廣東工業大學

专业综合拓展报告

基于 Django 的大学生就业信息共享平台

学　　院　先进制造学院

专　　业　人工智能

年级班级　22 级 3 班

学　　号　3122009223

学生姓名　罗展鹏

指导教师　林庆发

2025 年 12 月

成绩评定栏（教师填写）				
平时+答辩（占 40%）	项目设计报告（60%）	总评百分制	总评等级	签名

目录

摘　要

针对当前大学生就业信息分散、企业招聘资质难核验、求职与招聘信息匹配效率低等问题，本文设计了一款基于 Django 框架的大学生就业信息资源共享平台，采用前后端分离架构，结合 Python 编程语言、数据库管理技术及软件工程方法论，构建高效、安全的就业信息交互体系。

平台围绕“学生-企业-系统管理”三类用户需求划分核心功能模块：学生模块支持用户登录注册、简历上传编辑、就业经验分享、招聘信息精准筛选及求职实时提醒，解决学生获取就业资源碎片化的痛点；企业模块聚焦资质审核与基础招聘功能，实现企业合规入驻与求职者简历定向查看，保障招聘信息真实性；系统管理模块通过用户账号与权限管控、招聘及发帖内容合规审核、公告发布管理，维护平台信息安全与运营秩序。

开发过程中，首先完成需求分析与整体方案设计，明确模块划分原则与数据交互逻辑；其次基于 Django REST Framework 构建后端 API 接口，设计 MySQL 数据库存储用户、简历、招聘等核心数据，结合 Vue.js 完成前端页面开发；最后通过功能模块测试、组装测试及需求验证，确保平台各功能稳定运行——测试结果表明，平台可实现就业信息高效共享与精准匹配，满足学生求职、企业招聘及平台管理的多元化需求，为高校就业服务数字化提供实践参考。

本文的创新点在于将“信息共享”与“合规审核”深度结合，在提升就业信息流通效率的同时，通过多层级审核机制保障信息质量；技术层面采用前后端分离架构，降低系统耦合度，便于后续功能扩展与维护。

关键词：Django 框架；就业信息平台；前后端分离；数据库设计；信息共享

Abstract

To address the current issues of fragmented job information for college students, difficulties in verifying corporate recruitment qualifications, and low efficiency in matching job seekers with available positions, this paper designs and implements a Django-based platform for sharing employment information resources. Adopting a front-end and back-end separation architecture, the platform integrates Python programming, database management technologies, and software engineering methodologies to establish an efficient and secure employment information exchange system.

The platform organizes its core modules around three user categories: Students, Enterprises, and System Management. The Student module provides essential features including account registration, resume editing, job experience sharing, targeted job search filters, and real-time alerts, effectively addressing students' fragmented access to employment resources. The Enterprise module focuses on qualification verification and basic recruitment functions, enabling compliant company registration and targeted resume screening for job seekers, while ensuring the authenticity of job postings. The System Management module safeguards platform security and operational order through account access control, content compliance review for recruitment posts, and official announcement management.

During the development process, we first conducted requirement analysis and overall solution design, establishing clear principles for module division and data interaction logic. Next, we built backend API interfaces using Django REST Framework, designed MySQL databases to store core data including users, resumes, and recruitment information, and integrated Vue.js for front-end page development. Finally, through functional module testing, integration testing, and requirement verification, we ensured stable operation of all platform features. Test results demonstrate that the platform enables efficient sharing and precise matching of employment information, meeting diverse needs for student job hunting, corporate recruitment, and platform management. This provides practical references for digitalizing university employment services.

The innovation of this paper lies in the deep combination of "information sharing" and "compliance audit", which not only improves the efficiency of employment information circulation, but also ensures the quality of information through multi-level audit mechanism. On the technical level, the front-end and back-end separation architecture is adopted to reduce the coupling degree of the system, which is convenient for subsequent function expansion and maintenance.

Keywords:Django framework; job information platform; front-end and back-end separation; database design; information sharing

1. 引言

1.1. 编写目的

本详细设计说明书是针对专业综合设计课程作业编写，目的是在概要设计基础上进一步明确系统结构，详细阐述核心子系统及各模块的设计细节、类图构建、接口定义等内容，为后续系统开发、测试与维护提供标准化技术文档。本说明书的预期读者为项目开发团队成员、课程指导教师及后续参与系统扩展与维护的相关人员。

1.2. 立项背景

在数字化经济深度渗透的时代，高校毕业生就业是连接教育成果与社会需求的核心纽带，但其就业市场仍存在多重结构性矛盾。学生端面临就业信息碎片化、经验传递壁垒等问题，需耗费大量时间筛选无效信息，隐性求职经验难以规模化共享；企业端受资质审核缺失与匹配低效困扰，招聘信息真实性难保障，简历筛选成本高；高校端数字化服务能力不足，缺乏系统化工具整合信息、审核资质与提供个性化指导。

现有多数就业信息平台采用前后端耦合架构，扩展性差，且未平衡信息共享效率与合规审核需求。在此背景下，基于 Django 框架构建大学生就业信息共享平台具有现实必要性，其高安全性、模块化设计与前后端分离架构的灵活性，可有效破解上述痛点，为就业市场资源高效配置提供技术支撑。

1.3. 立项原因概述

1.3.1. 现存问题

1. 信息获取低效：学生求职信息分散于多渠道，格式不统一、更新不同步，筛选成本高；学生间就业经验多依赖线下传递，覆盖范围有限。
2. 招聘流程不规范：企业缺乏统一资质审核标准，招聘信息可能存在虚假宣传；简历筛选无精准工具，匹配率低、招聘周期长。
3. 平台管理缺失：高校就业服务数字化工具不足，难以整合信息与预警风险；现有平台技术架构落后，扩展性与安全性不足。

4. 交互体验不佳：部分平台操作流程繁琐，移动端适配不完善，用户学习成本高。

1.3.2. 机会点

1. 信息整合价值：通过平台集中多渠道就业信息，结合精准筛选功能，降低学生信息获取成本；搭建经验共享社区，打破隐性知识传递壁垒。
2. 合规与效率平衡：建立多层级企业资质审核机制，保障招聘信息真实性；采用智能匹配算法，提升企业招聘效率。
3. 技术架构优势：前后端分离架构降低系统耦合度，便于后续功能扩展；结合 Django 安全特性与 MySQL 数据存储能力，保障平台稳定运行。
4. 多主体服务升级：为高校提供数字化管理后台，强化就业指导与风险预警能力；优化用户交互流程，提升多端适配体验。

2. 需求概述

2.1. 面向的用户

A. 学生端用户

核心使用者为高校毕业生及在校求职学生。可通过 Web 端（支持移动端适配）完成注册登录、简历管理、招聘信息筛选、经验分享、求职提醒订阅等操作，获取精准岗位推荐与同伴经验支持。

B. 企业端用户

使用者为有招聘需求的企业 HR 或招聘负责人。通过 Web 管理后台提交资质材料完成入驻审核，审核通过后可发布招聘信息、筛选匹配简历、与学生在线沟通、管理招聘流程等。

C. 系统管理端用户

使用者为平台运营管理员及高校就业服务工作人员。通过 Web 管理后台进行用户管控、资质审核、内容合规审查、公告发布、数据统计、违规处理等操作，维护平台秩序与信息安全。

2.2. 系统概述

系统分为三个核心操作端，各端功能协同构建完整就业信息交互生态：

A. 学生端

核心功能包括：

账号管理：注册、登录、密码找回、个人信息编辑、实名认证；

简历管理：简历上传、在线编辑、多版本保存、预览与导出；

招聘信息模块：按行业/岗位/薪资等条件筛选、搜索、收藏、投递简历、查看投递状态；

经验共享模块：发布经验帖、浏览/点赞/评论/收藏他人内容、标签分类；

消息中心：接收系统通知、企业沟通消息、面试邀约提醒，支持已读/未读标记；

求职提醒：订阅岗位更新、投递反馈等实时通知。

B. 企业端（Web 管理后台）

核心功能包括：

入驻管理：提交资质材料、查看审核结果；

账号管理：登录、密码找回、企业信息编辑、操作员权限配置；

招聘管理：发布/编辑/下架招聘信息、设置岗位筛选条件；

简历管理：查看匹配简历、筛选简历、标记简历状态、下载简历；

沟通管理：与学生在线沟通、发送面试邀约/录用通知；

数据统计：查看招聘信息曝光量、投递量、筛选通过量等报表。

C. 系统管理端（Web 管理后台）

核心功能包括：

用户管理：查看学生/企业用户列表、账号冻结/解封/注销处理；

资质审核：审核企业入驻材料、记录审核日志；

内容审核：审核招聘信息/经验帖、处理违规内容、接收投诉举报；

公告管理：发布/编辑/下架公告、支持定向推送；

权限管理：配置管理员账号权限、划分操作范围；

数据管理：统计核心数据、生成/导出报表。

2.3. 开发工具与环境

工具/环境类型	具体配置
开发语言	Python 3.9+
后端框架	Django 4.2+、Django REST Framework 3.14+
前端框架	Vue.js 3.2+、Naive（UI 组件库）
数据库	MySQL 8.0+
开发工具	PyCharm 2023+、Visual Studio Code 1.80+
服务器环境	Nginx 1.20+（反向代理）、Gunicorn 20.1+（WSGI 服务器）
版本控制工具	Git
测试工具	Postman（接口测试）、Selenium（UI 测试）
运行环境	Windows 11

3. 系统描述

3.1. 系统总体概述

系统总体架构分为三层：

1. 表现层：Vue.js 开发的跨端适配界面，负责用户交互与数据展示；
2. 业务逻辑层：Django REST Framework 构建的 API 接口，处理核心业务逻辑；
3. 数据访问层：MySQL 数据库，负责用户、简历、招聘信息等核心数据存储。

