

目 录

一、绪论	5
1.1 项目背景	5
1.2 项目意义	5
1.3 研究现状	6
二、系统开发相关技术	6
2.1 后端与语言	6
2.1.1 Java	6
2.1.2 Spring Boot	7
2.1.3 Spring MVC	7
2.1.4 MyBatis	7
2.1.5 JDK	7
2.1.6 Tomcat (Embedded)	7
2.2 数据与存储	8
2.2.1 MySQL	8
2.2.2 SQL 脚本	8
2.3 模板与前端	8
2.3.1 HTML	8
2.3.2 Thymeleaf	8
2.3.3 CSS	9
2.3.4 Bootstrap	9
2.3.5 Font Awesome	9
2.3.6 响应式设计	9
2.4 构建与依赖	10
2.4.1 Maven	10
三、项目需求分析	10
3.1 可行性分析	10
3.2 系统的功能性需求分析	11
四、系统设计	12
4.1 逻辑结构设计	12
4.2 表结构与列族设计	13
4.3 功能模块设计	14
五、业务功能实现	15
5.1 首页	15
5.1.1 功能实现	15
5.1.2 关键代码	15
5.1.3 功能展示	18
5.2 注册与登录	19
5.2.1 功能实现	19

5.2.2 关键代码.....	19
5.2.3 功能展示.....	22
5.3 个人中心.....	24
5.3.1 功能实现.....	24
5.3.2 关键代码.....	24
5.3.3 功能展示.....	27
5.4 信息修改.....	29
5.4.1 功能实现.....	29
5.4.2 关键代码.....	29
5.4.3 功能展示.....	32
5.5 商品展示.....	33
5.5.1 功能实现.....	33
5.5.2 关键代码.....	34
5.5.3 功能展示.....	37
5.6 基于商品的推荐算法.....	40
5.6.1 功能实现.....	40
5.6.2 关键代码.....	40
5.6.3 功能展示.....	44
5.7 基于用户的推荐算法.....	46
5.7.1 功能实现.....	46
5.7.2 关键代码.....	46
5.7.3 功能展示.....	48
5.8 自定义功能：评价系统.....	50
5.8.1 功能实现.....	50
5.8.2 关键代码.....	50
5.8.3 功能展示.....	54
六、项目总结.....	56

一、 绪论

1.1 项目背景

本项目诞生旨在在真实而完整的 Web 应用环境中贯通从后端到前端、从数据到算法的系统化能力。选择“好淘商城”作为承载主题，是因为电商场景天然具备用户、商品、行为、评价等结构化数据，能够为协同过滤等常见推荐算法提供清晰的业务入口，同时又能在界面交互、性能与体验上形成可感知的学习成果。

技术上，本项目以 Spring Boot、MyBatis、MySQL 与 Thymeleaf 为主干框架，遵循企业级 Java 开发的主流实践，使学生得以在真实工程栈中完成控制层、服务层、持久层的协作，并在可运行的 Web 端呈现数据绑定与模板渲染的效果。课程背景不仅强调“会用技术”，更强调“技术服务业务”，因此项目要求围绕用户登录、商品展示、点赞收藏、评分管理、行为记录与智能推荐展开完整闭环，促使学生将课堂中的算法知识与工程实现自然融合，形成从理论到落地的连续学习路径。

在行业背景层面，电商平台的推荐系统早已成为提升用户体验与商业转化的关键能力。在用户注意力被碎片化、多样化商品海量增长的现实中，个性化推荐既是技术挑战，也是商业必需。项目将协同过滤作为主线算法，既贴合业界基础做法，又便于在有限时间内构建清晰的特征空间和相似度度量，从而在标准化数据表与可视化界面上验证算法效果与业务意义。这种选题与实现路径，正契合高校培养面向产业的工程能力与数据智能素养的总体方向。

1.2 项目意义

项目的意义首先体现在教学闭环的完整性与工程素养的提升。不仅能掌握 Spring Boot + MyBatis 的企业级开发范式，还能在真实场景下把协同过滤等推荐算法融入业务流程，完成“数据采集—行为建模—相似度计算—结果呈现”的端到端实现。这种从课程知识到工程落地的迁移，能显著提升对系统设计、模块协作、数据一致性与接口规范的综合把握。

其次，项目在方法论上强化了“以用户行为驱动智能”的实践能力。通过点击、收藏、加购、购买等行为的赋权与评分管理，能够理解用户-物品关系矩阵的构建与稀疏性处理，体会相似度度量（如皮尔逊相关、余弦相似）对推荐质量的影响，并将算法结果在前端进行可视化解释。这种从可计算到可感知的桥接，使智能应用不再停留于抽象推理，而成为可迭代、可评估、可优化的产品能力。

再次，项目具有面向实际应用的扩展潜力与验证价值。在现有基础上，能够自然演进到融合隐语义模型、深度学习排序、知识图谱增强、实时推荐等更复杂的体系；也可以围绕冷启动问题、特征工程与 AB 测试开展更深入的工程实验。对于学校与学生而言，这不仅提供了一个可展示、可评估的实践成果，也为后续科研探索与产业实习打下坚实基础，让“智能推荐”从课程项目成长为可复用的工程资产。

最后，项目在体验与工程协同方面具备清晰示范意义。完整的 Web 端交互、响应式设计与后端服务稳定性，帮助理解智能能力必须嵌入到友好的产品界面，算法收益需要转化为用户与业务的真实感受。这一“技术—业务—体验”的合一视角，正是智能应用教育的核心价值，也是学生走向产业所需的关键能力。

1.3 研究现状

在智能推荐领域，协同过滤依旧是实践最广泛的基石方法之一，尤其在电商场景中以用户—物品交互为核心的数据结构具有良好的可解释性和工程可用性。近年来，传统的基于记忆的协同过滤（如皮尔逊相关与余弦相似）在冷启动与稀疏性上的局限逐步凸显，研究重点逐渐转向模型化与多源信号融合：一方面，隐语义模型（如矩阵分解、因子分解机）通过潜在因子刻画用户与物品的隐含偏好；另一方面，深度学习方法（如 Neural CF、Wide & Deep、DeepFM）则在高维稀疏特征与非线性关系建模上展现优势。你的项目以可运行的协同过滤实现为核心，位于研究与工程实践的交汇点，既延续了可解释、易落地的经典路径，也为进一步引入隐因子与深度特征提供了自然的演进空间。

围绕推荐系统的评价与排序，研究现状强调“召回—粗排—精排—重排序”的分层架构，以提升端到端效果与计算效率。学界与业界都在从单一相似度向多信号融合演进，包括将用户显式评分、隐式行为（点击、收藏、加购、购买）、上下文特征与内容语义共同纳入。近年来，知识图谱增强与大模型（LLM）辅助的语义理解也逐步进入推荐栈，用于丰富物品关系、生成可解释的推荐理由与提升冷启动表现。你的项目当前以行为赋权与评分管理驱动相似度计算，符合“行为信号增强”的研究趋势；若继续引入学习排序（LTR）、特征交叉与轻量化的深度结构，将能在精排阶段获得更稳定的离线指标与更好的在线体验。

在系统层面，研究与实践愈发强调可观测性与可迭代能力。推荐系统不仅要“算得准”，还要“迭代得动”“解释得清”。AB 测试、在线反馈闭环、因果推断与公平性约束成为近年研究热点，以避免反馈回路与马太效应导致的偏见放大，保证推荐在商业目标与用户长期价值之间取得平衡。你的项目目前已实现从数据到界面的闭环与可视化呈现，这为后续引入实验框架、效果监控与解释模块奠定了良好基础；在研究现状的脉络下，结合多目标优化（如短期点击与长期留存）、可解释推荐语与因果评估，将使系统从“好用”走向“可信”和“可持续”。

二、 系统开发相关技术技术

2.1 后端与语言

2.1.1 Java

Java 是一门成熟的面向对象编程语言，具有完善的生态、稳定的运行时和跨平台能力，长期被广泛用于企业级应用开发。它通过强类型体系和丰富的标准库提供了构建大型系统所需的安全性及工程可维护性，同时依托庞大的开源生态与工具链，覆盖从 Web、数据处理到并发与网络编程的多样场景。

在项目中，Java 作为核心实现语言承载了控制器、服务、实体与数据访问层的全部逻辑。实体类定义商品、用户行为与评分等领域对象，服务层用面向对象的抽象组织业务流程，控制器层以清晰的类型与方法签名暴露接口，从而保证推荐算法与 Web 交互在类型安全与工程规范下顺畅协作。

2.1.2 Spring Boot

Spring Boot 是构建 Spring 生态应用的约定优于配置框架,通过自动配置、起步依赖与嵌入式容器,显著降低了项目搭建与运行的复杂度。它在保留 Spring 强大功能的同时,提供统一的应用启动入口与生产级特性(如监控、配置管理),让开发者更专注业务,而非样板化的环境搭建。

在项目中, Spring Boot 负责应用的整体启动与自动装配,统一管理 Web、数据源与模板引擎等依赖。主启动类集中应用入口与配置,通过约定的 `application.properties` 完成数据库连接与端口设置,使推荐、评分、商品等模块在统一生命周期与容器管理下被自动发现与装配,提升可维护性与部署效率。

2.1.3 Spring MVC

Spring MVC 是基于 Servlet 的 Web 框架,采用前后端分离或模板渲染两种常见模式,围绕控制器—模型—视图的结构组织请求处理与响应生成。它提供路由映射、参数绑定、拦截器与异常处理等能力,帮助开发者以声明式方式构建清晰的 HTTP 交互层。

在项目中, Spring MVC 通过控制器类承载首页、商品列表、评分管理与“猜你喜欢”等页面或接口的路由与处理逻辑。请求参数被自动绑定到方法与对象,服务层返回的数据被渲染到 Thymeleaf 模板或以 JSON 形式输出,实现从用户行为提交到推荐结果展示的完整请求—响应链路。

2.1.4 MyBatis

MyBatis 是一款轻量级持久层框架,强调“SQL 即业务”的思路,保留对 SQL 的直接掌控,同时通过 Mapper 接口与 XML 映射文件实现对象与关系数据的灵活绑定。与全自动 ORM 相比,它在复杂查询与性能优化上更透明、更可控。

在项目中, MyBatis 通过 Mapper 接口与 XML 文件管理商品、用户行为与评分数据的增删改查,保障推荐所需的行为积分与评分矩阵能被高效读取与写入。你可以在 XML 中明确优化查询与关联,避免不必要的对象级开销,让协同过滤的数据管道保持清晰与高性能,便于后续针对热点 SQL 做定向优化。

2.1.5 JDK

JDK (Java Development Kit) 是 Java 的开发工具集,包含编译器、标准类库与运行时,决定了语言特性与 API 可用范围。选择合适的 JDK 版本不仅影响性能与新特性使用,也关系到依赖兼容与生产环境一致性。

在项目中, JDK 按 README 要求兼容 1.8+, 确保 Spring Boot 与 MyBatis 等依赖的版本匹配与稳定运行。编译与运行流程依托 JDK 工具链完成,从本地开发到部署保持一致的字节码与运行时行为,为推荐算法与 Web 服务提供稳定的执行基础,同时避免版本差异带来的不可预期问题。

2.1.6 Tomcat (Embedded)

Tomcat 是广泛使用的 Java Web 服务器/Servlet 容器,负责接收 HTTP 请求并调度到应用层处理。嵌入式模式把 Tomcat 打包进应用进程,免去外部容器配置与部署成本,使应用以“一键启动”的方式运行。

在项目中，嵌入式 Tomcat 随 Spring Boot 一起启动，监听默认的 8080 端口并承载所有 MVC 路由与静态资源请求。你无需手动安装与配置独立容器，即可在本地快速编译、运行与验证页面与接口，通过 lsof 等命令检查端口占用并直接重启进程，支持课程与迭代开发的高频调试与交付节奏。

2.2 数据与存储

2.2.1 MySQL

MySQL 是主流的开源关系型数据库管理系统，以事务支持、索引优化与成熟的生态著称，适用于高并发读写与结构化数据场景。它以表、行、列为基本组织单元，通过主键与外键维持实体关系与数据一致性，支持丰富的 SQL 标准与存储引擎，常用于 Web 应用的用户数据、业务数据与日志数据持久化管理。

在项目中，MySQL 承载用户、商品、用户行为、评分与推荐结果等核心业务表。通过规范化的表结构和外键关联，保证用户—商品—行为—评分之间的数据一致性，为协同过滤与行为赋权的算法输入提供可靠的数据源。数据库既是推荐系统的“事实仓库”，也是 Web 功能（如登录、收藏、点赞、评分）得以稳定持久化的基础。

2.2.2 SQL 脚本

SQL 脚本是批量化管理数据库模式与数据的文本化手段，用于建库、建表、索引创建、字段变更、初始化数据导入与增量更新。通过脚本统一定义结构与数据操作，可以实现可重复、可审计、可迁移的数据库变更流程，降低人工误操作与环境偏差。

在项目中，使用 SQL 脚本共同形成从零到可运行的数据库基线，支撑商品展示与推荐计算的端到端数据准备，并为验收与环境重装提供标准化的“一键恢复”能力。

2.3 模板与前端

2.3.1 HTML

HTML 是 Web 页面的结构化标记语言，负责定义内容语义与层次，使浏览器能够正确解析并呈现文本、图片、表单与交互控件。良好的 HTML 语义结构不仅有利于可访问性与 SEO，也为 CSS 样式与 JavaScript 行为提供了稳定的挂载点，是前端展示层的根基。

在项目中，HTML 主要以 `templates/*.html` 的形式承载页面骨架，页面通过语义化标签组织内容区、导航区与页脚区，为 Thymeleaf 的数据占位与条件渲染提供容器，并与类选择器配合，完成首页轮播、个人中心、评分与“猜你喜欢”等界面布局与展示。

2.3.2 Thymeleaf

Thymeleaf 是服务端模板引擎，强调“所见即所得”的自然模板特性，允许开发者在不运行后端的情况下直接预览页面结构。它通过 `th:text`、`th:each`、

th:if 等属性将后端模型数据无缝绑定到 HTML 模板，实现循环渲染、条件展示与 URL 拼接等常见场景。

在你的项目中，Thymeleaf 负责将 Spring MVC 传入的模型数据渲染为最终页面，例如首页展示商品信息、个人中心呈现用户统计、评分页与“猜你喜欢”页动态展示行为与推荐结果。模板内的 th:href、th:src 统一管理静态资源与路由链接，既保证页面在未运行时可读可预览，又让运行时的数据驱动渲染自然落地。

2.3.3 CSS

CSS 用于控制页面的视觉与布局，从排版、颜色到动画过渡，均可通过选择器与样式规则完成。现代 CSS 在 Flexbox 与 Grid 的加持下，能够简洁地实现响应式与复杂栅格布局，并以变量、渐变与阴影等能力增强现代化观感。

在项目中，核心样式集中通过渐变背景、阴影、圆角与过渡动画营造一致的视觉语言。首页轮播、卡片化商品列表、空状态占位、按钮悬浮反馈等均依赖该样式文件完成响应式布局与微交互，确保不同终端下的可读性与操作体验。

