绑定，并提供了完整的日志记录与审计跟踪。

安全机制方面，采用Spring Security实现认证与授权，采用JWT令牌管理用户的会话，并将敏感数据加密存储，并且添加了CSRF防护和XSS过滤，有效防止了SQL注入和HTML注入；性能优化方面，采用了异步处理耗时操作，并将热点数据进行缓存，采用cookie保存用户登录数据并将用户足迹逐步保存进入数据库，并且采用了接口限流和熔断机制。

前端技术实现

基于Vue CLI搭建项目骨架，并采用单文件组件组织代码，并使用Vuex和Vue Router进行状态管理和路由管理，视图层采用响应式布局，适配多端设备，使用组件库构建界面，并采用了自定义的主题与样式，制作了渐入渐出的动画效果增强用户体验和画面流畅感。

工程化实践

项目统一使用IDEA SpringBoot+MyBatis+Jakarta+JDBC的Maven构建方式并统一代码风格，并构建Git仓库且将项目托管到GitHub上进行统一的版本控制