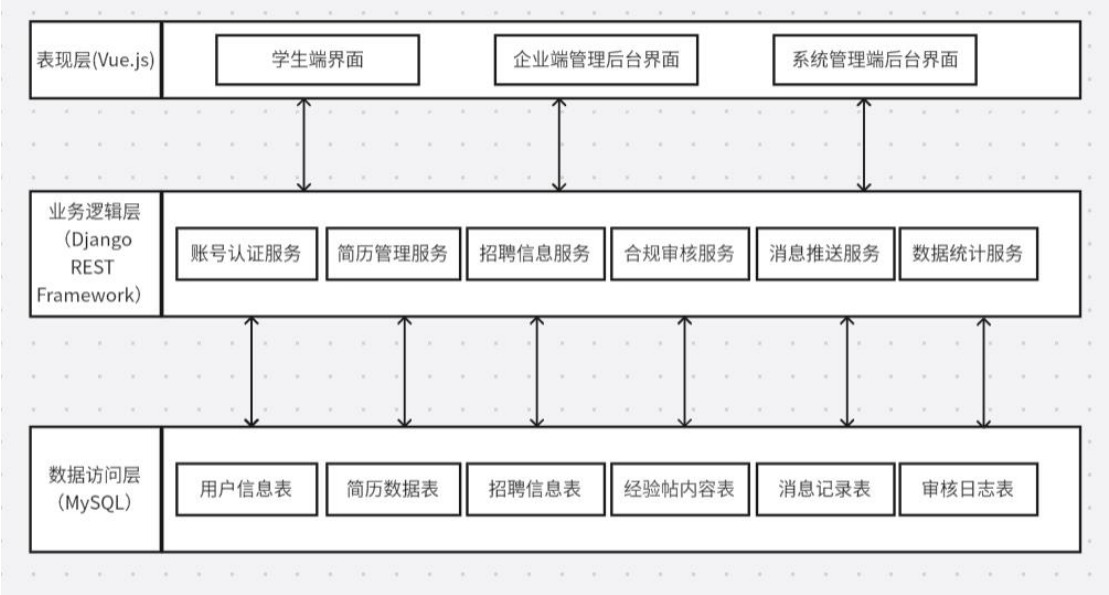

图 3.1 系统总体架构图

A. 学生端

以 “求职效率提升” 为核心，提供简洁直观的操作界面。首页展示推荐岗位、热门经验帖、平台公告；简历管理模块支持多版本编辑与格式导出；招聘信息模块提供多维度筛选与精准搜索功能；经验共享模块支持图文发布与互动交流；消息中心集中展示各类通知，确保用户不遗漏关键信息。

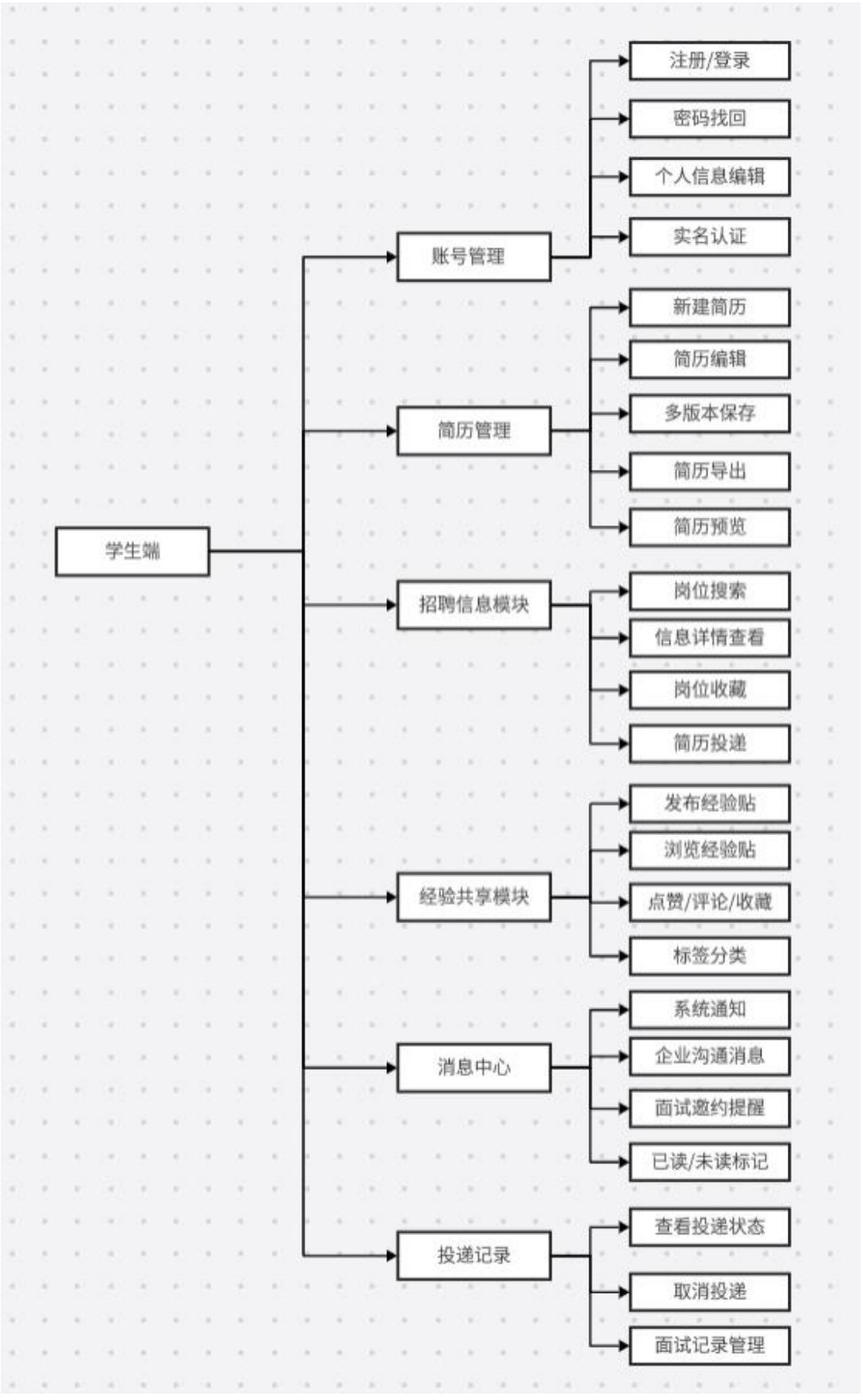

图 3.2 学生端功能模块图

B. 企业端

聚焦 “招聘流程规范化”，功能划分清晰。入驻审核模块引导企业提交资质材料；招聘发布模块提供标准化模板，支持岗位要求细化设置；简历管理模块集成智能匹配算法，按岗位条件自动筛选简历；数据统计模块可视化展示招聘效果，辅助企业优化招聘策略。

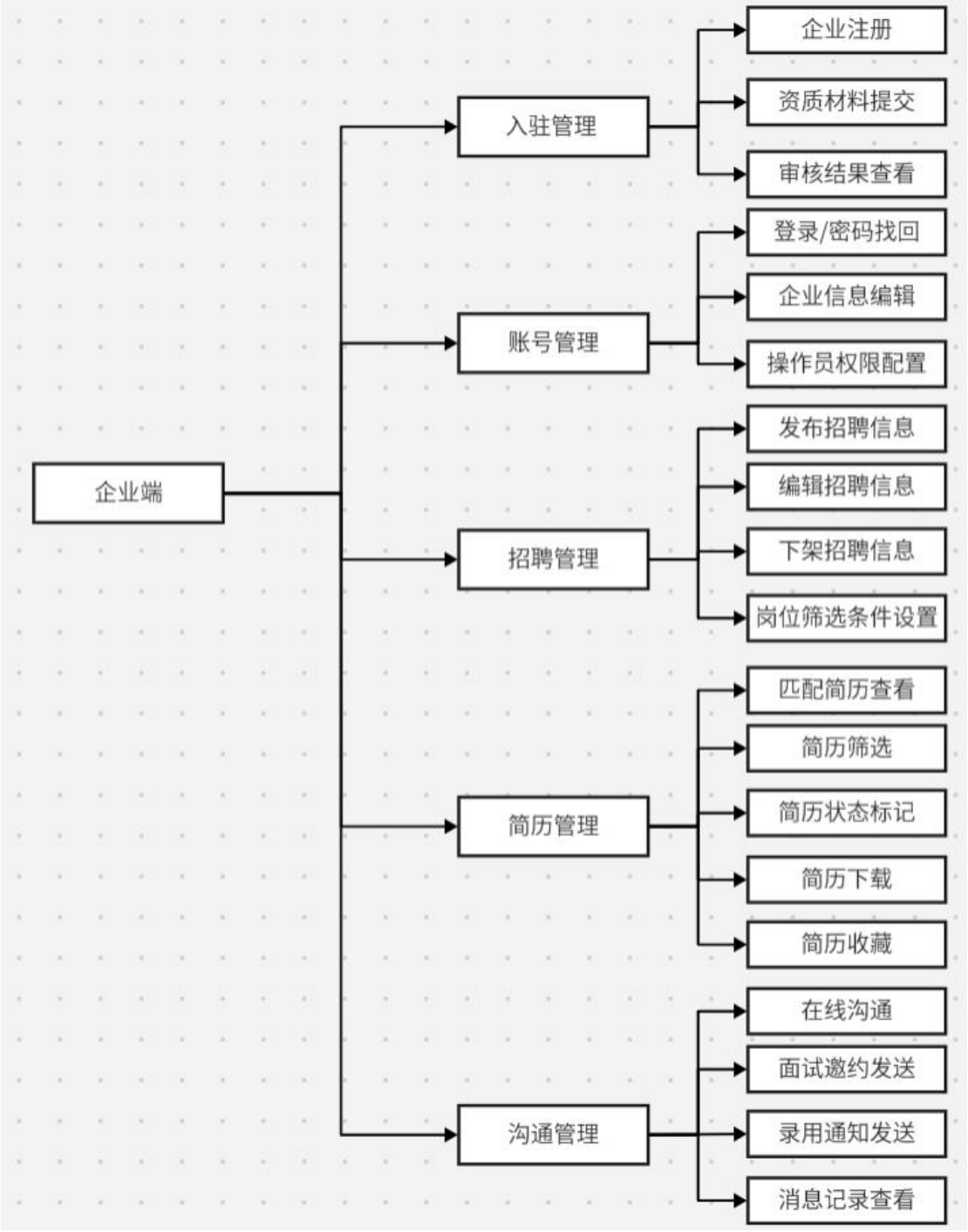

图 3.3 企业端功能模块图

C. 系统管理端

以 “平台合规运营” 为目标，构建多维度管理功能。资质审核模块支持材料在线核查与审核意见反馈；内容审核模块采用 “机器初审+人工复核” 机制；用户管理模块实现账号全生命周期管控；数据管理模块统计平台运营数据，为决策提供支撑。