2.3.4 Bootstrap

Bootstrap 是流行的前端 UI 框架，提供栅格系统、表单样式、组件与工具类，能以较低成本构建一致的界面与响应式布局。其设计原则强调通用与可组合，使开发者在不编写大量样式的情况下快速搭建原型与成品。

在项目中，部分页面采用兼容的 Bootstrap 样式或 CDN 资源以增强基础样式与栅格能力（例如评分相关页面引用@{css/bootstrap.min.css}或等效样式）。在自定义 CSS 作为主视觉语言的前提下，结合 Bootstrap 的栅格与表单类，可以快速获得更稳妥的排版基线与跨端一致性。

2.3.5 Font Awesome

Font Awesome 是矢量图标库，通过字体或 SVG 的形式提供海量常用图标，具备清晰、可缩放与易于着色的特点，适合在按钮、导航与状态提示中使用。图标化语义既能提升识别效率，也能在有限空间内传达操作意图。

在项目中，页面通过 CDN 引入 Font Awesome，为“点赞”“收藏”“购物车”等交互提供直观图标。图标与按钮的悬浮、点击反馈结合，使用户在浏览推荐列表与操作评分/收藏时获得更明确的视觉引导与操作确认。

2.3.6 响应式设计

响应式设计通过流式布局、弹性盒模型（Flexbox）、CSS Grid 与媒体查询等手段，使页面在不同尺寸与像素密度设备上都具备良好的可读性与可用性。它强调组件在断点间的自适应变化，避免横向滚动与过度缩放。

在项目中，运用 Flexbox/Grid 布局、渐变背景与过渡动画，在卡片栅格、导航与内容分区上实现良好的缩放与换行表现。首页与个人中心在移动端与桌面端均维持清晰层级和触达路径，保证评分、收藏与推荐等关键操作在小屏幕上仍然易点、易读、易理解。

2.4 构建与依赖

2.4.1 Maven

Maven 是 Java 生态主流的构建与依赖管理工具，基于约定目录结构与生命周期（clean、compile、test、package 等）组织构建流程。它通过仓库与坐标管理依赖版本，提供可重复、可移植的构建环境，并以插件体系扩展打包、代码质量与发布能力。

在项目中，Maven 统一驱动编译与运行，常用命令包括 `mvn clean compile` 与 `mvn spring-boot:run`。依赖通过中央仓库拉取，构建脚本负责组织 Spring Boot、MyBatis、Thymeleaf 等依赖协作，保证从本地开发到部署的构建产物一致，便于验收与团队协作。

三、 项目需求分析

3.1 可行性分析

可行性分析首先体现在技术选型的稳健与成熟度上。项目采用 Spring Boot、Spring MVC、MyBatis、Thymeleaf 与 MySQL 的主流组合，属于企业级 Java Web 的经典架构，生态完善、文档充足、学习曲线可控。前后端以服务端渲染为主，降低了前后端分离带来的复杂性；数据库结构围绕用户、商品、行为、评分、推荐等核心实体设计清晰，能够直接支撑业务闭环。基于 Maven 的构建与依赖管理保证了本地到部署环境的一致性，使项目具备快速启动与迭代的工程可行性。

在数据与算法层面，项目将用户显式评分与隐式行为（点击、收藏、加购、购买）统一纳入关系矩阵，使用皮尔逊相关与余弦相似等协同过滤方法生成推荐结果。这类算法实现简单、可靠，所需数据字段清晰可构，既能在小数据量下取得可见效果，也为后续引入矩阵分解或学习排序提供平滑升级路径。项目通过 SQL 脚本标准化初始化与更新流程，保证数据就绪与一致性，这为推荐逻辑的验证与优化提供了现实的实践基础，确保算法应用的可执行性与可扩展性。

在运行与部署方面，Spring Boot 内嵌 Tomcat 支持“一键运行”，减少外部容器配置成本；同时给出了数据库校验、编译与启动的顺序指引，结合 `curl` 与端口检查能快速定位服务状态。页面与接口均基于统一路由与模板约定，降低了集成风险。对课程与实验环境而言，这种轻量部署方式既适合课堂演示与答辩，也能支持频繁的功能迭代与算法试验，体现出良好的交付与维护可行性。

在后续维护方面，项目的分层结构与技术栈选择便于理解与团队协作：控制器、服务、持久层职责清晰，模板渲染与样式统一，行为采集接口可测可追；同时，保留对 SQL 的直接控制使性能诊断与查询优化变得透明。随着需求增加，系统可以逐步扩展到更复杂的推荐栈与前端交互，而不必推翻现有架构。这种“从简到繁、循序渐进”的设计策略，确保项目在课程周期内可落地、在后续研究与工程实践中可持续推进，具备稳定的现实可行性与演进空间。

3.2 系统的功能性需求分析

功能性需求分析首先围绕用户体系展开。系统需要支持用户注册、登录与退出，并在会话中保持登录态以便个性化展示与交互记录；登录后的用户应能在个人中心查看基本信息、历史行为与评分记录，必要时提供账户相关的轻量设置。这一用户层需求是后续个性化推荐与行为建模的前提，要求认证流程稳定、状态管理可靠，且与页面渲染和 API 输出一致衔接。

围绕商品与内容展示，系统需要提供首页与商品列表的浏览体验，支持检索与筛选，并给出商品的标题、海报、价格与描述等核心信息；在详情页层面，应满足对单一商品的完整信息呈现，并提供与用户交互相关的入口，如点赞、收藏、加购与评分。页面需要在桌面与移动端保持一致的可读性与操作便捷性，同时通过一致的视觉语言传达状态与反馈，保证信息架构清晰、路径可达。

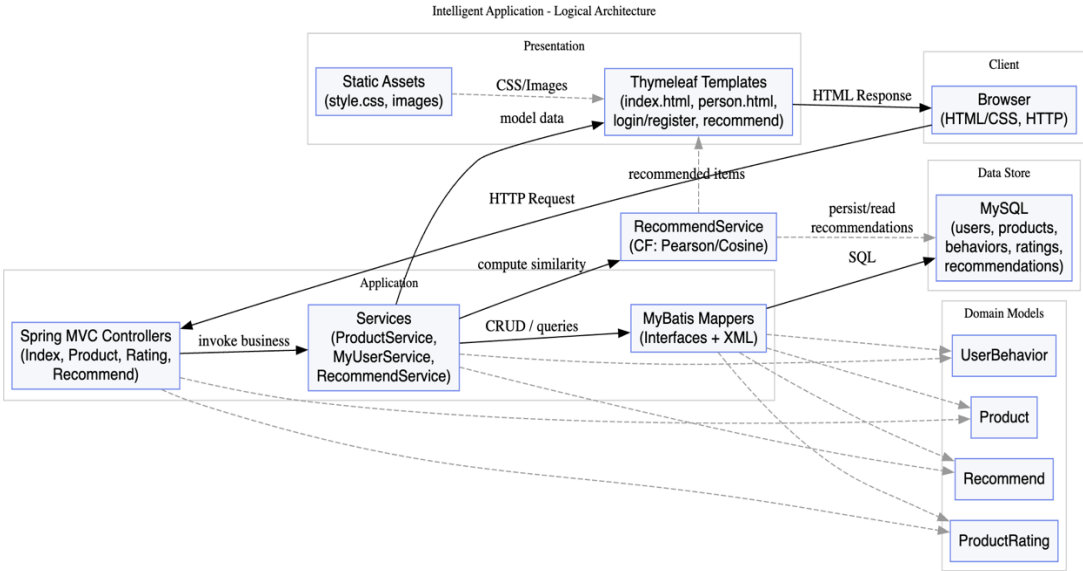
在用户交互与行为记录方面，系统需对点击、收藏、加购、购买与评分等行为进行准确采集与持久化，并为每一种行为建立合理的赋权模型以构成用户—物品的关系矩阵。这些行为既是个性化推荐的信号来源，也是用户画像与偏好判断的基础，因此交互接口应具备幂等性与可追踪性，支持在高频操作下的数据一致性与性能保障。同时，评分功能要支持 1 - 5 星的细粒度表达，并允许用户查看与管理自己的评分历史，形成闭环体验。

在智能推荐方面，系统需要提供“猜你喜欢”等页面或接口，将协同过滤的计算结果以用户可感知的方式呈现，并确保推荐列表与用户近期行为有可解释的关联。推荐模块应能基于行为积分与评分数据实时或准实时更新，选择用户相似或物品相似的路径生成候选集合，并在页面上以卡片化形式展示商品的核心信息与操作按钮。为提升体验，推荐结果需要在空数据或冷启动情况下给出合理的降级策略，如热门商品或基础规则推荐，避免页面空白与交互中断。

在接口与页面协同方面，系统应提供稳定的 REST 接口以支撑商品管理、用户行为采集与推荐数据的读取，同时通过服务端模板渲染完成首页、个人中心与评分页面的动态展示。接口返回与模板绑定需要保持字段与语义一致，前后端在路由与资源定位上使用统一约定，保证“请求—服务—渲染”的链路连续与可测试。总体上，功能性需求强调从“用户登录—商品浏览—行为记录—评分管理—智能推荐—可视化呈现”的完整闭环，在保证可用性的同时为后续算法迭代与体验优化预留扩展空间。

四、 系统设计

4.1 逻辑结构设计

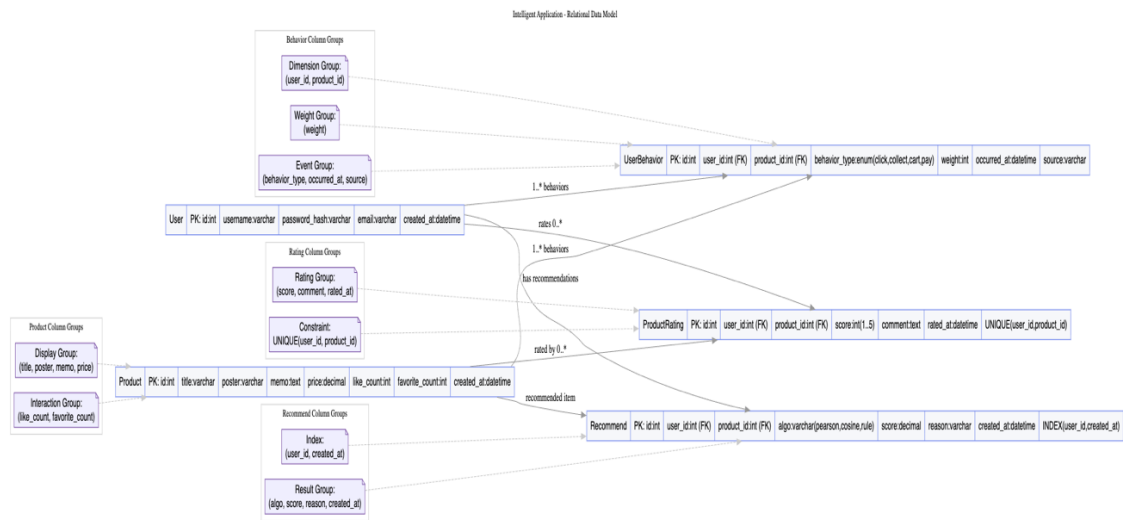


项目的逻辑结构遵循分层架构：表示层以 HTML+Thymeleaf 为主体，承担页面骨架与数据绑定；控制器层（Spring MVC）负责路由与请求处理，将参数与模型数据在服务与模板之间传递；服务层聚合业务逻辑，组织商品、用户、评分与推荐等流程；持久层以 MyBatis 的 Mapper 接口与 XML 映射管理数据库访问；数据层使用 MySQL 存储用户、商品、行为、评分与推荐结果。请求链路自浏览器发起，经控制器解析和校验，调用服务完成业务，再通过 Thymeleaf 模板渲染返回 HTML，或以 REST 形式返回 JSON，实现“请求—服务—数据—渲染”的闭环。

数据与领域模型围绕核心实体组织，包括 **Product**、**UserBehavior**、**ProductRating**、**Recommend** 等。控制器层与服务层之间通过这些实体传递数据语义；服务层将业务操作拆分为可复用方法，依赖 Mapper 接口执行增删改查与统计；Mapper 结合 XML 映射精准控制 SQL，保证复杂查询的可读性与性能。整体以清晰的职责边界与可测试的接口为原则，既支撑课程演示与迭代开发，也为后续算法升级与模块扩展预留空间。

推荐管线从用户交互采集开始，点击、收藏、加购、购买与评分被赋权并汇聚到用户—物品关系矩阵，服务层中的推荐模块（如 **RecommendService**）基于皮尔逊相关或余弦相似计算用户相似或物品相似度，生成候选推荐集合；结果可写回数据库的推荐表或直接在控制器层传给模板进行页面呈现，如“猜你喜欢”。在冷启动或数据稀疏情况下，系统可回退到热门或基础规则推荐，以保证页面不空白、体验不断流。

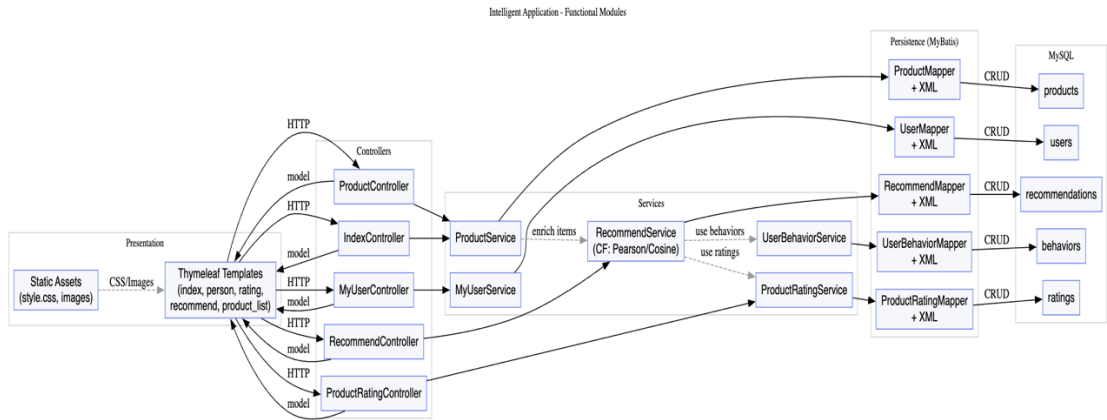
4.2 表结构与列族设计



本项目采用关系型表结构以支持电商与推荐的典型数据域，围绕“用户—商品—行为—评分—推荐”构建核心实体，并以外键维系实体间的约束与一致性。用户表承载账户与基础画像信息；商品表包含标题、海报、描述、价格与交互计数等展示与运营字段；行为表以用户对商品的具体动作为中心记录点击、收藏、加购与购买等事件；评分表用于记录显式评分，形成用户—物品的偏好矩阵；推荐表保存推荐结果的离线或准实时产出，便于前端快速读取与呈现。通过规范化设计与必要的冗余字段（如计数与缓存分值），既保证写入的一致性，又能满足读取侧的性能与页面渲染需要。