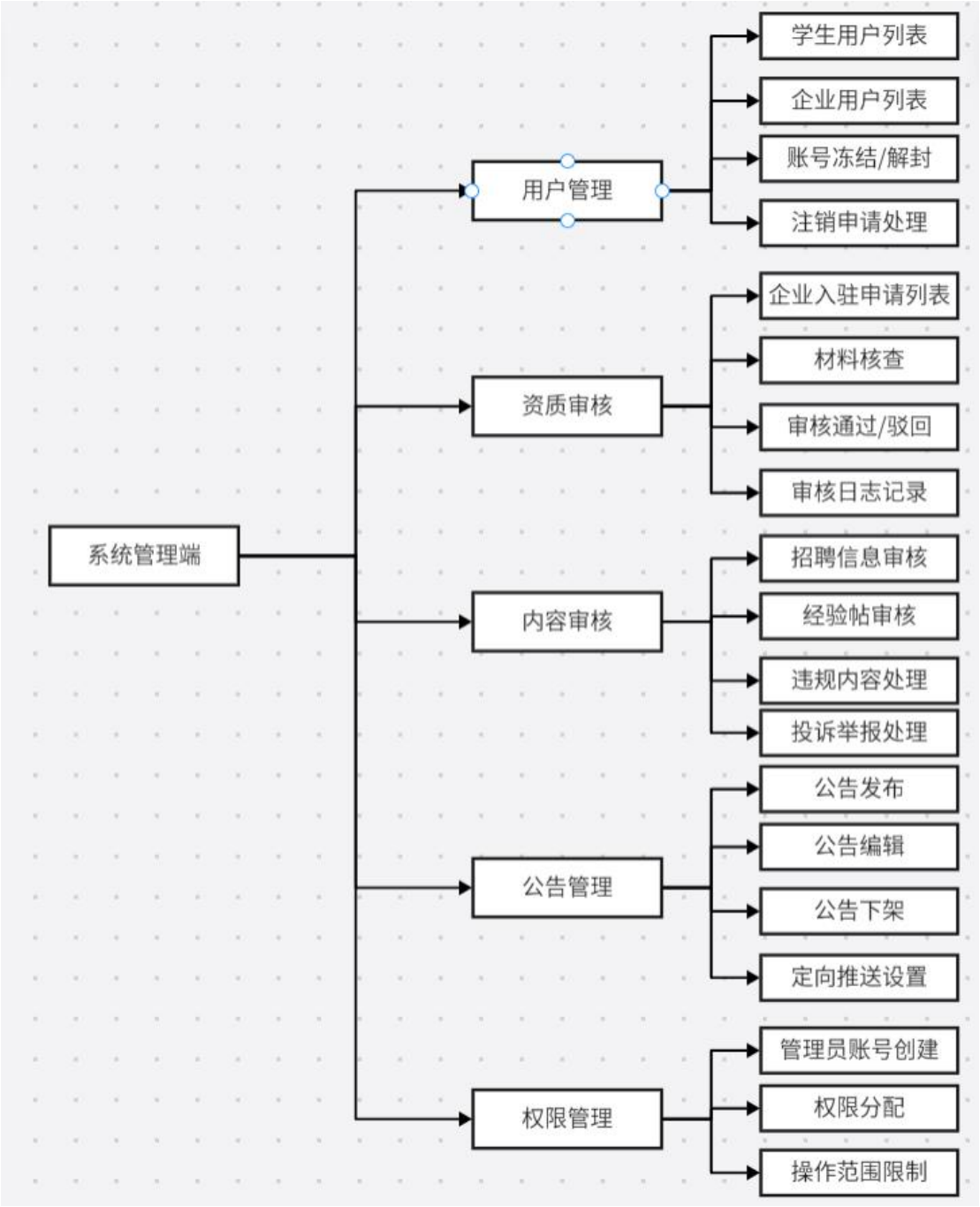

图 3.4 系统管理端功能模块图

3.2. 子系统信息交互的功能描述

信息交互子系统是平台核心子系统，负责三类用户间的信息流转与处理，核心功能包括：

1. 信息发布：支持学生发布经验帖、企业发布招聘信息、管理员发布公告；
2. 信息匹配：根据学生求职意向与企业岗位要求，实现招聘信息精准推送与简历匹配；
3. 合规审核：对企业入驻资质、招聘信息、经验帖内容进行多层级审核，保障信息质量；
4. 消息交互：实现学生与企业的在线沟通、系统通知推送、面试邀约发送等功能；
5. 数据同步：确保各端数据实时更新，如简历投递状态、审核结果、消息已读状态等。