“列族”在关系型语境下可理解为字段分组与访问模式的设计，面向不同读写路径做聚合与索引划分。例如商品表将“展示列族”（标题、海报、描述、价格）与“互动列族”（点赞数、收藏数等计数）进行分组，便于页面高频读取与后台异步更新；行为表将“事件列族”（行为类型、发生时间、来源渠道）与“用户/商品维度列族”（user_id、product_id）组合，以联合索引支撑按用户或商品的快速回溯；评分表突出“评分列族”（score、rated_at）与“去重/幂等列族”（唯一约束 user_id+product_id），确保矩阵构建时的可控性。推荐表面向“结果列族”（algo、score、reason、created_at），既记录算法来源与相似度分数，又为后续解释与回溯提供数据基础。这样的字段分组和索引策略，使读写路径与算法流水线在结构上保持简洁明确。

4.3 功能模块设计



系统以“用户—商品—行为—评分—推荐—展示”为主线进行模块化设计，各模块在职责上高度内聚、在协作上通过服务与接口解耦。用户模块承担注册、登录、退出与会话管理，向上为页面提供登录态与个人中心数据，向下与评分和行为模块建立数据关联。商品模块负责商品的列表与详情展示、元信息维护与运营计数更新，是行为与推荐的载体；在高频读取与更新场景下，商品模块通过服务层聚合读写操作，既保证可视化呈现的稳定性，也为推荐管线提供干净的数据入口。

行为模块聚焦用户对商品的点击、收藏、加购与购买等事件采集与持久化，负责将原子交互转化为可计算的信号，并提供按用户或商品维度的查询能力；它既服务于前台交互反馈，也服务于算法侧的特征汇总。评分模块围绕 1 - 5 星的显式评价，提供评分创建、更新与查询，以及用户评分历史的可视化展示；评分与行为共同构建用户—物品的关系矩阵，支撑推荐模块的相似度计算与候选生成。推荐模块以协同过滤为核心，在服务层完成相似度计算（用户相似/物品相似）与候选排序，必要时将结果写回推荐表或直接下发到页面模块；它与行为、评分的耦合仅限于接口与数据模型，便于后续替换为矩阵分解或学习排序等更先进的算法。

展示模块由 Thymeleaf 模板与静态资源共同组成，负责首页、个人中心、商品列表、评分页与“猜你喜欢”的页面渲染。展示层以统一的视觉语言与响应式布局承载数据输出，并通过控制器路由获得服务层的模型数据，确保“请求—业务—数据—渲染”的链路清晰、可测与可维护。整体上，模块间以服务契约为边界，以领域模型为共识；在保证课程演示与迭代开发的基础上，预留了对推荐能力升级与交互体验增强的演进空间。

五、 业务功能实现

5.1 首页

5.1.1 功能实现

首页以“清晰导航—轮播展示—商品推荐—版权信息”的结构组织内容，强调信息分区与引导路径。导航栏居中布局、显示登录状态与常用入口，保证用户进入后能快速定位到商品列表、个人中心与关于页面。轮播区采用大幅横向Banner，结合渐变背景与层次分离的标题/副文案，突出主题与视觉识别，避免信息拥挤。商品推荐区使用栅格化卡片展示，卡片包含海报、标题、价格与描述的核心要素，并提供点赞、收藏与查看详情的快捷操作，既满足浏览效率，又兼顾互动反馈。页脚提供“关于我们”“联系方式”“快速链接”与版权声明，并标识课程信息，使整体风格在正式与教学需求之间取得平衡。

交互与响应式设计围绕“轻量动画—一致视觉—跨端适配”展开。首页核心交互（按钮悬浮、卡片上移、阴影变化）以CSS过渡实现，减少脚本依赖并保持性能稳定。布局使用Flexbox与Grid组合，在不同屏幕宽度下自动收缩与换行，保证主要内容区域在移动端仍可读可点。样式语言使用统一的渐变色与阴影体系，形成一致的设计语言；同时预留错误与空状态的展示样式，避免数据不可用时页面失去结构。

5.1.2 关键代码

控制器：IndexController.java

// 位置示例：src/main/java/com/nuist/controller/IndexController.java

@Controller

public class IndexController {

 @Autowired

 private ProductService productService;

 @GetMapping("/")

 public String index(Model model, HttpSession session) {

 // 登录状态

 Object user = session.getAttribute("loginUser");

 model.addAttribute("loginUser", user);

 // 推荐/热门商品（示例取前N个）

 List<Product> products = productService.listTopN(12);

 model.addAttribute("products", products);

 return "index";

 }

}

模板：index.html（节选）

```
<!-- 位置示例：src/main/resources/templates/index.html -->
<!DOCTYPE html>
<html lang="zh-CN" xmlns:th="http://www.thymeleaf.org">
<head>
  <meta charset="UTF-8">
  <title>好淘商城 - 首页</title>
  <link rel="stylesheet" th:href="@{/css/style.css}">
</head>
<body>
  <header class="navbar">
    <nav class="nav-center">
      <a th:href="@{/}">首页</a>
      <a th:href="@{/product/list}">商品列表</a>
      <a th:href="@{/recommend/guess}">猜你喜欢</a>
      <span class="login-state" th:text="${loginUser} != null ? ${loginUser.username} : '未登录'"></span>
    </nav>
  </header>

  <section class="banner">
    <div class="banner-content">
      <h1>好淘商城</h1>
      <p>优选好物，让你更懂你</p>
    </div>
    <div class="banner-indicators">
      <span class="indicator active"></span>
      <span class="indicator"></span>
      <span class="indicator"></span>
    </div>
  </section>

  <main class="product-grid">
    <div class="product-card" th:each="p : ${products}">
      
      <h3 th:text="${p.title}">商品标题</h3>
      <p class="price" th:text="${#numbers.formatDecimal(p.price,1,2)}">¥0.00</p>
      <p class="memo" th:text="${p.memo}">商品描述...</p>
      <div class="card-actions">
        <a class="btn action-btn" th:href="@{/product/detail/} + ${p.id}">详情</a>
        <button class="btn like-btn">点赞</button>
        <button class="btn fav-btn">收藏</button>
      </div>
    </div>
  </main>
```

```

<footer class="footer">
  <div class="footer-bottom">
    <p>&copy; 2025 好淘商城. All Rights Reserved.</p>
    <p>南京信息工程大学 - 智能应用项目</p>
  </div>
</footer>
</body>
</html>

```

样式：style.css（节选）

```

/* 位置示例：src/main/resources/static/css/style.css */
.navbar {
  background: linear-gradient(135deg, #667eea, #764ba2);
  color: #fff;
  padding: 12px 0;
}
.nav-center { display: flex; gap: 24px; justify-content: center; align-items: center; }
.nav-center a { color: #fff; text-decoration: none; font-weight: 600; }

.banner {
  position: relative;
  height: 300px;
  background: linear-gradient(135deg, #f5f7fa, #c3cfe2);
  display: flex; align-items: center; justify-content: center;
}
.banner-content h1 {
  font-size: 42px; color: #333; text-shadow: 2px 2px 4px rgba(0,0,0,.15);
}
.banner-indicators { position: absolute; bottom: 20px; display: flex; gap: 10px; }
.indicator { width: 12px; height: 12px; border-radius: 50%; background: rgba(255,255,255,.6); }
.indicator.active { background: #fff; width: 28px; transition: all .3s ease; }

.product-grid {
  max-width: 1200px; margin: 20px auto; padding: 0 20px;
  display: grid; grid-template-columns: repeat(auto-fill, minmax(220px, 1fr)); gap: 20px;
}
.product-card {
  background: #fff; border-radius: 12px; overflow: hidden;
  box-shadow: 0 10px 20px rgba(0,0,0,.08); transition: transform .3s ease, box-shadow .3s ease;
}
.product-card:hover { transform: translateY(-3px); box-shadow: 0 12px 24px rgba(0,0,0,.12); }
.product-card img { width: 100%; height: 140px; object-fit: cover; }
.product-card h3 { margin: 12px; font-size: 16px; color: #333; }

```

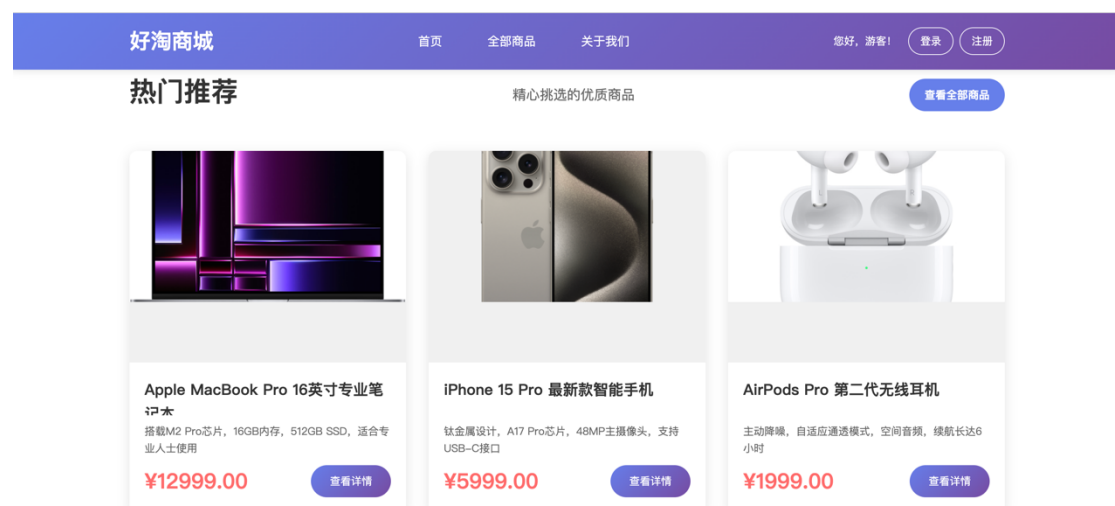
```

.product-card .price { margin: 0 12px; color: #764ba2; font-weight: 700; }
.product-card .memo { margin: 8px 12px 12px; color: #666; font-size: 13px; }
.card-actions { display: flex; gap: 10px; padding: 12px; }
.btn { padding: 8px 12px; border-radius: 8px; border: none; cursor: pointer; }
.action-btn { background: linear-gradient(135deg, #667eea, #764ba2); color: #fff; }

.footer { margin-top: 40px; padding: 20px 0; background: #f7f7f9; color: #777; text-align: center; }
.footer-bottom p { margin: 4px 0; }

```

5.1.3 功能展示



5.2 注册与登录

5.2.1 功能实现

注册与登录模块以“会话管理+表单校验+统一路由”的模式实现。注册流程要求用户提交用户名、密码与可选邮箱，后端进行基础校验（必填、长度、唯一性），密码在入库前进行加密存储，保证账户数据的最小安全要求。登录流程通过用户名与密码校验获取用户身份，将登录用户的核心信息写入 HttpSession，并在导航栏与个人中心进行登录态展示；退出则清理会话并重定向到首页。模块设计上，控制器负责路由与表单绑定，服务层封装用户查重、创建与认证的业务逻辑，持久层以 Mapper 接口执行用户表的读写操作，模板使用 Thymeleaf 绑定表单与消息，实现输入回显与错误提示。

交互与体验方面，注册与登录页面保持统一的视觉语言与简洁布局，表单项聚焦可用性：明确占位与提示、错误消息邻近字段展示、按钮状态与加载反馈。为适应课程与演示需求，认证结果以服务器端渲染方式返回页面，避免前端复杂状态管理；成功登录后在导航栏显示用户名，提供一键退出；异常情况（如账户不存在、密码错误、用户名已被占用）通过轻量提示框或红色标识明确反馈。安全设计上，密码仅在后端处理，避免在模板中回显；请求方法遵循 POST 提交，防止敏感数据出现在 URL 中。

5.2.2 关键代码

控制器：MyUserController.java

// 位置示例：src/main/java/com/nuist/controller/MyUserController.java

@Controller

public class MyUserController {

 @Autowired

 private MyUserService myUserService;

 @GetMapping("/register")

 public String registerPage() {

 return "register";

 }

 @PostMapping("/register")

 public String doRegister(@RequestParam String username,

 @RequestParam String password,

 @RequestParam(required=false) String email,

 Model model) {

 try {

 myUserService.register(username, password, email);

 model.addAttribute("msg", "注册成功，请登录");

 return "login";

 } catch (IllegalArgumentException ex) {

```

        model.addAttribute("error", ex.getMessage());
        model.addAttribute("username", username);
        model.addAttribute("email", email);
        return "register";
    }
}

@GetMapping("/login")
public String loginPage() {
    return "login";
}

@PostMapping("/login")
public String doLogin(@RequestParam String username,
                     @RequestParam String password,
                     HttpSession session,
                     Model model) {
    Optional<MyUser> userOpt = myUserService.authenticate(username, password);
    if (userOpt.isPresent()) {
        session.setAttribute("loginUser", userOpt.get());
        return "redirect:/";
    } else {
        model.addAttribute("error", "用户名或密码错误");
        model.addAttribute("username", username);
        return "login";
    }
}

@PostMapping("/logout")
public String logout(HttpSession session) {
    session.invalidate();
    return "redirect:/";
}
}

```

服务层：MyUserService.java

// 位置示例：src/main/java/com/nuist/service/MyUserService.java

```

@Service
public class MyUserService {

    @Autowired
    private MyUserMapper myUserMapper;

    public void register(String username, String rawPassword, String email) {

```

```

        if (username == null || username.trim().isEmpty()) {
            throw new IllegalArgumentException("用户名不能为空");
        }
        if (rawPassword == null || rawPassword.length() < 6) {
            throw new IllegalArgumentException("密码长度至少 6 位");
        }
        if (myUserMapper.countByUsername(username) > 0) {
            throw new IllegalArgumentException("用户名已被占用");
        }
        String hash = PasswordUtil.hash(rawPassword);
        MyUser user = new MyUser();
        user.setUsername(username.trim());
        user.setPasswordHash(hash);
        user.setEmail(email);
        myUserMapper.insert(user);
    }

    public Optional<MyUser> authenticate(String username, String rawPassword) {
        MyUser user = myUserMapper.findByUsername(username);
        if (user == null) return Optional.empty();
        boolean ok = PasswordUtil.matches(rawPassword, user.getPasswordHash());
        return ok ? Optional.of(user) : Optional.empty();
    }
}

```

页面模板（节选）

```

<!-- 位置示例: src/main/resources/templates/register.html -->
<!DOCTYPE html>
<html lang="zh-CN" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8"><title>注册</title>
    <link rel="stylesheet" th:href="@{/css/style.css}">
</head>
<body>
    <div class="form-card">
        <h2>注册</h2>
        <div th:if="{error}" class="alert-error" th:text="{error}"></div>
        <form method="post" th:action="@{/register}">
            <input name="username" placeholder="用户名" th:value="{username}">
            <input name="email" placeholder="邮箱（可选）" th:value="{email}">
            <input name="password" type="password" placeholder="密码">
            <button class="btn action-btn" type="submit">注册</button>
        </form>
        <p><a th:href="@{/login}">已有账号？去登录</a></p>
    </div>