4. 详细设计

4.1. 信息交互子系统结构

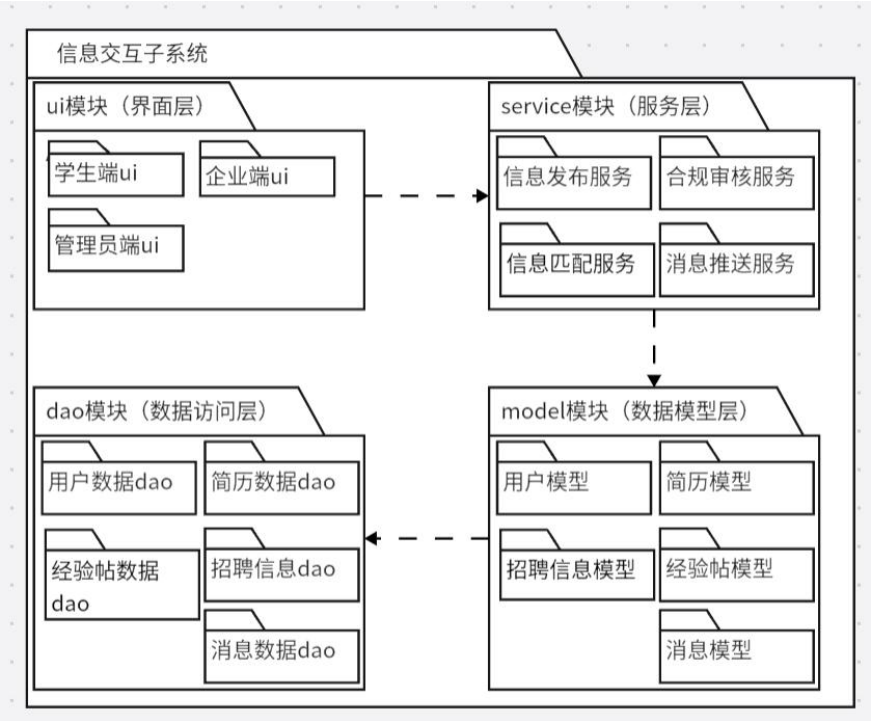

图 4.1 信息交互子系统结构图

4.2. 类图

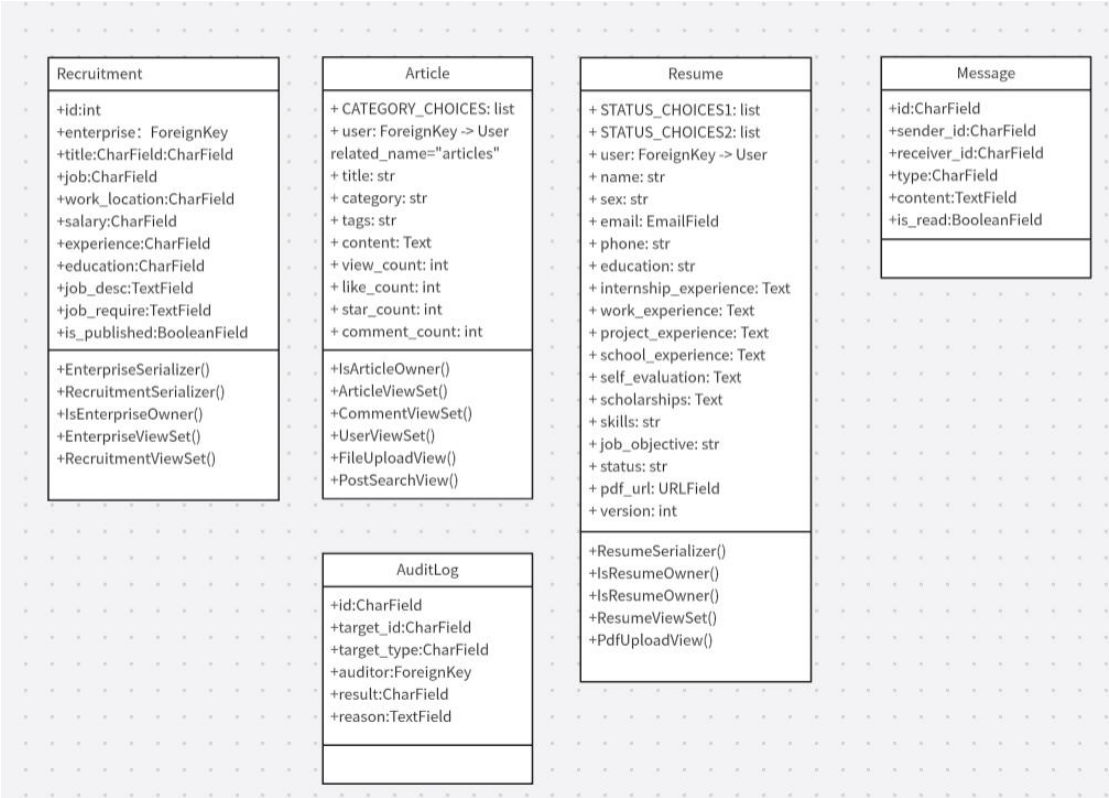

图 4.2 核心实体类图（包含属性和方法）

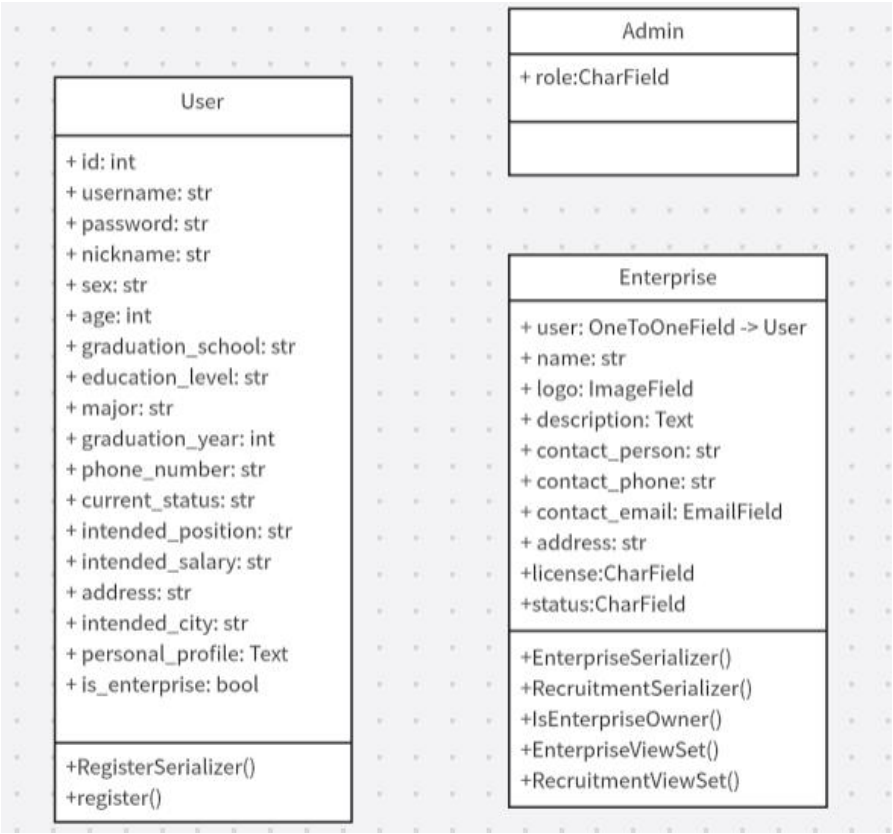

图 4.3 核心实体类图（包含属性和方法）

4.3. ui 模块设计说明

4.3.1. 模块描述

ui 模块是信息交互子系统的界面层，负责接收用户输入、展示系统处理结果，协调各服务模块的功能调用，确保用户操作流程顺畅。

4.3.2. 功能

提供三类用户的专属操作界面，适配不同使用场景；
接收用户操作请求（如简历上传、招聘发布、审核操作），验证输入合法性；
调用 service 模块的对应服务，传递请求参数；
接收服务返回结果，以直观形式展示（如列表、详情页、弹窗提示）；
实现消息实时推送与展示，支持已读/未读标记。

4.3.3. 交互的模块

service 模块（信息发布服务、信息匹配服务、合规审核服务、消息推送服务）

4.3.4. 模块设计

表 4.1 核心文件及类说明表

文件/类名	功能描述	核心方法
Home.vue（学生端主界面）	学生端入口，整合招聘、资源、社区等功能入口	- init(): 初始化页面，加载用户信息与推荐岗位 - handleMenuSelect(key): 跳转至目标功能模块（招聘/社区/资源） - handleSearch(): 关键词搜索岗位/经验帖 - handleToolClick(tool): 跳转至就业工具（简历生成/薪资查询）
ResumeCreate.vue（简历创建组件）	学生创建简历，支持表单填写与 PDF 上传	- handleSubmit(): 提交简历数据（含表单验证） - handleFileChange(params): 上传 PDF 简历文件- validateForm(): 校验姓名、邮箱等必填字段
ResumeEdit.vue（简历编	学生编辑已有简历	- fetchResumeDetail(): 加载简历详情并回显表单- handleSubmit(): 提交修改后的简历数据

文件/类名	功能描述	核心方法
辑组件）		
ResumeList.vue（简历列表组件）	展示学生所有简历，支持编辑/删除	- fetchResumes(): 加载当前用户的简历列表 - handleEdit(id): 跳转至简历编辑页 - handleDelete(id): 确认并删除简历
RecruitmentList.vue（企业招聘列表组件）	企业管理自身发布的招聘信息	- fetchRecruitments(): 加载企业发布的招聘列表- handleEdit(id): 编辑指定招聘信息 - handleTogglePublish(row): 切换招聘信息发布/下架状态 - handleDelete(id): 删除招聘信息
RecruitmentCreate.vue（招聘发布组件）	企业发布新招聘信息	- handleSubmit(): 提交招聘信息（含岗位要求、薪资等） - validateForm(): 校验招聘标题、工作地点等必填字段
CommunityIndex.vue（社区首页组件）	经验帖列表展示，支持筛选/搜索	- fetchPosts(): 加载经验帖列表（支持分类 / 标签筛选） - handleSearch(): 关键词搜索经验帖 - toPostDetail(id): 跳转至经验帖详情页
ArticlesDetail.vue（经验帖详情组件）	经验帖详情展示，支持点赞/收藏/评论	- fetchArticleDetail(): 加载帖子详情、作者信息 - handleLike(): 点赞/取消点赞帖子 - handleCollect(): 收藏/取消收藏帖子 - submitComment(): 发布评论 - handleFollow(): 关注/取消关注作者
CommunityCreate.vue（经验帖发布组件）	发布经验帖，支持富文本/标签/附件	- submitForm(): 提交帖子数据（标题/内容/标签） - handleFileSuccess(response): 上传附件（文档/图片） - selectTags(): 选择/添加帖子标签（最多 5 个）
EnterpriseRegister.vue（企业入驻组件）	企业用户注册与资质提交	- handleRegister():提交注册信息与资质材料 - checkForm(): 校验企业名称、营业执照等必填项 - submitMaterials(): 上传营业执照图片

文件/类名	功能描述	核心方法
EnterpriseEdit.vue（企业信息编辑组件）	企业完善/修改基本信息	- fetchEnterpriseInfo(): 加载企业信息并回显 - handleFileChange(fileList): 上传/更新企业Logo - handleSubmit(): 提交修改后的企业信息
UserInfo.vue（个人信息组件）	学生编辑个人信息与求职意向	- getUserInfo(): 加载用户信息并回显表单 - handleSave(): 提交个人信息修改（学历/意向岗位/城市）
MessageCenter.vue（消息中心组件）	消息展示与管理（系统通知/投递提醒/面试邀约）	- loadMessages(): 加载消息列表（支持按类型筛选） - markRead(messageId): 标记单条/批量消息为已读 - deleteMessage(messageId): 删除消息 - selectUnreadCount(): 统计未读消息数量

界面布局设计

- 学生端：采用 “顶部导航栏+中间内容区+底部功能栏（移动端）” 布局 PC 端为 “顶部导航+左侧功能菜单+右侧主内容区”，支持快速切换模块；
- 企业端：采用 “顶部导航栏+左侧功能树（企业信息/招聘管理/简历管理）+右侧操作区” 布局；
- 管理员端：采用 “顶部导航栏+左侧审核菜单（企业资质/招聘信息/内容审核）+右侧审核列表” 布局，支持批量审核与日志查看。

4.4. service 模块设计说明

4.4.1. 模块描述

service 模块是信息交互子系统的核心业务逻辑层，封装各类业务规则与处理流程，完全对齐前端 Vue 组件与后端 Django 视图功能。接收 ui 模块的请求后，调用 dao 模块执行数据操作，处理结果后返回给前端，涵盖简历、招聘、经验帖、评论点赞、消息通知全流程。

4.4.2. 功能

1. 信息发布服务：处理简历（含 PDF 附件）、招聘信息、经验帖（含附件/标签）的发布与更新，包含数据验证与状态初始化；
2. 信息匹配服务：基于关键词匹配算法（岗位要求 vs 简历技能），实现招聘信息与简历的精准匹配，支持个性化推荐；
3. 合规审核服务：执行企业资质审核、招聘信息审核、经验帖审核流程，实现审核状态流转与日志记录；
4. 消息推送服务：处理系统通知、简历投递提醒、面试邀约、评论点赞通知的发送，支持实时推送；
5. 评论点赞服务：处理经验帖的评论发布、点赞/取消点赞、收藏/取消收藏逻辑，维护互动数据一致性。