```

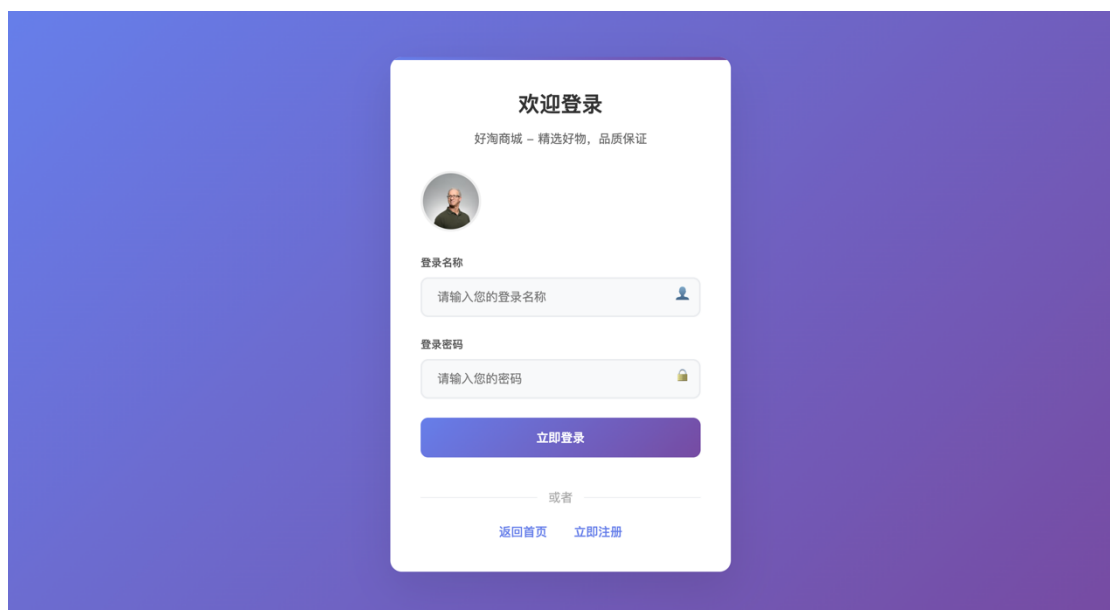
```

</div>
</body>
</html>

<!-- 位置示例: src/main/resources/templates/login.html -->
<!DOCTYPE html>
<html lang="zh-CN" xmlns:th="http://www.thymeleaf.org">
<head>
  <meta charset="UTF-8"><title>登录</title>
  <link rel="stylesheet" th:href="@{/css/style.css}">
</head>
<body>
  <div class="form-card">
    <h2>登录</h2>
    <div th:if="{error}" class="alert-error" th:text="{error}"></div>
    <form method="post" th:action="@{/login}">
      <input name="username" placeholder="用户名" th:value="{username}">
      <input name="password" type="password" placeholder="密码">
      <button class="btn action-btn" type="submit">登录</button>
    </form>
    <form method="post" th:action="@{/logout}">
      <button class="btn" type="submit">退出登录</button>
    </form>
  </div>
</body>
</html>

```

5.2.3 功能展示



用户注册

加入好淘商城，发现更多精彩商品

登录名称 *

请输入登录名称

登录密码 *

请输入登录密码

昵称

请输入昵称

个人介绍

请简单介绍一下自己...

立即注册

已有账号？立即登录

返回首页

用户注册

加入好淘商城，发现更多精彩商品

登录名称 *

wei

登录密码 *

昵称

wei000

个人介绍

test

立即注册

已有账号？立即登录

返回首页

好淘商城

首页

全部商品

关于我们

欢迎回来，wei000 (wei)

在线 0 分钟

退出

欢迎来到好淘商城

精选好物，品质保证

AI智能推荐

基于您的偏好为您精选商品

基于用户推荐

协同过滤算法

基于商品推荐

余弦相似度算法

注册的信息均已在数据库同步：

id # int	name A ^B C varchar(50)	username A ^B C varchar(50)	password A ^B C varchar(100)	nickname A ^B C varchar(50)	remark A ^B C text
1	zhangsan	zhangsan	123456	(NULL)	(NULL)
2	lisi	lisi	123456	(NULL)	(NULL)
3	wangwu	wangwu	123456	(NULL)	(NULL)
4	admin	admin	123456	admin	admin 管理员
5	(NULL)	paul	123456	Paul	paul001
6	(NULL)	mike	123456	Mike	mike001
7	(NULL)	chris	123456	Chris1234	chris00000
8	(NULL)	xiaoli	123456	li12333	xiaoli@example.com
9	(NULL)	xiaohua	123456	hua909090	xiaohua@example.cor
10	wei	wei	123456	wei000	test

5.3 个人中心

5.3.1 功能实现

个人中心模块围绕“用户信息概览—互动与评分记录—快捷入口”的层次化布局实现。登录后，系统在 HttpSession 中保持登录态，个人中心从会话中获取用户标识，汇总展示用户的基础信息（用户名、邮箱等）与行为统计（点赞/收藏/加购/购买计数），并列出最近的评分记录与常用操作入口（如商品列表、关于我们、退出登录）。为保证体验一致性，页面采用卡片化信息区块与统计组件，结合轻量动画与统一配色，让数据可读、状态明确。

在数据流上，控制器负责从会话获取用户并调用服务层进行统计查询；服务层聚合 UserBehavior 与 ProductRating 的查询结果，按行为类型赋权或直接计数，返回个人中心视图需要的模型数据；持久层以 Mapper 接口+XML 执行高频统计与分页查询，避免在服务层手动拼装复杂 SQL。模板采用 Thymeleaf 进行绑定与条件渲染：登录态显示用户信息与统计卡片，非登录态给出“去登录”提示与引导；空数据情况下显示优雅的占位与操作建议，确保页面在数据不全时仍保持完整结构与可用性。

5.3.2 关键代码

控制器：PersonController.java

// 位置示例：src/main/java/com/nuist/controller/PersonController.java

@Controller

public class PersonController {

@Autowired

private UserCenterService userCenterService;

@GetMapping("/person")

public String person(Model model, HttpSession session) {

MyUser loginUser = (MyUser) session.getAttribute("loginUser");

if (loginUser == null) {

```

        model.addAttribute("error", "请先登录");
        return "login";
    }
    UserCenterView view = userCenterService.buildUserCenterView(loginUser.getId());
    model.addAttribute("user", loginUser);
    model.addAttribute("stats", view.getStats());
    model.addAttribute("recentRatings", view.getRecentRatings());
    return "person";
}
}

```

服务层：UserCenterService.java

// 位置示例：src/main/java/com/nuist/service/UserCenterService.java

@Service

public class UserCenterService {

@Autowired

private UserBehaviorMapper behaviorMapper;

@Autowired

private ProductRatingMapper ratingMapper;

public UserCenterView buildUserCenterView(**Integer** userId) {

 UserStats stats = **new** UserStats();

 stats.setClickCount(behaviorMapper.countByUserAndType(userId, "click"));

 stats.setFavoriteCount(behaviorMapper.countByUserAndType(userId, "collect"));

 stats.setCartCount(behaviorMapper.countByUserAndType(userId, "cart"));

 stats.setPayCount(behaviorMapper.countByUserAndType(userId, "pay"));

List<ProductRating> recent = ratingMapper.listRecentByUser(userId, 10);

 UserCenterView view = **new** UserCenterView();

 view.setStats(stats);

 view.setRecentRatings(recent);

return view;

}

}

模板：person.html（节选）

<!-- 位置示例：src/main/resources/templates/person.html -->

<!DOCTYPE html>

<html lang="zh-CN" xmlns:th="http://www.thymeleaf.org">

<head>

 <meta charset="UTF-8"><title>个人中心</title>

 <link rel="stylesheet" th:href="@{/css/style.css}">

```

</head>
<body>
  <header class="navbar">
    <nav class="nav-center">
      <a th:href="@{/}">首页</a>
      <a th:href="@{/product/list}">商品列表</a>
      <a th:href="@{/about}">关于我们</a>
      <span class="login-state" th:text="{user.username}">用户</span>
    </nav>
  </header>

  <main class="profile-content">
    <section class="profile-header">
      <h2>你好, <span th:text="{user.username}">用户名</span></h2>
      <p th:if="{user.email}" th:text="{user.email}">邮箱</p>
    </section>

    <section class="stats-grid">
      <div class="stat-card">
        <span class="stat-number" th:text="{stats.clickCount}">0</span>
        <span class="stat-label">点击</span>
      </div>
      <div class="stat-card">
        <span class="stat-number" th:text="{stats.favoriteCount}">0</span>
        <span class="stat-label">收藏</span>
      </div>
      <div class="stat-card">
        <span class="stat-number" th:text="{stats.cartCount}">0</span>
        <span class="stat-label">加购</span>
      </div>
      <div class="stat-card">
        <span class="stat-number" th:text="{stats.payCount}">0</span>
        <span class="stat-label">购买</span>
      </div>
    </section>

    <section class="rating-list">
      <h3>我的评分</h3>
      <div th:if="{#lists.isEmpty(recentRatings)}" class="empty-state">
        <i class="empty-icon">🐼</i><p>你还没有评分, 去看看商品吧~</p>
      </div>
      <div th:each="r : {recentRatings}" class="rating-item">
        <span th:text="'商品 ID: ' + {r.productId}"></span>
        <span th:text="'评分: ' + {r.score}"></span>
      </div>
    </section>
  </main>
</body>

```

```

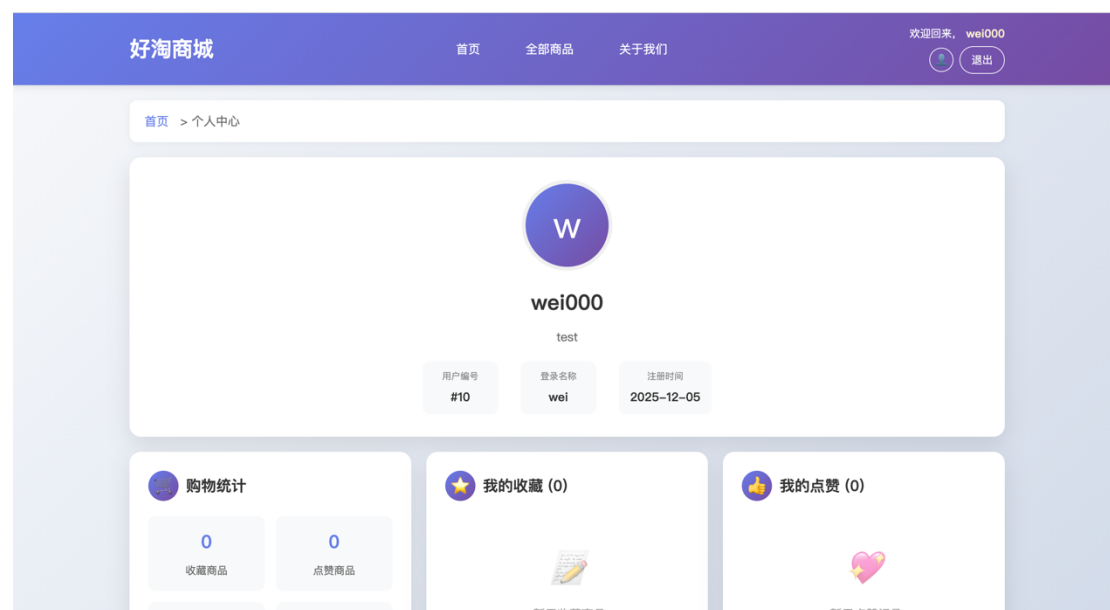
        <span th:text="\${#temporals.format(r.ratedAt, 'yyyy-MM-dd HH:mm')}"></span>
    </div>
</section>

<div class="action-buttons">
    <a class="btn-action btn-primary" th:href="@{/product/list}">去逛逛</a>
    <a class="btn-action btn-secondary" th:href="@{/recommend/guess}">猜你喜欢</a>
</div>
</main>

<footer class="footer">
    <div class="footer-bottom">
        <p>&copy; 2025 好淘商城. All Rights Reserved.</p>
        <p>南京信息工程大学 - 智能应用项目</p>
    </div>
</footer>
</body>
</html>

```

5.3.3 功能展示



5.4 信息修改

5.4.1 功能实现

“修改信息”功能围绕“登录校验—表单校验—幂等更新—回显反馈”设计实现。登录用户进入个人中心的资料编辑页，通过表单提交受控字段（如昵称/邮箱/手机号等），后端对输入进行基础校验（非空、长度、格式），并限制可修改范围与频次，防止越权更新与误操作。更新成功后，返回成功提示并回显最新数据；若校验失败或数据库写入异常，用邻近字段的错误提示引导用户修正。出于安全考虑，敏感字段如密码采用独立流程，仅在专用“修改密码”接口处理；

“修改信息”接口不允许修改账户主键或权限相关字段。设计上，控制器负责路由与会话检查，服务层封装校验与更新逻辑，持久层以 Mapper 执行局部字段更新（UPDATE ... SET ... WHERE id=?），模板用 Thymeleaf 完成数据绑定与错误消息显示。

在数据流与体验方面，编辑页以卡片化表单承载用户资料，输入项提供占位与格式提示；保存按钮在提交期间显示加载状态，避免重复提交；成功后以轻量提示（绿色反馈条）告知用户更新结果并刷新会话中的用户信息，保持导航栏与个人中心的统一显示。为保证幂等与一致性，更新逻辑以用户 ID 定位，不使用可变用户名作为唯一标识；若涉及邮箱/手机号的唯一性约束，则在服务层先行查重再写入，必要时提供“值未变更”的静默路径。整体实现以“可控、可测、可扩展”为原则，便于后续增加头像、个性签名、通知偏好等非核心字段。

5.4.2 关键代码

控制器：ProfileController.java

```
// 位置示例：src/main/java/com/nuist/controller/ProfileController.java
@Controller
@RequestMapping("/profile")
public class ProfileController {

    @Autowired
    private ProfileService profileService;

    @GetMapping("/edit")
    public String editPage(Model model, HttpSession session) {
        MyUser loginUser = (MyUser) session.getAttribute("loginUser");
        if (loginUser == null) return "redirect:/login";
        model.addAttribute("user", profileService.getById(loginUser.getId()));
        return "profile_edit";
    }

    @PostMapping("/edit")
    public String doEdit(@RequestParam(required=false) String nickname,
                        @RequestParam(required=false) String email,
                        @RequestParam(required=false) String phone,
```

```

        HttpSession session, Model model) {
    MyUser loginUser = (MyUser) session.getAttribute("loginUser");
    if (loginUser == null) return "redirect:/login";
    try {
        MyUser updated = profileService.updateBasicInfo(loginUser.getId(), nickname, email, phone);
        session.setAttribute("loginUser", updated); // 刷新会话用户
        model.addAttribute("user", updated);
        model.addAttribute("msg", "更新成功");
    } catch (IllegalArgumentException ex) {
        model.addAttribute("user", profileService.getById(loginUser.getId()));
        model.addAttribute("error", ex.getMessage());
    }
    return "profile_edit";
}
}

```

服务层: ProfileService.java

// 位置示例: src/main/java/com/nuist/service/ProfileService.java

```

@Service
public class ProfileService {

    @Autowired
    private MyUserMapper myUserMapper;

    public MyUser getById(Integer id) {
        return myUserMapper.findById(id);
    }

    public MyUser updateBasicInfo(Integer userId, String nickname, String email, String phone) {
        if (nickname != null && nickname.trim().length() > 32) {
            throw new IllegalArgumentException("昵称长度不可超过 32");
        }
        if (email != null && !email.trim().isEmpty() && !email.matches("^\\w.-]+@[\\w.-]+\\.\\[A-Za-z]{2,}$"))
        {
            throw new IllegalArgumentException("邮箱格式不正确");
        }
        if (phone != null && !phone.trim().isEmpty() && !phone.matches("^\\d{6,18}$")) {
            throw new IllegalArgumentException("手机号格式不正确");
        }
        // 唯一性示例: 邮箱查重 (排除本人)
        if (email != null && !email.trim().isEmpty() && myUserMapper.countByEmailExcludeId(email.trim(),
        userId) > 0) {
            throw new IllegalArgumentException("该邮箱已被使用");
        }
    }
}

```

```

        myUserMapper.updateProfile(userId,
            safeTrim(nickname),
            safeTrim(email),
            safeTrim(phone));
        return myUserMapper.findById(userId);
    }

    private String safeTrim(String s) { return s == null ? null : s.trim(); }
}

```

模板: profile_edit.html (节选)

```

<!-- 位置示例: src/main/resources/templates/profile_edit.html -->
<!DOCTYPE html>
<html lang="zh-CN" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8"><title>修改信息</title>
    <link rel="stylesheet" th:href="@{/css/style.css}">
</head>
<body>
    <div class="form-card">
        <h2>修改信息</h2>
        <div th:if="{msg}" class="alert-success" th:text="{msg}"></div>
        <div th:if="{error}" class="alert-error" th:text="{error}"></div>

        <form method="post" th:action="@{/profile/edit}">
            <label>昵称</label>
            <input name="nickname" th:value="{user.nickname}">
            <label>邮箱</label>
            <input name="email" th:value="{user.email}" placeholder="name@example.com">
            <label>手机号</label>
            <input name="phone" th:value="{user.phone}" placeholder="仅数字">
            <button class="btn action-btn" type="submit">保存</button>
        </form>

        <p class="hint">密码修改请前往“修改密码”页面处理。</p>
    </div>
</body>
</html>

```

5. 4. 3 功能展示

修改个人资料

完善您的个人信息，打造专属档案

首页

个人中心

修改资料

个人信息编辑

请认真填写以下信息，确保数据的准确性

Z

当前登录用户
zhangsan

ID

用户编号

1

注册时间

2025-12-02

最后更新

2025-12-02

ID

用户编号 (UID)

1

2025-12-02

2025-12-02

ID

用户编号 (UID)

1

系统自动分配，不可修改

登录名称

zhangsan

登录名称不可修改

登录密码

为了安全，请定期更换密码

昵称

昵称将作为您的显示名称，最多50个字符

个人介绍

介绍一下您自己吧...

个人介绍帮助其他用户更好地了解您，最多500个字符

保存修改

返回个人中心

32

1

2025-12-02

2025-12-02

1

用户编号 (UID)

系统自动分配, 不可修改

zhangsan

登录名称

登录名称不可修改

登录密码

为了安全, 请定期更换密码

张三张三张三

昵称

昵称将作为您的显示名称, 最多50个字符

我是张三test

个人介绍

个人介绍帮助其他用户更好地了解您, 最多500个字符

保存修改

返回个人中心

id	name	username	password	nickname	remark	email
# int	varchar(50)	varchar(50)	varchar(100)	varchar(50)	text	varchar(100)
1	zhangsan	zhangsan	123456	张三张三张三	我是张三 test	zhangsan@example.c
2	lisi	lisi	123456	(NULL)	(NULL)	lisi@example.com
3	wangwu	wangwu	123456	(NULL)	(NULL)	wangwu@example.co
4	admin	admin	123456	admin	admin 管理员	(NULL)
5	(NULL)	paul	123456	Paul	paul001	(NULL)
6	(NULL)	mike	123456	Mike	mike001	(NULL)
7	(NULL)	chris	123456	Chris1234	chris00000	(NULL)
8	(NULL)	xiaoli	123456	li12333	xiaoli@example.com	(NULL)
9	(NULL)	xiaohua	123456	hua909090	xiaohua@example.cor	(NULL)
10	wei	wei	123456	wei000	test	(NULL)

5.5 商品展示

5.5.1 功能实现

“全部商品与商品分类展示”模块以“列表页为中心、分类为引导、检索为补充”的设计组织信息与交互。列表页承担主浏览入口，采用卡片化栅格呈现商品的标题、海报、价格与简短描述，提供查看详情的快捷操作；在没有前端复杂状态管理的课程语境下，使用服务端渲染保证一致与稳定。分类展示为用户提供快速收敛路径：顶部或侧边栏的分类导航允许按分类过滤商品；在数据层面以 category 字段或分类表映射实现按分类查询与分页，确保高频读取路径简单可控。为兼顾体验与性能，列表页支持基础分页与空数据占位，移动端采用响应式布局保持良好可读性。

数据流上，控制器负责路由与参数绑定（如/product/list?category=电子产品&page=1），服务层封装列表与分类查询、分页计算与空状态处理；持久层以 Mapper 接口+XML 实现条件过滤与排序（如按上架时间或热门程度），并预留限流与索引优化策略。模板使用 Thymeleaf 进行数据绑定与条件渲染：没有分类参数时显示“全部商品”，存在分类参数时高亮当前分类并显示相应结果；分页区域根据总数与页码计算上一页、下一页链接。整体设计强调“简明、稳健、可扩展”：从基础的分类过滤与分页出发，顺畅演进到多条件筛选（价格区间、关键字）与排序（销量、评分）而不破坏现有结构。

5.5.2 关键代码

控制器：ProductController.java

// 位置示例：src/main/java/com/nuist/controller/ProductController.java

```
@Controller
@RequestMapping("/product")
public class ProductController {

    @Autowired
    private ProductService productService;

    @GetMapping("/list")
    public String list(@RequestParam(required=false) String category,
                      @RequestParam(defaultValue="1") int page,
                      @RequestParam(defaultValue="12") int size,
                      Model model) {
        PageResult<Product> result = productService.listByCategory(category, page, size);
        model.addAttribute("products", result.getItems());
        model.addAttribute("total", result.getTotal());
        model.addAttribute("page", page);
        model.addAttribute("size", size);
        model.addAttribute("category", category);
        // 可选：提供所有分类用于导航
        model.addAttribute("categories", productService.listAllCategories());
        return "product_list";
    }
}
```

服务层：ProductService.java

// 位置示例：src/main/java/com/nuist/service/ProductService.java

```
@Service
public class ProductService {

    @Autowired
    private ProductMapper productMapper;

    public PageResult<Product> listByCategory(String category, int page, int size) {
        int offset = (Math.max(page, 1) - 1) * size;
        List<Product> items = productMapper.selectByCategory(category, size, offset);
        int total = productMapper.countByCategory(category);
        return new PageResult<>(items, total);
    }

    public List<String> listAllCategories() {
        return productMapper.listDistinctCategories();
    }
}
```

```

    }

    // 示例：热门商品
    public List<Product> listTopN(int n) {
        return productMapper.selectTopN(n);
    }
}

```

Mapper XML（节选）：ProductMapper.xml

```

<!-- 位置示例：src/main/resources/mappers/ProductMapper.xml -->
<mapper namespace="com.nuist.mapper.ProductMapper">

    <select id="selectByCategory" resultType="com.nuist.entity.Product">
        SELECT id, title, poster, memo, price, category, like_count, favorite_count
        FROM product
        <where>
            <if test="category != null and category != "">
                category = #{category}
            </if>
        </where>
        ORDER BY created_at DESC
        LIMIT #{limit} OFFSET #{offset}
    </select>

    <select id="countByCategory" resultType="int">
        SELECT COUNT(1) FROM product
        <where>
            <if test="category != null and category != "">
                category = #{category}
            </if>
        </where>
    </select>

    <select id="listDistinctCategories" resultType="string">
        SELECT DISTINCT category FROM product WHERE category IS NOT NULL AND category <> ""
        ORDER BY category
    </select>

    <select id="selectTopN" resultType="com.nuist.entity.Product">
        SELECT id, title, poster, memo, price, category, like_count, favorite_count
        FROM product
        ORDER BY like_count DESC, favorite_count DESC, created_at DESC
        LIMIT #{n}
    </select>

```

```
</mapper>
```

模板: product_list.html (节选)

```
<!-- 位置示例: src/main/resources/templates/product_list.html -->
<!DOCTYPE html>
<html lang="zh-CN" xmlns:th="http://www.thymeleaf.org">
<head>
  <meta charset="UTF-8"><title>全部商品</title>
  <link rel="stylesheet" th:href="@{/css/style.css}">
</head>
<body>
  <header class="navbar">
    <nav class="nav-center">
      <a th:href="@{/}">首页</a>
      <span class="spacer"></span>
      <div class="category-nav">
        <a th:href="@{/product/list}" th:classappend="${category} == null ? 'active' : "">全部</a>
        <a th:each="c : ${categories}"
          th:href="@{/product/list?category=${c}}"
          th:text="${c}"
          th:classappend="${category} == ${c} ? 'active' : ""></a>
      </div>
    </nav>
  </header>

  <main class="product-grid">
    <div th:if="${#lists.isEmpty(products)}" class="empty-state">
      <div class="icon"></div>
      <h3>暂无商品</h3>
      <p>试试切换分类或稍后再来</p>
    </div>

    <div class="product-card" th:each="p : ${products}">
      
      <h3 th:text="${p.title}">商品标题</h3>
      <p class="price" th:text="${#numbers.formatDecimal(p.price,1,2)}">¥0.00</p>
      <p class="memo" th:text="${p.memo}">商品描述...</p>
      <div class="card-actions">
        <a class="btn action-btn" th:href="@{/product/detail/${p.id}}">详情</a>
      </div>
    </div>
  </main>
```

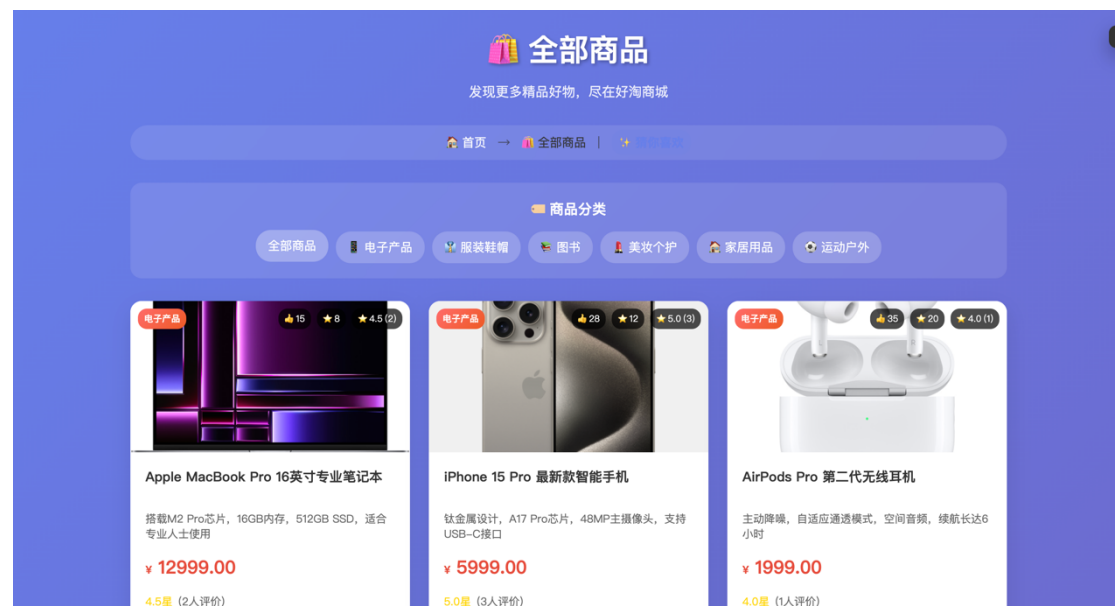
```

<section class="pagination" th:if="${total > size}">
    <a th:href="@{/product/list} + (${category} != null ? ('?category=' + ${category}) : '') + '&page=' +
    (${page}-1)}"
        th:classappend="${page} <= 1 ? 'disabled' : ''">上一页</a>
    <span th:text="${page}"></span>
    <a th:href="@{/product/list} + (${category} != null ? ('?category=' + ${category}) : '') + '&page=' +
    (${page}+1)}"
        th:classappend="${page} * ${size} >= ${total} ? 'disabled' : ''">下一页</a>
</section>

<footer class="footer">
    <div class="footer-bottom">
        <p>&copy; 2025 好淘商城. All Rights Reserved.</p>
        <p>南京信息工程大学 - 智能应用项目</p>
    </div>
</footer>
</body>
</html>

```

5.5.3 功能展示



商品分类

全部商品

电子产品

服装鞋帽

图书

美妆个护

家居用品

运动户外

电子产品

15

8

4.5 (2)

Apple MacBook Pro 16英寸专业笔记本

搭载M2 Pro芯片，16GB内存，512GB SSD，适合专业人士使用

¥ 12999.00

4.5星 (2人评价)

喜欢 收藏

加入购物车 立即购买

电子产品

28

12

5.0 (3)

iPhone 15 Pro 最新款智能手机

钛金属设计，A17 Pro芯片，48MP主摄像头，支持USB-C接口

¥ 5999.00

5.0星 (3人评价)

喜欢 收藏

加入购物车 立即购买

电子产品

35

20

4.0 (1)

AirPods Pro 第二代无线耳机

主动降噪，自适应通透模式，空间音频，续航长达6小时

¥ 1999.00

4.0星 (1人评价)

喜欢 收藏

加入购物车 立即购买

电子产品

22

14

3.0 (1)

Xiaomi 13 Ultra

电子产品

10

5

0.0 (0)

电子产品

42

25

5.0 (1)

商品分类

全部商品

电子产品

服装鞋帽

图书

美妆个护

家居用品

运动户外

服装鞋帽

50

28

0.0 (0)

Nike Air Max 270 时尚运动鞋

全掌气垫设计，透气网面材质，多色可选，适合日常运动和休闲穿着

¥ 599.00

暂无评分 (0人评价)

喜欢 收藏

加入购物车 立即购买

服装鞋帽

38

22

4.0 (1)