4.4.3. 交互的模块

ui 模块、dao 模块

4.4.4. 模块设计

表 4.2 核心服务类及方法表

服务类名	功能描述	核心方法
ResumeService（简历服务）	处理简历创建、更新、查询、导出、投递全流程	- createResume(resumeData, pdfFile): 创建简历（关联 PDF 附件） - updateResume(resumeId, resumeData, pdfFile): 更新简历（支持附件替换） - getResumeById(resumeId): 按 ID 查询简历详情 - exportResume(resumeId): 导出简历为 PDF 文件 - deliverResume(resumeId, recruitmentId): 向指定岗位投递简历（记录投递关系） - getUserResumes(userId): 获取当前用户的所有简历列表
RecruitmentService（招聘服务）	处理招聘信息发布、更新、下架、简历匹配	- createRecruitment(recruitmentData, enterpriseId): 发布招聘（关联企业，默认“待审核”状态） - updateRecruitment(recruitmentId, recruitmentData): 更新招聘信息 - offlineRecruitment(recruitmentId): 下架招聘（更新 is_published 为 False） - matchResumes(recruitmentId): 按岗位要求匹配符合条件的简历列表 - getRecruitmentStats(enterpriseId): 统计企业招聘数据（发布数/投递数/匹配数） - getPublishedRecruitments(enterpriseId): 获取企业已发布的招聘列表
PostService（经验帖服务）	处理经验帖发布、更新、删除、评论、点赞	- createPost(postData, attachments): 发布帖子（含图片/文档附件，标签去重） - updatePost(postId, postData): 更新帖子内容 - deletePost(postId): 删除帖子（级联删除评论/点赞记录） - auditPost(postId, auditResult, reason): 审核帖子（通过 / 驳回，记录原因） - getHotPosts(limit): 获取热门帖子（按浏览量 + 点赞数排序） - addComment(postId, commentData): 发布帖子评论 - toggleLike(postId, userId): 点赞/取消点赞帖子 - toggleCollect(postId, userId): 收藏/取消收藏帖子
CommentService（评论服务）	处理评论发布、点赞、删除	- createComment(commentData): 发布评论（关联帖子/用户） - toggleLike(commentId, userId): 点赞/取消点赞评论

服务类名	功能描述	核心方法
		- deleteComment(commentId, userId): 删除评论（仅作者 / 管理员可操作） - getPostComments(postId, page, size): 分页查询帖子的评论列表
AuditService（审核服务）	处理企业资质、招聘信息、经验帖审核	- auditEnterprise(enterpriseId, auditResult, reason): 审核企业资质（通过 / 驳回，更新 status） - auditRecruitment(recruitmentId, auditResult, reason): 审核招聘信息（敏感词检测+人工复核） - auditPost(postId, auditResult, reason): 审核经验帖（内容合规性校验） - getAuditList(auditType, status, page, size): 获取待审核列表（按类型/状态筛选） - recordAuditLog(auditLogData): 记录审核日志（审核人/结果/时间/原因）
MessageService（消息服务）	处理系统通知、互动消息、面试邀约	- sendMessage(messageData): 发送点对点消息（如简历投递提醒） - pushNotice(userId, noticeData): 推送系统通知（如审核结果 / 岗位推荐） - getMessagesByReceiver(receiverId, type, page, size): 分页查询接收者消息 - markRead(messageIds): 批量标记消息为已读 - deleteMessage(messageId): 删除消息 - getUnreadCount(receiverId): 统计未读消息数量
MatchService（匹配服务）	实现简历与岗位的智能匹配	- matchRecruitmentToStudent(studentId): 向学生推荐匹配岗位（基于求职意向+简历技能） - matchResumeToEnterprise(enterpriseId): 向企业推荐匹配简历（基于岗位要求） - calcMatchScore(resumeData, recruitmentData): 计算匹配度（0-100分，基于学历/经验/技能匹配）

业务流程设计

1. 简历投递流程：

学生提交投递请求（简历 ID+招聘 ID→ResumeService 验证（简历存在且属于当前用户、招聘信息已发布）→记录投递关系→MessageService 向企业推送投

递提醒→返回投递成功状态；

2. 招聘信息发布流程：

企业提交招聘信息→RecruitmentService 验证企业资质（已完善）→初始化状态为 “待审核”→AuditService 执行机器初审（敏感词检测→管理员人工复核→更新状态（已发布/驳回→MessageService 推送审核结果；

3. 经验帖点赞流程：

用户点击点赞→PostService 验证用户登录状态→检查是否已点赞→未点赞则创建点赞记录并更新帖子点赞数，已点赞则删除记录并减少点赞数→返回最新点赞状态与数量。

4.5. dao 模块设计说明

4.5.1. 模块描述

dao 模块是信息交互子系统的数据访问层，基于 Django ORM 封装数据库操作，为 service 模块提供标准化数据访问接口。屏蔽 MySQL 底层实现（如 SQL 拼接、连接管理），确保数据操作的安全性、一致性与高效性，完全对齐后端模型与视图功能。

4.5.2. 功能

1. 实现核心实体（用户、简历、招聘信息、经验帖、评论等）的 CRUD 操作，支持单条/批量处理；
2. 提供多条件组合查询（如按状态、时间、关联对象、关键词筛选），满足 service 层多样化业务需求；
3. 维护实体间关联关系（如学生 - 简历、企业 - 招聘、帖子 - 评论的一对多关系），确保数据完整性；
4. 优化查询性能（索引利用、关联预加载、分页处理），支撑高并发场景（如招聘高峰期、社区热门帖子访问）。

4.5.3. 交互的模块

service 模块、model 模块

4.5.4. 模块设计

表 4.3 核心 DAO 类及方法表

DAO 类名	功能描述	核心方法
UserDAO	处理学生、企业、管理员三类用户的通用数据操作	- insert(userData): 创建用户（支持学生/企业/管理员类型，密码加密存储） - update(userId, updateData): 更新用户信息（如联系方式、密码） - delete(userId): 逻辑删除用户（设置 is_deleted=True） - selectById(userId): 按 ID 查询用户详情（含关联角色信息） - selectByConditions(conditions, page, size): 多条件分页查询（如按角色、状态、关键词） - selectByUsername(username): 按用户名查询用户（用于登录校验）
StudentDAO	处理学生专属数据操作（继承 UserDAO）	- selectByStudentId(studentId): 按学号查询学生- updateJobIntention(studentId, intentionData): 更新求职意向（行业/岗位/城市） - selectByJobIntention(intentionConditions): 按求职意向查询学生（用于岗位推荐）
EnterpriseDAO	处理企业专属数据操作（继承 UserDAO）	- selectByLicense(license): 按营业执照编号查询企业（防重复入驻） - updateQualification(enterpriseId, qualificationData): 更新企业资质（营业执照/联系人信息） - selectByAuditStatus(status, page, size): 按审核状态查询企业列表 - selectByUserId(userId): 按关联用户 ID 查询企业信息
ResumeDAO	处理简历数据操作（含 PDF 附件关联）	- insert(resumeData, pdfUrl): 创建简历（关联学生 ID 与 PDF 附件 URL） - update(resumeId, updateData, pdfUrl): 更新简历（支持附件替换） - delete(resumeId): 删除简历（级联删除关联投递记录） - selectById(resumeId): 查询简历详情（含学生基本信息） - selectByStudentId(studentId, page, size):

		分页查询学生的所有简历 - selectByConditions(conditions): 按条件查询简历（用于岗位匹配）
RecruitmentDAO	处理招聘信息数据操作	- insert(recruitmentData, enterpriseId): 发布招聘（关联企业 ID） - update(recruitmentId, updateData): 更新招聘信息 - updateStatus(recruitmentId, status): 更新招聘状态（待审核/已发布/已下架） - delete(recruitmentId): 删除招聘信息 - selectById(recruitmentId): 查询招聘详情（含企业信息） - selectByEnterpriseId(enterpriseId, status, page, size): 按企业 ID 与状态查询招聘列表 - selectByConditions(conditions, page, size): 多条件查询招聘（行业/薪资/地点）
PostDAO	处理经验帖数据操作（含附件 / 标签）	- insert(postData, imageUrls): 发布帖子（关联学生 ID 与图片 URL） - update(postId, updateData): 更新帖子内容/标签 - delete(postId): 删除帖子（级联删除评论/点赞/收藏记录） - selectById(postId): 查询帖子详情（含作者信息） - selectHotPosts(limit): 按浏览量 + 点赞数查询热门帖子 - selectByConditions(conditions, page, size): 多条件查询帖子（分类/标签/关键词） - updateInteractionCount(postId, type): 更新互动数（浏览量/点赞数/评论数）
CommentDAO	处理评论数据操作	- insert(commentData): 发布评论（关联帖子 ID /用户 ID） - delete(commentId): 删除评论 - selectByPostId(postId, page, size): 分页查询帖子的评论 - selectById(commentId): 查询评论详情 - updateLikeCount(commentId, count): 更新评论点赞数
LikeDAO	处理点赞数据操作（帖子 / 评论）	- insert(likeData): 创建点赞记录（关联用户/目标对象） - delete(likeData): 删除点赞记录 - exists(userId, targetId, targetType): 检查是否已点赞 - countByTarget(targetId, targetType): 统

		计目标对象的点赞数
MessageDAO	处理消息数据操作	- insert(messageData): 发送消息 - updateReadStatus(messageIds, isRead): 批量标记消息已读 - delete(messageId): 删除消息 - selectByReceiver(receiverId, type, page, size): 分页查询接收者消息 - selectUnreadCount(receiverId): 统计未读消息数量
AuditLogDAO	处理审核日志数据操作	- insert(logData): 记录审核日志 - selectByTarget(targetId, targetType): 查询指定对象的审核历史 - selectByConditions(conditions, page, size): 多条件查询审核日志（审核人/结果/时间）

数据访问优化

1. 关联查询优化：使用 select_related（外键关联）和 prefetch_related（多对多关联）预加载数据，减少 “N+1 查询”（如查询帖子时同时加载作者信息）；
2. 索引策略：为高频查询字段建立索引，包括 student_id（简历表）、enterprise_id（招聘表）、status（全表通用）、receiver_id（消息表）、target_id（点赞/评论表）；
3. 分页与懒加载：所有列表查询默认支持分页（page+size 参数），大文本字段（如简历详情、帖子内容）采用懒加载；
4. 批量操作优化：批量创建/更新数据使用 bulk_create/bulk_update 方法（如批量标记消息已读）；
5. 缓存机制：对热门数据（热门招聘、经验帖）、静态数据（审核标准）使用 Redis 缓存，降低数据库压力。

4.6. model 模块设计说明

4.6.1. 模块描述

model 模块基于 Django 模型与序列化器，定义系统核心实体的数据结构，直接映射数据库表结构，明确实体间关联关系。封装字段约束与数据验证逻辑，为 dao 模块提供数据存储格式，为 service 模块与 ui 模块提供数据传输标准载体。