阿迪达斯运动套装三条纹经典

吸汗透气材质，经典三条纹设计，适合健身跑步和日常穿着

¥ 599.00

4.0星 (1人评价)

喜欢 收藏

加入购物车 立即购买

服装鞋帽

45

23

0.0 (0)

优衣库羽绒服轻薄保暖外套

90%白鸭绒填充，轻薄便携，防水面料，多种颜色可选，适合秋冬穿着

¥ 299.00

暂无评分 (0人评价)

喜欢 收藏

加入购物车 立即购买

服装鞋帽

32

18

4.0 (1)

运动鞋透气跑步鞋

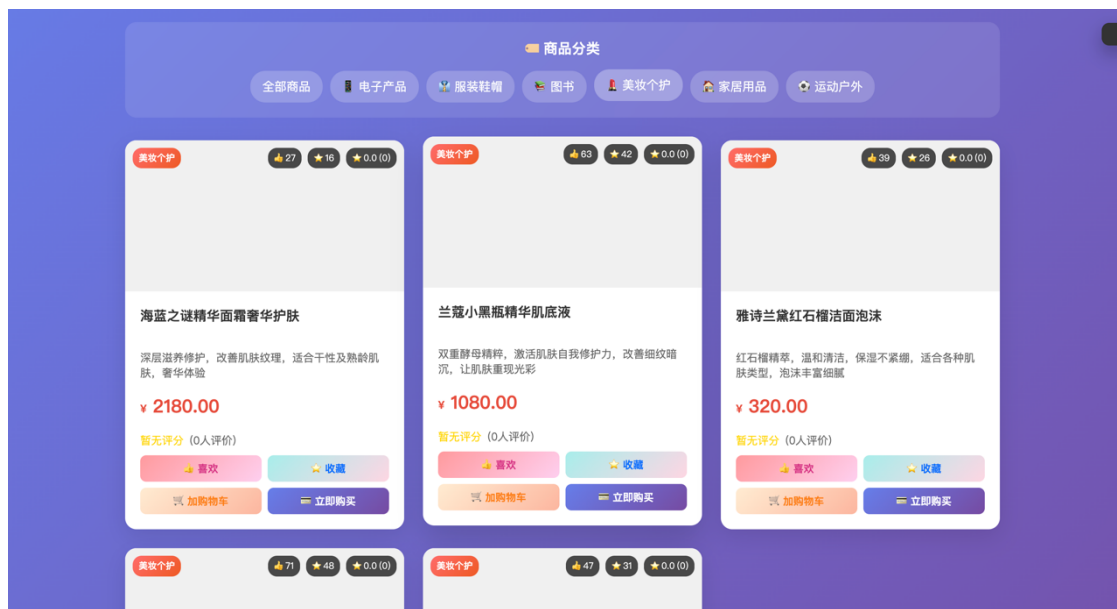
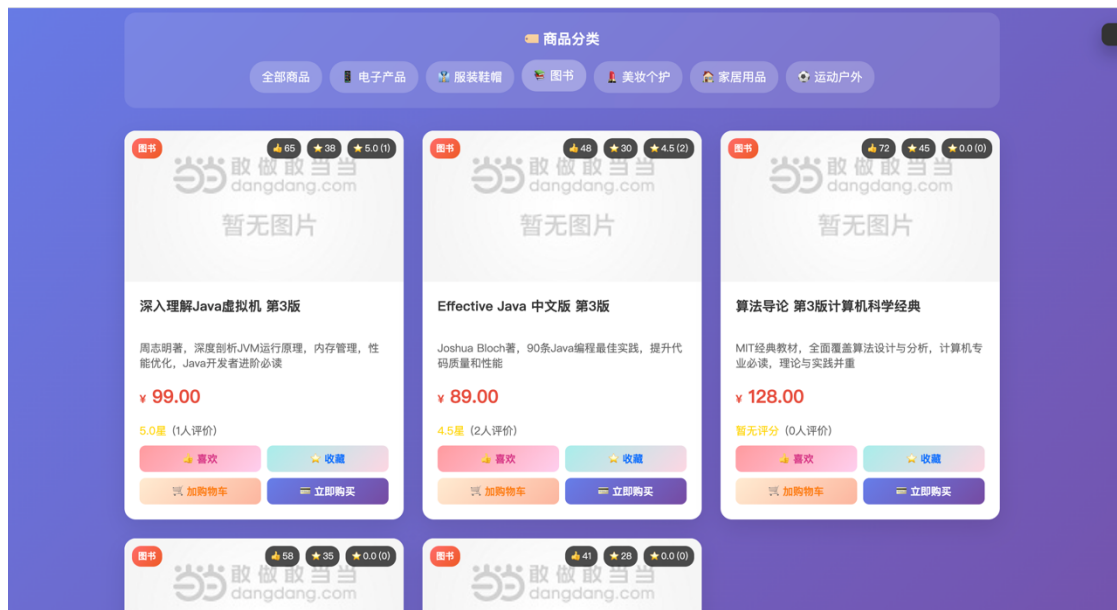
服装鞋帽

28

15

4.0 (1)

女士连衣裙时尚修身



id # int	title # varchar(200)	name # varchar(100)	price # decimal(10,2)	stock # int	description # text	category # varchar(50)	image_url # varchar(500)
1	Apple MacBook Pro 16英寸专业笔记本	MacBook Pro	12999.00	50	苹果笔记本电脑	电子产品	https://store.storeimages.cdn-apple.com/4982/as-images.apple.com/is/mbp16
2	iPhone 15 Pro 最新款智能手机	iPhone 15	5999.00	100	苹果最新款手机	电子产品	https://store.storeimages.cdn-apple.com/4982/as-images.apple.com/is/iphon
3	AirPods Pro 第二代无线耳机	AirPods Pro	1999.00	200	苹果无线耳机	电子产品	https://store.storeimages.cdn-apple.com/4982/as-images.apple.com/is/MQD
4	小米 13 Ultra 徕卡影像旗舰	Mi 13 Ultra	5999.00	80	徕卡影像旗舰手机	电子产品	/images/product4.png
5	戴尔 XPS 13 超极本商务笔记本	Dell XPS 13	9999.00	40	轻薄便携商务本	电子产品	/images/product5.png
6	索尼 WH-1000XM5 旗舰降噪耳机	Sony WH-1000XM5	2799.00	150	旗舰降噪耳机	电子产品	/images/product6.png
7	华为 MatePad Pro 专业创作平板	MatePad Pro	3999.00	60	专业创作平板	电子产品	https://consumer.huawei.com/content/dam/huawei-cbg-site/common/mkt/pd
8	罗技 MX Master 3S 高端办公鼠标	Logitech MX Master 3S	799.00	120	高端办公鼠标	电子产品	https://resource.logitech.com/vw_692_c_ipad_ar_4-3_q_auto_f_auto_dpr_1-0/d
9	Nike Air Max 270 时尚运动鞋	Nike 运动鞋	599.00	150	舒适运动鞋	服装鞋帽	https://static.nike.com/a/images/t_PDP_1728_v1/f_auto,q_auto,eco/9948685
10	阿迪达斯运动套装三条纹经典	Adidas Sports Set	599.00	180	舒适运动套装	服装鞋帽	https://assets.adidas.com/images/h_840,f_auto,q_auto,fl_lossy,c_fill,g_auto/f
11	深入理解Java虚拟机 第3版	Java虚拟机	99.00	300	程序员必读经典	图书	https://img3m7.ddimg.cn/83/35/29517847-1_u_2.jpg
12	Effective Java 中文版 第3版	Effective Java	89.00	250	Java编程最佳实践	图书	https://img3m2.ddimg.cn/56/23/23259681-1_u_9.jpg
13	海蓝之谜精华面霜奢华护肤	La Mer Creme	2180.00	80	奢华护肤品	美妆个护	https://www.cremedelamer.com/media/export/cms/products/pdp/la_mer_the
14	优衣库羽绒轻薄保暖外套	优衣库羽绒服	299.00	120	轻薄保暖羽绒服	服装鞋帽	https://image.uniqlo.com/UQ/ST3/AsianCommon/Images/goods/449458/item
15	李宁运动鞋透气跑步鞋	李宁跑步鞋	399.00	200	透气跑步鞋	服装鞋帽	https://img.lining.com/resources/mall/goods/2024/04/08/1612468873102645
16	ZARA女士连衣裙时尚修身	ZARA连衣裙	299.00	80	时尚连衣裙	服装鞋帽	https://static.zara.net/photos///2023/V/0/1/p/1165/144/800/2/w/563/1165144E
17	算法导论 第3版计算机科学经典	算法导论	128.00	180	计算机算法经典教材	图书	https://img3m0.ddimg.cn/24/14/9317290-1_u_15.jpg
18	Python编程从入门到实践 第2版	Python编程	89.00	220	Python入门经典教程	图书	https://img3m1.ddimg.cn/83/5/25197791-1_u_4.jpg
19	设计模式可复用面向对象软件的基础	设计模式	79.00	150	软件设计经典著作	图书	https://img3m7.ddimg.cn/56/23/23464752-1_u_2.jpg
20	兰蔻小黑瓶精华肌底液	兰蔻小黑瓶	1080.00	100	抗老精华肌底液	美妆个护	https://www.lancome.com.cn/dw/image/v2/BJQL_PRD/on/demandware.static
21	雅诗兰黛红石榴洁面泡沫	雅诗兰黛洁面	320.00	150	温和洁面泡沫	美妆个护	https://www.esteelauder.com/media/export/cms/products/558x768/el_sku_R
22	SK-II神仙水护肤精华水	SK-II神仙水	1590.00	80	明星护肤精华水	美妆个护	https://www.sk-ii.com/on/demandware.static/-/Sites-skii-master-catalog/def
23	资生堂红腰子精华液	资生堂红腰子	680.00	120	修护精华液	美妆个护	https://www.shiseido.com/dw/image/v2/BDHC_PRD/on/demandware.static/-/

5.6 基于商品的推荐算法

5.6.1 功能实现

本项目的商品智能推荐以协同过滤为核心，通过用户显式评分与隐式行为（点击、收藏、加购、购买）共同构建用户—物品关系矩阵。在算法层面分别实现用户相似与物品相似两条路径：用户相似以皮尔逊相关衡量用户偏好的一致性，从相似用户的评分与行为推断目标用户可能喜欢的物品；物品相似以余弦相似衡量商品之间的“被同一批用户偏好”的程度，再结合用户对某一商品的偏好推荐相似的其它商品。为提升效果与鲁棒性，行为信号进行赋权（如购买>加购>收藏>点击），与评分信号共同参与相似度与推荐分数计算；在冷启动或稀疏数据时回退到热门/规则推荐，保证页面不空白、体验不断流。

工程设计上，推荐模块在服务层完成数据聚合、相似度计算与候选排序，持久层以 Mapper+XML 高效读取评分与行为数据并可选写回推荐结果；控制器层只负责路由与参数解析，将推荐结果传给 Thymeleaf 模板以卡片形式展示。为避免耦合与便于升级，服务层对外暴露“按用户推荐”“看了又看/相似商品推荐”等方法，内部统一使用向量化与标准化的计算流程；数据读取路径按用户/商品维度建立索引与批量查询，尽可能减少 N+1 与无谓的对象转换。整体以“可解释、可运行、可扩展”为原则，方便后续演进到矩阵分解、学习排序（LTR）或轻量深度模型。

5.6.2 关键代码

服务层：RecommendService.java（节选-基于商品推荐算法）

// 位置示例：src/main/java/com/nuist/service/RecommendService.java

@Service

public class RecommendService {

 @Autowired

 private ProductRatingMapper ratingMapper;

 @Autowired

 private UserBehaviorMapper behaviorMapper;

 @Autowired

 private ProductMapper productMapper;

 // 基于用户相似（皮尔逊相关）

 public List<Product> recommendForUser(Integer userId, int limit) {

 Map<Integer, Double> targetVector = buildPreferenceVector(userId);

 Set<Integer> rated = ratingMapper.listProductIdsByUser(userId);

 // 计算与其它用户的相似度

 Map<Integer, Double> simByUser = new HashMap<>();

 for (Integer otherUserId : ratingMapper.listActiveUserIds()) {

 if (otherUserId.equals(userId)) continue;

 Map<Integer, Double> otherVector = buildPreferenceVector(otherUserId);

```

        double sim = pearson(targetVector, otherVector);
        if (sim > 0) simByUser.put(otherUserId, sim);
    }

    // 聚合相似用户的偏好，过滤已评分
    Map<Integer, Double> score = new HashMap<>();
    for (Map.Entry<Integer, Double> e : simByUser.entrySet()) {
        Integer uid = e.getKey();
        double w = e.getValue();
        Map<Integer, Double> vec = buildPreferenceVector(uid);
        for (Map.Entry<Integer, Double> item : vec.entrySet()) {
            if (rated.contains(item.getKey())) continue;
            score.merge(item.getKey(), w * item.getValue(), Double::sum);
        }
    }
    return pickTop(score, limit);
}

// 基于物品相似（余弦相似）
public List<Product> similarItems(Integer productId, int limit) {
    Map<Integer, Double> vecA = buildItemVector(productId);
    Map<Integer, Double> score = new HashMap<>();
    for (Integer otherPid : productMapper.listAllIds()) {
        if (otherPid.equals(productId)) continue;
        Map<Integer, Double> vecB = buildItemVector(otherPid);
        double sim = cosine(vecA, vecB);
        if (sim > 0) score.put(otherPid, sim);
    }
    return pickTop(score, limit);
}

// 用户偏好向量 = 评分(1..5) + 行为加权
private Map<Integer, Double> buildPreferenceVector(Integer userId) {
    Map<Integer, Double> vec = ratingMapper.userRatingVector(userId); // productId -> score
    Map<Integer, Integer> acts = behaviorMapper.userBehaviorCounts(userId); // productId:typeCount...
    acts.forEach((pid, cnt) -> vec.merge(pid, behaviorWeightSum(userId, pid), Double::sum));
    return vec;
}

// 物品向量 = 用户对该商品的评分/行为加权
private Map<Integer, Double> buildItemVector(Integer productId) {
    return ratingMapper.itemRatingVector(productId); // userId -> score; 若需要可合并行为权重
}