4.6.2. 功能

- 定义实体的字段属性（名称、类型、约束），映射数据库表结构；
- 声明实体间关联关系（继承、一对多），确保数据关联合理性；
- 实现字段级数据验证（格式校验、长度限制、枚举值约束）；
- 提供数据操作辅助方法（状态转换、时间格式化）。

4.6.3. 交互的模块

dao 模块、service 模块、ui 模块

4.6.4. 模块设计

表 4.4 核心数据模型及属性表

模型类名	核心属性	数据类型（Django 字段）	关联关系
User（用户基类）	id	CharField(primary_key=True, max_length=32)	抽象基类，被 Student、Enterprise、Admin 继承
	username	CharField(unique=True, max_length=50)	
	email	EmailField(unique=True)	
	password	CharField (max_length=128)（加密存储）	
	create_time	DateTimeField(auto_now_add=True)	
	is_deleted	BooleanField(default=False)	逻辑删除标记
Student（学生类）	student_id	CharField(unique=True, max_length=20)	继承自 User

模型类名	核心属性	数据类型（Django 字段）	关联关系
	real_name	CharField(max_length=20)	
	phone	CharField (max_length=11, validators=[手机号正则])	
	education	CharField (max_length=20, choices=[(' 本科 ',' 本科 '), (' 硕士 ',' 硕士 '), ...])	
	major	CharField(max_length=50)	
	job_intention	JSONField(default=dict)	求职意向（行业、岗位、城市）
Enterprise（企业类）	name	CharField(unique=True, max_length=100)	继承自 User
	industry	CharField (max_length=50, choices=[(' 互联网 ',' 互联网 '), (' 教育 ',' 教育 '), ...])	
	license	CharField(unique=True, max_length=50)	营业执照编号
	license_url	CharField(max_length=255)	营业执照图片 URL
	contact_person	CharField(max_length=20)	
	contact_phone	CharField(max_length=11)	
	contact_email	EmailField()	
	address	CharField(max_length=100)	
	status	CharField (max_length=20, choices=[(' 待审核 ',' 待审核 '), (' 已通过 ',' 已通过 '), (' 已驳回 ',' 已驳回 ')])	资质审核状态
Admin（管理员类）	role	CharField (max_length=20, choices=[(' 超级管理员 ',' 超级管理员 '), (' 内容审核员 ',' 内容审核员 '), ...])	继承自 User

模型类名	核心属性	数据类型（Django 字段）	关联关系
Resume（简历类）	id	CharField(primary_key=True, max_length=32)	
	student	ForeignKey(Student, on_delete=CASCADE, related_name='resumes')	与 Student 为多对一关系
	name	CharField(max_length=50)	简历名称（如 “2025 校招简历”）
	gender	CharField (max_length=2, choices=[('男','男'),('女','女')])	
	experience	TextField()	实习/工作经历
	skills	TextField()	技能描述（逗号分隔）
	pdf_url	CharField(max_length=255, null=True)	简历 PDF 附件 URL
	update_time	DateTimeField(auto_now=True)	
Recruitment（招聘信息类）	id	CharField(primary_key=True, max_length=32)	

模型类名	核心属性	数据类型（Django 字段）	关联关系
	enterprise	ForeignKey(Enterprise, on_delete=CASCADE, related_name='recruitments')	与 Enterprise 为多对一关系
	title	CharField(max_length=100)	招聘标题
	job	CharField(max_length=50)	岗位名称
	work_location	CharField(max_length=100)	工作地点
	salary	CharField(max_length=50, choices=[('5k-8k','5k-8k'), ...])	薪资范围
	experience	CharField(max_length=20)	工作经验要求
	education	CharField(max_length=20)	学历要求
	job_desc	TextField()	岗位职责
	job_require	TextField()	任职要求
	is_published	BooleanField(default=False)	发布状态
	create_time	DateTimeField(auto_now_add=True)	
Article（经验帖类）	id	CharField(primary_key=True, max_length=32)	

模型类名	核心属性	数据类型（Django 字段）	关联关系
	student	ForeignKey(Student, on_delete=CASCADE, related_name='posts')	与 Student 为多对一关系
	title	CharField(max_length=100)	
	content	TextField()	
	tags	CharField(max_length=100)	标签（逗号分隔）
	image_urls	CharField(max_length=512, null=True)	图片 URL（逗号分隔）
	view_count	PositiveIntegerField(default=0)	浏览量
	like_count	PositiveIntegerField(default=0)	点赞数
	comment_count	PositiveIntegerField(default=0)	评论数
	status	CharField (max_length=20, choices=[(' 待审核 ',' 待审核 '), (' 已发布 ',' 已发布 '), (' 已驳回 ',' 已驳回 ')])	审核状态
	create_time	DateTimeField(auto_now_add=True)	
Comment（评论类）	id	CharField(primary_key=True, max_length=32)	

模型类名	核心属性	数据类型（Django 字段）	关联关系
	post	ForeignKey(Post, on_delete=CASCADE, related_name='comments')	与 Post 为多对一关系
	user	ForeignKey(User, on_delete=CASCADE, related_name='comments')	与 User 为多对一关系
	content	TextField()	
	like_count	PositiveIntegerField(default=0)	点赞数
	create_time	DateTimeField(auto_now_add=True)	
Like（点赞类）	id	CharField(primary_key=True, max_length=32)	
	user	ForeignKey(User, on_delete=CASCADE, related_name='likes')	与 User 为多对一关系
	target_id	CharField(max_length=32)	点赞目标 ID（帖子/评论）
	target_type	CharField (max_length=20, choices=[('post',' 帖子 '), ('comment',' 评论 ')])	点赞目标类型
	create_time	DateTimeField(auto_now_add=True)	
Message（消息类）	id	CharField(primary_key=True, max_length=32)	

模型类名	核心属性	数据类型（Django 字段）	关联关系
	sender_id	CharField(max_length=32)	发送者 ID（用户/系统）
	receiver_id	CharField(max_length=32)	接收者 ID
	type	CharField (max_length=20, choices=[('system',' 系统通知 '), ('delivery',' 投递提醒 '), ('comment',' 评论通知 '), ('interview',' 面试邀约 ')])	
	content	TextField()	
	is_read	BooleanField(default=False)	是否已读
	create_time	DateTimeField(auto_now_add=True)	
AuditLog（审核日志类）	id	CharField(primary_key=True, max_length=32)	
	target_id	CharField(max_length=32)	审核对象 ID（企业/招聘/帖子）
	target_type	CharField (max_length=20, choices=[('enterprise',' 企业资质 '), ('recruitment',' 招聘信息 '), ('post',' 经验帖 ')])	
	auditor	ForeignKey(Admin, on_delete=SET_NULL, null=True, related_name='audit_logs')	与 Admin 为多对一关系
	result	CharField (max_length=10, choices=[(' 通过 ',' 通过 '), (' 驳回 ',' 驳回 ')])	
	reason	TextField(null=True)	驳回原因
	create_time	DateTimeField(auto_now_add=True)	

数据验证规则

必填项约束：username（用户）、name（企业/简历）、title（招聘/帖子）、license（企业）等核心字段设置 blank=False；

格式校验：手机号通过自定义验证，邮箱通过 Django 内置 EmailValidator 验证；

唯一性约束：username（用户）、student_id（学生）、license（企业）、name（企业）设置 unique=True；

枚举值限制：状态字段（status）、分类字段（industry/scale）通过 choices 参数限制可选值；

长度限制：字符串字段（job/title/tags）设置 max_length，避免冗余数据；

时间格式：所有时间字段统一使用 DateTimeField，自动按 “yyyy-MM-dd HH:mm:ss” 存储与转换。

5. 测试

5.1. 测试计划

5.1.1. 测试目标

验证系统是否满足需求中的功能需求、性能需求、安全性需求与易用性需求，覆盖学生端、企业端、管理员端全流程场景。确保系统在多环境下稳定、可靠、安全运行，为三类用户提供流畅的操作体验，同时验证缓存优化、数据关联等技术方案的有效性。

5.1.2. 测试范围

测试类型	测试内容	测试目的	测试工具与方法
功能测试	1. 账号管理（学生/企业/管理员注册、登录、权限区分） 2. 简历管理（创建、编辑、导出、投递、附件上传） 3. 招聘信息管理（发布、编辑、上下架、简历匹配、数据统计） 4. 经验共享（帖子发布、编辑、删除、点赞/收藏/评论、标签管理、附件上传） 5. 互动功能（帖子评论、评论点赞、作者关注） 6. 审核功能（企业资质、招聘信息、经验帖审核、日志记录） 7. 消息功能（系统通知、投递提醒、互动消息、已读标记） 8. 管理员功能（用户管理、批量审核、数据统计）	验证各功能模块按需求实现，流程顺畅，数据处理准确，权限控制有效	黑盒测试法，手工测试+Postman接口测试+数据库查询验证
界面测试	1. 各端界面元素（按钮、输入框、列表、弹窗）显示一致性 2. 界面布局合理性与响应式适配 3. 错误提示、操作反馈的准确性与易懂性 4. 功能入口清晰度与操作流程连贯性	验证界面符合设计规范，用户操作便捷，视觉效果统一，反馈及时	手工测试，覆盖主流浏览器与移动设备
性能	1. 页面加载/接口响应时间	验证系统性能满足需	JMeter 压力测试