```

```

private double pearson(Map<Integer, Double> a, Map<Integer, Double> b) {
    Set<Integer> keys = new HashSet<>(a.keySet()); keys.retainAll(b.keySet());
    if (keys.size() < 2) return 0.0;
    double meanA = keys.stream().mapToDouble(k -> a.get(k)).average().orElse(0);
    double meanB = keys.stream().mapToDouble(k -> b.get(k)).average().orElse(0);
    double num = 0, da = 0, db = 0;
    for (Integer k : keys) {
        double xa = a.get(k) - meanA;
        double xb = b.get(k) - meanB;
        num += xa * xb; da += xa * xa; db += xb * xb;
    }
    double den = Math.sqrt(da) * Math.sqrt(db);
    return den == 0 ? 0 : num / den;
}

private double cosine(Map<Integer, Double> a, Map<Integer, Double> b) {
    Set<Integer> keys = new HashSet<>(a.keySet()); keys.retainAll(b.keySet());
    if (keys.isEmpty()) return 0.0;
    double dot = 0, na = 0, nb = 0;
    for (Integer k : a.keySet()) na += a.get(k) * a.get(k);
    for (Integer k : b.keySet()) nb += b.get(k) * b.get(k);
    for (Integer k : keys) dot += a.get(k) * b.get(k);
    double den = Math.sqrt(na) * Math.sqrt(nb);
    return den == 0 ? 0 : dot / den;
}

private double behaviorWeightSum(Integer userId, Integer productId) {
    // 简化：权重示例（可从配置读取）
    int click = behaviorMapper.countByUserProductType(userId, productId, "click");
    int collect = behaviorMapper.countByUserProductType(userId, productId, "collect");
    int cart = behaviorMapper.countByUserProductType(userId, productId, "cart");
    int pay = behaviorMapper.countByUserProductType(userId, productId, "pay");
    return click * 0.2 + collect * 0.6 + cart * 0.8 + pay * 1.5;
}

private List<Product> pickTop(Map<Integer, Double> score, int limit) {
    return score.entrySet().stream()
        .sorted((x,y) -> Double.compare(y.getValue(), x.getValue()))
        .limit(limit)
        .map(e -> productMapper.findById(e.getKey()))
        .filter(Objects::nonNull)
        .collect(Collectors.toList());
}
}

```

控制器：RecommendController.java（节选）

// 位置示例：src/main/java/com/nuist/controller/RecommendController.java

```
@Controller
@RequestMapping("/recommend")
public class RecommendController {

    @Autowired
    private RecommendService recommendService;

    @GetMapping("/guess")
    public String guessYouLike(HttpSession session, Model model) {
        MyUser loginUser = (MyUser) session.getAttribute("loginUser");
        if (loginUser == null) return "redirect:/login";

        List<Product> items = recommendService.recommendForUser(loginUser.getId(), 12);
        model.addAttribute("items", items);
        return "recommend/guess_you_like";
    }

    @GetMapping("/similar/{productId}")
    public String similar(@PathVariable Integer productId, Model model) {
        List<Product> items = recommendService.similarItems(productId, 10);
        model.addAttribute("items", items);
        return "recommend/guess_you_like";
    }
}
```

模板：guess_you_like.html（节选）

<!-- 位置示例：src/main/resources/templates/recommend/guess_you_like.html -->

```
<!DOCTYPE html>
<html lang="zh-CN" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8"><title>猜你喜欢</title>
    <link rel="stylesheet" th:href="@{/css/style.css}">
</head>
<body>
    <header class="navbar">
        <nav class="nav-center">
            <a th:href="@{/}">首页</a>
            <a th:href="@{/product/list}">全部商品</a>
            <a th:href="@{/person}">个人中心</a>
        </nav>
    </header>

    <main class="product-grid">
```

```

<div th:if="{#lists.isEmpty(items)}" class="empty-state">
  <div class="icon"></div>
  <h3>暂无推荐</h3>
  <p>先去浏览/评分一些商品吧</p>
</div>

<div class="product-card" th:each="p : ${items}">
  
  <h3 th:text="{p.title}">商品标题</h3>
  <p class="price" th:text="{#numbers.formatDecimal(p.price,1,2)}">¥0.00</p>
  <p class="memo" th:text="{p.memo}">商品描述...</p>
  <div class="card-actions">
    <a class="btn action-btn" th:href="@{/product/detail/{p.id}}">详情</a>
  </div>
</div>
</main>

<footer class="footer">
  <div class="footer-bottom">
    <p>&copy; 2025 好淘商城. All Rights Reserved.</p>
    <p>南京信息工程大学 - 智能应用项目</p>
  </div>
</footer>
</body>
</html>

```

5.6.3 功能展示



模拟 159 条用户行为用以实现算法：

名称	行	数据长度	引擎	创建日期	修改日期	排序规则	注释
myuser	10	16 KB	InnoDB	2025-12-02 14:29:18	2025-12-05 09:54:15	utf8mb4_0900_ai...	用户表
product	23	16 KB	InnoDB	2025-12-02 09:06:15	2025-12-03 09:44:46	utf8mb4_0900_ai...	商品表
product_rating	14	16 KB	InnoDB	2025-12-03 10:27:35	2025-12-03 10:28:11	utf8mb4_0900_ai...	商品评分表
recommend	5	16 KB	InnoDB	2025-12-02 16:41:22	2025-12-03 09:56:46	utf8mb4_0900_ai...	推荐结果表
user_behavior	159	16 KB	InnoDB	2025-12-02 16:41:22	2025-12-03 10:54:06	utf8mb4_0900_ai...	用户行为记录表

id	user_id	product_id	behavior_type	score	create_time	update_time
# int	# int	# int	enum varchar(20)	# int	timestamp	timestamp
106	6	4	click	1	2025-12-03 09:49:44	2025-12-03 09:49:44
107	6	4	cart	4	2025-12-03 09:49:44	2025-12-03 09:49:44
108	6	5	click	1	2025-12-03 09:49:44	2025-12-03 09:49:44
109	6	5	pay	5	2025-12-03 09:49:44	2025-12-03 09:49:44
110	6	6	click	1	2025-12-03 09:49:44	2025-12-03 09:49:44
111	6	7	click	1	2025-12-03 09:49:44	2025-12-03 09:49:44
112	6	7	fav	3	2025-12-03 09:49:44	2025-12-03 09:49:44
113	6	8	click	1	2025-12-03 09:49:44	2025-12-03 09:49:44
114	6	8	cart	4	2025-12-03 09:49:44	2025-12-03 09:49:44
115	6	14	click	1	2025-12-03 09:49:44	2025-12-03 09:49:44

智能推荐

首页商品猜你喜欢mike退出

猜你喜欢

基于商品的协同过滤推荐

基于智能算法为您精选商品推荐

基于用户推荐基于商品推荐

刷新推荐我的统计

敢做敢当 dangdang.com

暂无图片

Effective Java

Java编程最佳实践

基于商品协同过滤算法推荐

李宁跑步鞋

透气跑步鞋

李宁跑步鞋

透气跑步鞋

基于商品协同过滤算法推荐

ZARA连衣裙

时尚连衣裙

ZARA连衣裙

时尚连衣裙

基于商品协同过滤算法推荐

智能推荐

首页商品猜你喜欢paul退出

猜你喜欢

基于用户推荐基于商品推荐

刷新推荐我的统计

Dell XPS 13

轻薄便携商务本

基于商品协同过滤算法推荐

¥9999.00

匹配度: 16.51

喜欢收藏加入购物车购买

优衣库羽绒服

轻薄保暖羽绒服

基于商品协同过滤算法推荐

¥299.00

匹配度: 11.80

喜欢收藏加入购物车购买

MatePad Pro

专业创作平板

基于商品协同过滤算法推荐

¥3999.00

匹配度: 11.50

喜欢收藏加入购物车购买

5.7 基于用户的推荐算法

5.7.1 功能实现

基于用户的推荐以“用户—用户协同过滤”为核心，通过衡量用户之间的偏好相似度（皮尔逊相关），从与目标用户相似的群体中汇聚他们喜欢但目标用户尚未评分/交互的商品，生成候选集合并排序。信号来源同时包含显式评分与隐式行为（点击、收藏、加购、购买），行为按权重折算进偏好向量，使相似度兼顾“愿望”与“行动”。工程上，推荐服务负责构建用户偏好向量、计算相似度、聚合相似用户的偏好并过滤已看/已评商品；持久层提供按用户取评分、行为计数与已交互商品 ID 集合的高效查询；控制器只做路由与参数解析，把推荐结果交给模板渲染。为保证鲁棒性与体验连续性，在冷启动或稀疏数据下回退到热门/规则推荐。

5.7.2 关键代码

服务层：RecommendService.java（节选：用户协同过滤）

```
@Service
public class RecommendService {

    @Autowired
    private ProductRatingMapper ratingMapper;

    @Autowired
    private UserBehaviorMapper behaviorMapper;

    @Autowired
    private ProductMapper productMapper;

    // 基于用户的协同过滤推荐
    public List<Product> recommendForUser(Integer userId, int limit) {
        Map<Integer, Double> target = buildPreferenceVector(userId);
        Set<Integer> already = unionInteractedProductIds(userId);

        // 计算与其他用户的相似度（皮尔逊）
        Map<Integer, Double> sim = new HashMap<>();
        for (Integer other : ratingMapper.listActiveUserIds()) {
            if (other.equals(userId)) continue;
            Map<Integer, Double> vec = buildPreferenceVector(other);
            double s = pearson(target, vec);
            if (s > 0) sim.put(other, s);
        }

        // 汇聚相似用户的偏好，过滤已交互
        Map<Integer, Double> score = new HashMap<>();
        for (Map.Entry<Integer, Double> e : sim.entrySet()) {
            Integer u = e.getKey(); double w = e.getValue();
```

```

        Map<Integer, Double> vec = buildPreferenceVector(u);
        for (Map.Entry<Integer, Double> item : vec.entrySet()) {
            Integer pid = item.getKey();
            if (already.contains(pid)) continue;
            score.merge(pid, w * item.getValue(), Double::sum);
        }
    }
    return pickTop(score, limit);
}

// 偏好向量 = 用户评分 + 行为权重折算
private Map<Integer, Double> buildPreferenceVector(Integer userId) {
    Map<Integer, Double> v = ratingMapper.userRatingVector(userId); // productId -> score
    Map<Integer, Map<String, Integer>> counts =
behaviorMapper.userBehaviorCountsByProduct(userId);
    counts.forEach((pid, typeCnt) -> v.merge(pid, toWeight(typeCnt), Double::sum));
    return v;
}

private double toWeight(Map<String, Integer> c) {
    int click = c.getDefault("click", 0);
    int collect = c.getDefault("collect", 0);
    int cart = c.getDefault("cart", 0);
    int pay = c.getDefault("pay", 0);
    return click * 0.2 + collect * 0.6 + cart * 0.8 + pay * 1.5;
}

private Set<Integer> unionInteractedProductIds(Integer userId) {
    Set<Integer> s = new HashSet<>(ratingMapper.listProductIdsByUser(userId));
    s.addAll(behaviorMapper.listProductIdsByUser(userId));
    return s;
}

private double pearson(Map<Integer, Double> a, Map<Integer, Double> b) {
    Set<Integer> keys = new HashSet<>(a.keySet()); keys.retainAll(b.keySet());
    if (keys.size() < 2) return 0.0;
    double ma = keys.stream().mapToDouble(k -> a.get(k)).average().orElse(0);
    double mb = keys.stream().mapToDouble(k -> b.get(k)).average().orElse(0);
    double num = 0, da = 0, db = 0;
    for (Integer k : keys) {
        double xa = a.get(k) - ma, xb = b.get(k) - mb;
        num += xa * xb; da += xa * xa; db += xb * xb;
    }
    double den = Math.sqrt(da) * Math.sqrt(db);

```

```

        return den == 0 ? 0 : num / den;
    }

    private List<Product> pickTop(Map<Integer, Double> score, int limit) {
        return score.entrySet().stream()
            .sorted((x,y) -> Double.compare(y.getValue(), x.getValue()))
            .limit(limit)
            .map(e -> productMapper.findById(e.getKey()))
            .filter(Objects::nonNull)
            .collect(Collectors.toList());
    }
}

```

控制器：RecommendController.java（用户推荐入口）

```

@Controller
@RequestMapping("/recommend")
public class RecommendController {

    @Autowired
    private RecommendService recommendService;

    @GetMapping("/user")
    public String byUser(HttpSession session, Model model) {
        MyUser loginUser = (MyUser) session.getAttribute("loginUser");
        if (loginUser == null) return "redirect:/login";
        List<Product> items = recommendService.recommendForUser(loginUser.getId(), 12);
        model.addAttribute("items", items);
        return "recommend/guess_you_like";
    }
}

```

5.7.3 功能展示



模拟 159 条用户行为用以实现算法：

名称	行	数据长度	引擎	创建日期	修改日期	排序规则	注释
 myuser	10	16 KB	InnoDB	2025-12-02 14:29:18	2025-12-05 09:54:15	utf8mb4_0900_ai...	用户表
 product	23	16 KB	InnoDB	2025-12-02 09:06:15	2025-12-03 09:44:46	utf8mb4_0900_ai...	商品表
 product_rating	14	16 KB	InnoDB	2025-12-03 10:27:35	2025-12-03 10:28:11	utf8mb4_0900_ai...	商品评分表
 recommend	5	16 KB	InnoDB	2025-12-02 16:41:22	2025-12-03 09:56:46	utf8mb4_0900_ai...	推荐结果表
 user_behavior	159	16 KB	InnoDB	2025-12-02 16:41:22	2025-12-03 10:54:06	utf8mb4_0900_ai...	用户行为记录表

id	user_id	product_id	behavior_type	score	create_time	update_time
# int	# int	# int	varchar(20)	# int	timestamp	timestamp
1	1	2	click	1	2025-12-02 16:55:07	2025-12-02 16:55:07
2	1	2	fav	3	2025-12-02 16:55:07	2025-12-02 16:55:07
3	1	2	click	1	2025-12-02 16:55:07	2025-12-02 16:55:07
4	1	3	collect	2	2025-12-02 16:55:10	2025-12-02 16:55:10
5	1	3	click	1	2025-12-02 16:55:10	2025-12-02 16:55:10
6	1	3	click	1	2025-12-02 16:55:10	2025-12-02 16:55:10

智能推荐

首页商品猜你喜欢zhangsan退出

猜你喜欢

基于用户的协同过滤推荐

基于智能算法为您精选商品推荐

基于用户推荐基于商品推荐

刷新推荐我的统计



MacBook Pro

苹果笔记本电脑

基于协同过滤算法，相似用户也喜欢

智能推荐

首页商品猜你喜欢mike退出

猜你喜欢

基于用户的协同过滤推荐

基于智能算法为您精选商品推荐

基于用户推荐基于商品推荐

刷新推荐我的统计



暂无图片

Java虚拟机

程序员必读经典

基于协同过滤算法，相似用户也喜欢

每次使用推荐算法结果均已记录在数据库：

id # int	user_id # int	product_id # int	score # double	reason # varchar(200)	create_time timestamp	update_time timestamp
1	2	3	29	基于协同过滤算法，相似用户也喜欢	2025-12-03 08:32:	2025-12-03 08:32:
5	6	12	2.72719446866	基于商品协同过滤算法推荐	2025-12-03 09:56:	2025-12-03 09:56:
6	6	15	2.38629516005	基于商品协同过滤算法推荐	2025-12-03 09:56:	2025-12-03 09:56:
7	6	16	1.95237412036	基于商品协同过滤算法推荐	2025-12-03 09:56:	2025-12-03 09:56:
10	1	1	12	基于协同过滤算法，相似用户也喜欢	2025-12-05 11:25:	2025-12-05 11:25:
13	6	11	9	基于协同过滤算法，相似用户也喜欢	2025-12-05 11:26:	2025-12-05 11:26:
15	6	17	11	热门商品推荐	2025-12-05 11:26:	2025-12-05 11:26:
16	5	5	16.5103453626	基于商品协同过滤算法推荐	2025-12-05 11:28:	2025-12-05 11:28:
17	5	14	11.8045021729	基于商品协同过滤算法推荐	2025-12-05 11:28:	2025-12-05 11:28:
18	5	7	11.4957528386	基于商品协同过滤算法推荐	2025-12-05 11:28:	2025-12-05 11:28:
19	5	8	9.9828568667	基于商品协同过滤算法推荐	2025-12-05 11:28:	2025-12-05 11:28:
20	5	2	7.22104452785	基于商品协同过滤算法推荐	2025-12-05 11:28:	2025-12-05 11:28:
21	5	4	4.8639829980	基于商品协同过滤算法推荐	2025-12-05 11:28:	2025-12-05 11:28:

5.8 自定义功能：评价系统

5.8.1 功能实现

评价系统以“显式评分为主、行为信号为辅”的策略实现，核心目标是将用户对商品的主观偏好量化为可计算、可展示、可迭代的结构化数据。评分采用 1 - 5 星的离散刻度，强调简单直观与广泛适用；评价提交遵循“登录校验—表单校验—幂等写入”的流程，避免重复评分与异常覆盖。在数据一致性与可解释性上，以“用户 ID+商品 ID”的唯一约束保证每个用户对同一商品仅有一条评分记录；允许更新评分以反映偏好变化，并保留评分时间用于推荐与排序的时效性参考。前端采用卡片化与轻量提示反馈，确保评分操作感知明确、过程顺畅；空数据与错误状态以占位与信息引导处理。

在工程落地上，控制器负责路由与参数解析，将评分的创建/更新与查询请求转交给服务层；服务层封装评分校验、查重与写入逻辑，并在必要时触发统计更新（如平均分、评分数）以便页面展示；持久层以 Mapper+XML 实现高频的查询与幂等更新，避免复杂业务逻辑落入 SQL 层。模板通过 Thymeleaf 完成表单与列表的绑定，支持登录态下的评分提交与历史记录展示。评价数据在推荐模块中与行为信号共同构成用户—物品关系矩阵，既提升协同过滤的效果，也让“猜你喜欢”与“相似商品”的解释更具说服力。

5.8.2 关键代码

控制器：ProductRatingController.java

// 位置示例：src/main/java/com/nuist/controller/ProductRatingController.java

```

@Controller
@RequestMapping("/rating")
public class ProductRatingController {

    @Autowired
    private ProductRatingService ratingService;

```

```

@PostMapping("/submit")
public String submit(@RequestParam Integer productId,
                    @RequestParam Integer score,
                    @RequestParam(required=false) String comment,
                    HttpSession session, Model model) {
    MyUser loginUser = (MyUser) session.getAttribute("loginUser");
    if (loginUser == null) return "redirect:/login";
    try {
        ratingService.upsertRating(loginUser.getId(), productId, score, comment);
        model.addAttribute("msg", "评分提交成功");
    } catch (IllegalArgumentException ex) {
        model.addAttribute("error", ex.getMessage());
    }
    return "redirect:/product/detail/" + productId;
}

@GetMapping("/my-ratings")
public String myRatings(HttpSession session, Model model) {
    MyUser loginUser = (MyUser) session.getAttribute("loginUser");
    if (loginUser == null) return "redirect:/login";
    model.addAttribute("ratings", ratingService.listByUser(loginUser.getId(), 50));
    return "rating/my-ratings";
}
}

```

服务层：ProductRatingService.java

// 位置示例：src/main/java/com/nuist/service/ProductRatingService.java

```

@Service
public class ProductRatingService {

    @Autowired
    private ProductRatingMapper ratingMapper;

    public void upsertRating(Integer userId, Integer productId, Integer score, String comment) {
        if (score == null || score < 1 || score > 5) {
            throw new IllegalArgumentException("评分需为 1~5");
        }
        ProductRating existing = ratingMapper.findByUserProduct(userId, productId);
        if (existing == null) {
            ProductRating pr = new ProductRating();
            pr.setUserId(userId);
            pr.setProductId(productId);
            pr.setScore(score);
            pr.setComment(safe(comment));
        }
    }
}

```

```

        pr.setRatedAt(LocalDateTime.now());
        ratingMapper.insert(pr);
    } else {
        ratingMapper.updateScoreAndComment(existing.getId(), score, safe(comment),
LocalDateTime.now());
    }
}

public List<ProductRating> listByUser(Integer userId, int limit) {
    return ratingMapper.listRecentByUser(userId, limit);
}

private String safe(String s) { return s == null ? null : s.trim(); }
}

```

Mapper XML（节选）：ProductRatingMapper.xml

```

<!-- 位置示例：src/main/resources/mappers/ProductRatingMapper.xml -->
<mapper namespace="com.nuist.mapper.ProductRatingMapper">

    <select id="findByUserProduct" resultType="com.nuist.entity.ProductRating">
        SELECT id, user_id, product_id, score, comment, rated_at
        FROM product_rating
        WHERE user_id = #{userId} AND product_id = #{productId}
        LIMIT 1
    </select>

    <insert id="insert" parameterType="com.nuist.entity.ProductRating">
        INSERT INTO product_rating(user_id, product_id, score, comment, rated_at)
        VALUES(#{userId}, #{productId}, #{score}, #{comment}, #{ratedAt})
    </insert>

    <update id="updateScoreAndComment">
        UPDATE product_rating
        SET score = #{score},
            comment = #{comment},
            rated_at = #{ratedAt}
        WHERE id = #{id}
    </update>

    <select id="listRecentByUser" resultType="com.nuist.entity.ProductRating">
        SELECT id, user_id, product_id, score, comment, rated_at
        FROM product_rating
        WHERE user_id = #{userId}
        ORDER BY rated_at DESC
    </select>
</mapper>

```

```

LIMIT #{limit}
</select>

</mapper>

```

评分模板：my-ratings.html（节选）

```

<!-- 位置示例：src/main/resources/templates/rating/my-ratings.html -->
<!DOCTYPE html>
<html lang="zh-CN" xmlns:th="http://www.thymeleaf.org">
<head>
  <meta charset="UTF-8"><title>我的评分</title>
  <link rel="stylesheet" th:href="@{/css/style.css}">
</head>
<body>
  <header class="navbar">
    <nav class="nav-center">
      <a th:href="@{/}">首页</a>
      <a th:href="@{/product/list}">全部商品</a>
      <a th:href="@{/person}">个人中心</a>
    </nav>
  </header>

  <main class="rating-content">
    <h2>我的评分</h2>

    <div th:if="${#lists.isEmpty(ratings)}" class="empty-state">
      <div class="icon">★</div>
      <h3>还没有评分记录</h3>
      <p>去逛逛商品并留下你的评价吧</p>
    </div>

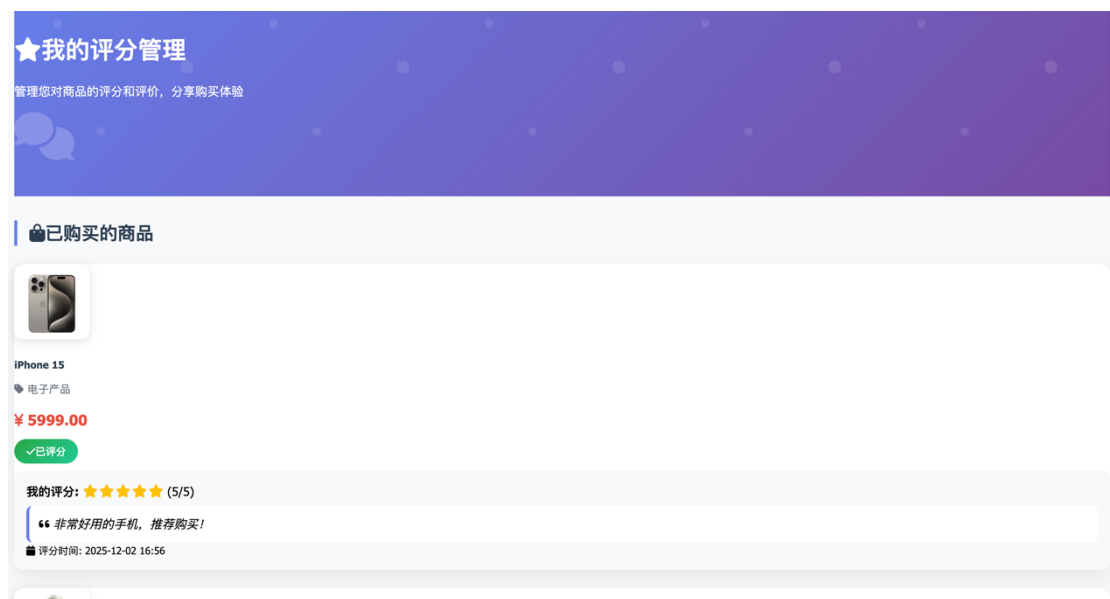
    <div class="rating-item" th:each="r : ${ratings}">
      <span th:text="'商品 ID: ' + ${r.productId}"></span>
      <span th:text="'评分: ' + ${r.score}"></span>
      <span th:text="'${#temporals.format(r.ratedAt, 'yyyy-MM-dd HH:mm')}"></span>
      <span th:if="${r.comment}" th:text="${r.comment}"></span>
    </div>
  </main>

  <footer class="footer">
    <div class="footer-bottom">
      <p>© 2025 好淘商城. All Rights Reserved.</p>
      <p>南京信息工程大学 - 智能应用项目</p>
    </div>
  </footer>

```

```
</footer>
</body>
</html>
```

5.8.3 功能展示



电子产品

¥ 2799.00

✓已评分

我的评分: ★★★★★ (5/5)

“音质很棒, 值得推荐”

评分时间: 2025-12-02 16:58

评分统计

3

总评分数

4.7

平均评分

在商品界面可看见平均分：

全部商品

电子产品

服装鞋帽

图书

美妆个护

家居用品

运动户外

电子产品

Apple MacBook Pro 16英寸专业笔记本

搭载M2 Pro芯片, 16GB内存, 512GB SSD, 适合专业人士使用

¥ 12999.00

4.5星 (2人评价)

喜欢 收藏

加入购物车 立即购买

电子产品

iPhone 15 Pro 最新款智能手机

钛金属设计, A17 Pro芯片, 48MP主摄像头, 支持USB-C接口

¥ 5999.00

5.0星 (3人评价)

喜欢 收藏

加入购物车 立即购买

电子产品

AirPods Pro 第二代无线耳机

主动降噪, 自适应通透模式, 空间音频, 续航长达6小时

¥ 1999.00

4.0星 (1人评价)

喜欢 收藏

加入购物车 立即购买

电子产品

Xiaomi 13 Ultra

16GB + 1TB

白色1TB版本 5月6日10点开售

2.2星 (14人评价)

喜欢 收藏

加入购物车 立即购买

电子产品

1.0星 (0人评价)

喜欢 收藏

加入购物车 立即购买

电子产品

5.0星 (1人评价)

喜欢 收藏

加入购物车 立即购买

所有评价均在数据库显示：

id # int	user_id # int	product_id # int	rating # int	comment # text	create_time timestamp	update_time timestamp	
1	1	1	2	5 非常好用的手机, 推荐购买!	2025-12-02 16:56:00	2025-12-02 16:56:00	
2	1		3	4 书很不错, 内容很实用	2025-12-02 16:57:00	2025-12-02 16:57:00	
3	1		6	5 音质很棒, 值得推荐	2025-12-02 16:58:00	2025-12-02 16:58:00	
4	2	1	4	4 笔记本性能不错, 外观也很好看	2025-12-03 08:28:00	2025-12-03 08:28:00	
5	2	2	2	5 手机很流畅, 拍照效果很好	2025-12-03 08:29:00	2025-12-03 08:29:00	
6	2	4	3	3 平板一般般, 性价比不高	2025-12-03 08:30:00	2025-12-03 08:30:00	
7	5	12	5	5 Effective Java这本书真的很经典!	2025-12-03 09:50:00	2025-12-03 09:50:00	
8	5	15	4	4 李宁的运动鞋质量不错, 穿着舒适	2025-12-03 09:51:00	2025-12-03 09:51:00	
9	5	16	4	4 ZARA的连衣裙款式很好看	2025-12-03 09:52:00	2025-12-03 09:52:00	
10	6	1	5	5 联想笔记本性能很强, 推荐!	2025-12-03 09:53:00	2025-12-03 09:53:00	
11	6	2	5	5 华为手机信号很好, 系统流畅	2025-12-03 09:54:00	2025-12-03 09:54:00	
12	7	10	4	4 雅诗兰黛的面霜保湿效果很好	2025-12-03 09:55:00	2025-12-03 09:55:00	
13	8	11	5	5 香奈儿香水味道很持久, 很喜欢	2025-12-03 09:56:00	2025-12-03 09:56:00	
14	8	12	4	4 Java书籍内容很全面, 适合进阶学习	2025-12-03 09:57:00	2025-12-03 09:57:00	

六、项目总结

本项目以“好淘商城”为载体，完成了一个从后端到前端、从数据到算法的完整智能应用闭环。系统围绕用户、商品、行为、评分与推荐五大数据域设计表结构与访问路径，前端以 HTML+Thymeleaf 渲染、CSS 响应式布局承载核心页面，后端以 Spring Boot、Spring MVC、MyBatis 与 MySQL 实现稳定的服务与数据管道，满足课程实践对“可运行、可展示、可验证”的基本要求。项目运行与部署流程清晰、脚本初始化完备，在本地环境中能快速编译、启动与验证，体现出良好的工程可操作性。

在智能推荐方面，项目以协同过滤为主线，兼顾用户相似与物品相似两类路径，综合显式评分与隐式行为构建用户—物品关系矩阵，并通过合理的行为赋权提升推荐质量与鲁棒性。推荐结果以“猜你喜欢”与“相似商品”进行可视化呈现，空数据与冷启动提供降级策略，确保体验连续。这一实现既延续业界经典做法、具备可解释性，也为后续引入矩阵分解、学习排序或轻量深度模型预留了演进空间，具有可扩展的研究与工程价值。

从教学与工程素养角度看，项目以分层架构组织控制器、服务、持久与模板，各层职责清晰、接口契约稳定，既便于理解与团队协作，也便于单元替换与增量迭代。数据侧坚持幂等更新与唯一约束，保证评分与行为在高频操作下的数据一致性；运行侧采用嵌入式容器与统一构建命令，实现“一键启动”，提升开发与演示效率。整体实现体现了“技术服务业务”的教学理念，让学生在真实场景中把握 Web 工程与智能算法的结合点。

综上，本项目完成了电商场景下智能推荐系统的核心功能与展示落地，形成了从需求分析、系统设计、数据建模、算法实现到页面呈现的完整链路。