测试类型	测试内容	测试目的	测试工具与方法
测试	（含缓存生效验证） 2. 并发用户访问稳定性（招聘高峰期/社区热门帖子） 3. 百万级数据查询效率（简历/招聘信息筛选） 4. 附件上传/下载速度（≤10MB 文件） 5. 系统连续运行稳定性（720 小时无故障）	求，缓存优化有效，高峰期无卡顿/崩溃	工具（模拟并发）；手工测试记录响应时间；Redis 缓存监控
安全性测试	1. 账号密码加密存储与 HTTPS 传输验证 2. 权限控制（越权访问防护，如学生无法访问企业管理端） 3. 防 SQL 注入、XSS 跨站脚本、CSRF 攻击 4. 敏感数据保护（简历、企业资质、用户隐私） 5. 操作日志完整性与可追溯性 6. 附件上传安全性（文件类型校验、大小限制）	验证系统安全防护有效，防止数据泄露与恶意攻击	手工测试（越权访问、输入特殊字符）；安全测试工具；接口权限校验
兼容性测试	1. 操作系统（Windows 11、Linux CentOS 7.9、） 2. 浏览器（Chrome 118+、Edge 117+、Firefox 118+） 3. 移动设备（Android 13+，手机 / 平板） 4. 屏幕分辨率（1920×1080、1366×768、移动端自适应）	验证系统在不同环境下运行正常，无兼容性问题	手工测试，覆盖不同环境与设备；浏览器兼容性测试工具
易用性测试	1. 操作流程简洁性（如简历投递≤3 步、帖子发布≤5 步） 2. 功能入口清晰度与导航连贯性 3. 错误提示易懂性与引导性 4. 用户学习成本（新用户独立完成核心操作≤10 分钟） 5. 互动反馈及时性（点赞/收藏实时刷新）	验证系统易于使用，降低不同用户的操作难度	手工测试，模拟新用户 / 熟练用户不同场景

5.1.3. 测试环境

环境类型	配置详情
硬件环境	服务器：CPU Intel Xeon E5-2670 v3，内存 16GB，硬盘 1TB，Redis 缓存服务器（8GB 内存）客户端：CPU 12th Gen Intel(R)Core(TM)i5-12500H，内存 16GB；移动设备：OnePlus Ace5、OPPO Pad3
软件环境	操作系统：服务器 CentOS 7.9，客户端 Windows 11、Android 13/14 浏览器：Chrome 118.0+、Edge 117.0+、Firefox 118.0+
数据库环境	MySQL 8.0.33，Redis 6.2.6（缓存环境）测试数据量：模拟用户 10 万条（学生 7 万、企业 2.5 万、管理员 0.5 万），简历 5 万条，招聘信息 2 万条，经验帖 3 万条，评论 8 万条
应用环境	后端：Django 4.2.7，Django REST Framework 3.14.0，Celery 5.3.6（异步任务） 前端：Vue.js 3.2.45，Naive UI 2.38.0，Vue Router 4.2.5 开发工具：Postman 10.21.0，JMeter 5.6.3

5.2. 测试用例设计

5.2.1. 界面测试

表 5.1 界面测试用例

序号	功能描述	输入/动作	期望结果	测试类型	备注
UI-001	学生端社区首页展示	打开学生端 Web 页面，进入“经验社区”模块	1. 顶部导航栏显示“首页、招聘、社区、资源、消息” 2. 左侧显示分类/标签筛选栏，中间显示经验帖列表（含标题、作者、互动数） 3. 右侧显示热门帖子/活跃用户，布局无重叠、无错位	界面测试	覆盖 Chrome、Edge 浏览器
UI-002	经验帖详情页面展示	点击任意经验帖进入详情页	1. 页面包含帖子标题、作者信息、发布时间、标签、内容、附件（如有）	界面测试	

序号	功能描述	输入/动作	期望结果	测试类型	备注
			2. 互动区域显示点赞、收藏、评论按钮，评论列表按时间倒序排列 3. 作者信息区显示关注按钮，布局清晰，操作入口直观		
UI-003	企业端招聘列表页面	登录企业端，进入“招聘管理”模块	1. 列表显示招聘标题、岗位、薪资、状态（已发布/已下架）、发布时间 2. 操作列显示“编辑、上下架、删除”按钮，状态标识清晰（绿色 = 已发布，灰色 = 已下架） 3. 支持批量选择与批量操作，按钮布局合理	界面测试	
UI-004	管理员审核页面	登录管理员端，进入“企业资质审核”模块	1. 列表显示企业名称、营业执照编号、提交时间、审核状态 2. 操作列显示“通过、驳回”按钮，支持批量审核 3. 驳回时弹出输入框，可填写驳回原因，界面交互流畅	界面测试	
UI-005	移动端经验帖发布	用手机打开社区发布页面	1. 表单自适应屏幕宽度，标签输入框支持回车添加（最多显示 5 个） 2. 富文本编辑器适配触摸操作，附件上传按钮尺寸≥48px×48px 3. 提交按钮固定在底部，无遮挡，点击反馈明显	兼容性测试 + 界面测试	覆盖 iOS、Android 设备
UI-006	错误提示显示	发布经验帖时输入 6 个标签	1. 实时提示“最多支持添加 5 个标签，请删除多余标签” 2. 标签输入框边框变红，提交按钮置灰不可点击 3. 提示文字简洁易懂，	界面测试 + 功能测试	

序号	功能描述	输入/动作	期望结果	测试类型	备注
			引导明确		

5.2.2. 简历管理功能测试

表 5.2 简历管理功能测试用例表

序号	功能描述	输入/动作	期望结果	测试类型	备注
RES-001	新建简历	登录学生端，进入 “简历管理”→“新建简历”，填写完整信息（姓名、学历、专业、实习经历等），点击 “保存”	1. 简历保存成功，跳转至简历列表页，显示 “新建成功” 提示 2. 列表显示简历名称、更新时间、PDF 附件状态（如有） 3. 数据库中新增简历记录，关联当前学生 ID	功能测试	
RES-002	上传 PDF 简历附件	新建简历时，点击 “上传文件”，选择≤10MB 的 PDF 文件	1. 文件上传成功，进度条显示完成，系统自动解析姓名、学历字段并填充 2. 解析后的字段可编辑，附件显示 “预览” “替换” 按钮 3. 数据库中存储附件 URL，关联当前简历 ID	功能测试	
RES-003	上传不支持格式附件	新建简历时，上传 15MB 的 PNG 图片文件	1. 弹窗提示 “仅支持 PDF/Word 格式，文件大小不超过 10MB” 2. 上传失败，无解析操作，界面保留原输入内容 3. 提示文字红色居中显示，引导明确	功能测试 + 易用性测试	
RES-004	简历导出	进入简历列表，选择某份含附件的简历，点击 “导出”	1. 成功导出 PDF 格式简历，内容包含表单填写信息 + 附件内容 2. 导出文件命名格式为 “姓名 - 简历 - 日期.pdf”，格式规范无错乱 3. 导出速度≤3 秒（10MB 内文件）	功能测试	
RES-005	简历多版本保存	编辑现有简历，修改 “技能” 字段，点击 “保存	1. 简历列表新增一条记录，标记 “版本 2”，原版本保留	功能测	

序号	功能描述	输入/动作	期望结果	测试类型	备注
		为新版本”	2. 版本切换入口清晰,可查看历史版本差异 3. 数据库中新增简历记录,版本号自动递增	试	
RES-006	简历投递	选择某招聘信息，点击“投递简历”，选择目标简历版本，确认投递	1. 弹窗提示“投递成功”，跳转至 “投递记录” 页面 2. 投递记录显示招聘岗位、企业名称、投递时间、状态（待查看） 3. 企业端收到投递提醒,数据库新增投递关联记录	功能测试	

5.2.3. 招聘信息管理功能测试

表 5.3 招聘信息管理功能测试用例表

序号	功能描述	输入/动作	期望结果	测试类型	备注
REC-001	发布招聘信息	登录企业端，进入 “招聘发布” 模块，填写完整信息（标题、岗位、薪资、要求等），点击 “提交”	1. 招聘信息保存成功，状态为 “待审核” 2. 管理员端收到审核提醒，列表显示该招聘信息 3. 数据库新增招聘记录，关联当前企业 ID	功能测试	
REC-002	招聘信息审核通过	管理员登录，进入 “招聘审核” 模块，选择待审核招聘信息，点击 “通过”	1. 招聘信息状态更新为“已发布” 2. 企业端收到审核通过通知，可在 “已发布” 列表查看 3. 学生端招聘列表可筛选到该信息,状态标识为“已发布”	功能测试	

序号	功能描述	输入/动作	期望结果	测试类型	备注
REC-003	招聘信息下架	企业端进入招聘列表，选择已发布的招聘信息，点击 “下架”	1. 弹窗提示“确认下架该招聘信息？下架后学生端将不可见”，点击 “确认” 2. 招聘状态更新为“已下架”，学生端无法筛选到该信息 3. 数据库中 is_published 字段更新为 False	功能测试	
REC-004	简历匹配推荐	企业端进入某招聘信息详情，点击 “匹配简历”	1. 系统按岗位要求（学历、专业、技能）匹配符合条件的简历列表 2. 列表显示简历匹配度（0-100 分）、学生姓名、学历、技能 3. 支持按匹配度排序，可直接查看简历详情	功能测试	
REC-005	招聘数据统计	企业端进入“数据统计” 模块，选择时间范围“近 30 天”	1. 显示招聘发布数、投递数、匹配数、查看数等核心指标 2. 图表展示投递趋势、简历来源分布，数据与数据库一致 3. 支持报表导出为 Excel 格式	功能测试	
REC-006	重复招聘标题发布	企业端发布与已发布招聘标题完全一致的信息	1. 弹窗提示“该招聘标题已存在，是否确认发布重复信息？” 2. 选择 “确认” 则正常发布，标题后自动添加“（2）” 标识 3. 选择 “取消” 则返回编辑页面，可修改标题	功能测试+易用性测试	

5.2.4. 经验帖互动功能测试

表 5.4 经验帖互动功能测试用例表

序号	功能描述	输入/动作	期望结果	测试类型	备注
POST-001	发布经验帖	登录学生端，进入 “发布经验” 模块，填写标题、富文本内容、3 个标签，上传 PDF 附件，点击 “提交”	1. 帖子提交成功，状态为 “待审核” 2. 管理员端收到审核提醒，列表显示该帖子 3. 数据库新增帖子记录，标签以逗号分隔存储，附件 URL 关联正确	功能测试	
POST-002	帖子点赞	进入经验帖详情页，点击 “点赞” 按钮（未点赞状态）	1. 点赞按钮颜色变红，点赞数+1，实时刷新无需刷新页面 2. 数据库 like_count 字段 +1，新增点赞关联记录（用户 ID + 帖子 ID） 3. 再次点击则取消点赞，点赞数-1，状态恢复	功能测试	
POST-003	帖子收藏	进入经验帖详情页，点击 “收藏” 按钮（未收藏状态）	1. 收藏按钮颜色变红，收藏数+1，弹窗提示 “收藏成功” 2. 学生端 “我的收藏” 模块显示该帖子 3. 数据库新增收藏关联记录，状态正确	功能测试	
POST-004	发布评论	进入经验帖详情页，输入评论内容 “很实用的经验！”，点击 “发表评论”	1. 评论发布成功，实时显示在评论列表顶部 2. 帖子评论数+ 1，评论显示用户名、头像、发布时间 3. 数据库新增评论记录，关联帖子 ID 与用户 ID	功能测试	
POST-005	评论点赞	点击某条评论的 “点赞” 按钮	1. 评论点赞数+1，按钮颜色变红 2. 数据库 comment 表中 like_count 字段+1 3. 再次点击取消点赞，数据同步更新	功能测试	
POST-006	关注作者	进入经验帖详情页，点击作者信息区 “关注” 按钮（未关注状态）	1. 关注按钮变为 “已关注”，作者粉丝数+1 2. 学生端 “我的关注” 模块显示该作者 3. 数据库新增关注关联记	功能测试	

序号	功能描述	输入/动作	期望结果	测试类型	备注
			录，状态正确		
POST-007	帖子审核驳回	管理员审核经验帖时，填写驳回原因 “内容包含敏感词”，点击 “驳回”	1. 帖子状态更新为 “已驳回” 2. 学生端收到驳回通知，显示驳回原因 3. 学生可编辑帖子重新提交，数据库记录审核日志	功能测试	

5.2.5. 管理员审核功能测试

表 5.5 管理员审核功能测试用例表

序号	功能描述	输入/动作	期望结果	测试类型	备注
AUD-001	企业资质审核通过	管理员登录，进入“企业资质审核” 模块，选择待审核企业，点击“通过”	1. 企业状态更新为 “已通过” 2. 企业端收到审核通过通知，可正常发布招聘信息 3. 数据库更新企业 status 字段，生成审核日志（审核人、时间、结果）	功能测试	
AUD-002	企业资质审核驳回	管理员选择待审核企业，点击 “驳回”，输入原因“营业执照模糊”	1. 企业状态更新为 “已驳回” 2. 企业端收到驳回通知，显示驳回原因，可重新提交材料 3. 数据库记录审核日志，驳回原因清晰存储	功能测试 + 易用性测试	
AUD-003	批量审核招聘信息	管理员勾选多条待审核招聘信息，点击 “批量通过”	1. 所选招聘信息状态均更新为 “已发布” 2. 对应企业端收到批量审核通知 3. 数据库批量更新状态，每条记录生成独立审核日志	功能测试	
AUD-004	审核日	管理员进入	1. 显示该企业的所有审核	功能	

序号	功能描述	输入/动作	期望结果	测试类型	备注
	志查询	“审核日志”模块，输入企业 ID 查询	记录（资质审核、招聘审核） 2. 日志包含审核时间、审核人、审核结果、驳回原因（如有） 3. 支持按时间范围筛选，查询结果准确	测试	
AUD-005	违规帖子处理	管理员发现经验帖包含违规内容，点击“下架”并填写原因	1. 帖子状态更新为“已下架”，学生端不可见 2. 学生端收到违规通知，显示下架原因 3. 数据库记录违规处理日志，支持后续追溯	功能测试 + 安全性测试	

5.2.6. 性能与安全性测试

表 5.6 性能与安全性测试用例表

序号	功能描述	输入/动作	期望结果	测试类型	备注
PERF-001	页面加载速度	打开学生端首页（缓存未命中）	1. 首页加载时间≤2 秒，核心内容（推荐岗位、热门帖子）优先显示 2. 图片懒加载生效，非首屏图片延迟加载 3. 缓存命中后，页面加载时间≤1 秒	性能测试	JMeter 记录
PERF-002	并发用户访问	用 JMeter 模拟 200 个并发用户访问社区首页	1. 系统无卡顿、无崩溃，响应时间≤3 秒 2. 数据库无死锁，Redis 缓存命中率≥80% 3. 服务器 CPU 使用率≤70%，内存占用稳定	性能测试	JMeter 压力测试
SEC-001	越权访问防护	学生端直接访问企业端招聘管理接口（修改 URL）	1. 接口返回 403 Forbidden，提示 “无访问权限” 2. 系统记录越权访问日志，包含 IP、时间、操作 3. 页面跳转至登录页或	安全性测试	手工测试

序号	功能描述	输入/动作	期望结果	测试类型	备注
			无权限提示页		
SEC-002	SQL注入防护	在招聘搜索框输入 “' OR 1=1 --”	1. 系统正常解析输入，无 SQL 注入漏洞 2. 搜索结果为正常匹配内容，无数据泄露 3. 系统记录异常输入日志，未执行恶意 SQL 语句	安全性测试	手工测试
SEC-003	敏感数据保护	抓包查看简历投递接口数据	1. 数据传输采用 HTTPS 加密，无明文泄露 2. 简历详情仅授权用户（投递企业、本人、管理员）可查看 3. 敏感字段（手机号、邮箱）在列表页部分隐藏（如 “151****8610”）	安全性测试	Wireshark 抓包验证

6. 结论及展望

6.1. 结论

本综合设计完成了基于 Django 的大学生就业信息共享平台的详细设计与全面测试方案制定。平台围绕 “学生-企业-系统管理” 三类用户需求，采用前后端分离架构，构建了信息发布、合规审核、精准匹配、互动交流、消息通知五大核心服务，覆盖简历管理、招聘发布、经验共享、审核管控全流程场景。

详细设计阶段明确了核心子系统 “信息交互子系统” 的四层架构（ui、service、dao、model），划分了各模块核心功能与交互逻辑，构建了完整的实体类图与数据关联关系，同时融入 Redis 缓存、批量操作优化等技术方案，为开发落地提供了标准化技术文档。测试阶段制定了全面的测试计划，设计了覆盖功能、界面、性能、安全、兼容性的多维度测试用例，重点验证了经验帖互动、管理员审核、企业招聘状态管理等新增核心功能，确保系统上线后能稳定、可靠、安全运行。

方案的创新点在于将 “合规审核”“精准匹配” 与 “互动交流” 深度融合，通过多层级审核机制保障信息质量，结合关键词匹配算法提升供需对接效率，同时通过 Redis 缓存优化高并发场景下的响应速度。但受限于设计范围，方案仍存在优化空间：一是匹配算法仅采用关键词匹配，精准度有待通过机器学习算法提升；二是移动端适配虽覆盖基础功能，但复杂操作（如富文本编辑）的交互细节可进一步优化；三是数据统计功能较为基础，缺乏就业趋势分析等深度洞察能力。

6.2. 展望

未来可从以下方面对系统进行优化与扩展：

1. 算法优化：引入机器学习算法（如协同过滤、逻辑回归），结合学生求职行为、企业招聘偏好、历史匹配数据，优化简历与岗位的匹配模型，提升推荐精准度；
2. 功能扩展：新增视频面试、内推机制、就业指导课程等模块，丰富平台服务场景；升级经验共享社区，增加话题分类、导师答疑、直播分享等功能，提升社区活跃度；
3. 体验提升：深化移动端适配，开发独立 APP，优化触摸操作体验与离线缓存功能；引入智能客服机器人，实时解答用户常见疑问，降低学习成本；

4. 生态构建：规划与高校就业系统、企业 HR 系统的对接方案，设计标准化接口实现数据互通；引入第三方服务（如背景调查、薪资查询、职业测评），完善就业服务生态；
5. 性能与安全优化：扩大 Redis 缓存覆盖范围（如热门简历、招聘信息），进一步提升响应速度；引入分布式数据库，支撑千万级数据存储与访问；强化安全防护，增加验证码、登录异常检测等功能，防范恶意攻击。

7. 参考文献

[1]林彬,杨彬彬,孙芳.基于 Django 框架的高校就业信息可视化平台的设计与实现[J].辽宁师范大学学报(自然科学版),2022,45(02):201-208.DOI:CNKI:SUN:LNSZ.0.2022-02-009.

[2]刘洋.基于 WEB 结构学生就业信息平台的设计与实现[J].自动化与仪器仪表,2016,(06):146-148.DOI:10.14016/j.cnki.1001-9227.2016.06.146.

[3]孙军,刘洋,吴枫,等.基于高校的信息共享平台功能设计与实现[J].项目管理技术,2019,17(09):102-107.DOI:CNKI:SUN:XMGJ.0.2019-09-018.

[4]朱鹏威,曹烨帆,董天舒.“互联网+”视阈下高校信息化教学资源共享平台建设研究[J].情报科学,2016,34(12):133-136.DOI:10.13833/j.cnki.is.2016.12.025.

[5]刘昶,李威,王德志.基于 Django 与 Vue 的煤矿企业员工心理健康平台的设计与实现[J].现代计算机,2024,30(21):206-210+216.DOI:CNKI:SUN:XDJS.0.2024-21-039.

[6]陈峰,斯佳佳.基于 Celery 的国产服务器测试平台设计与实现[J].计算机测量与控制,2024,32(09):80-85.DOI:10.16526/j.cnki.11-4762/tp.2024.09.012.

[7]胡从寅,杨文远,赵鑫,等.基于 Django+Vue.js 的设计作品交易平台的实现[J].软件,2023,44(11):42-46.DOI:CNKI:SUN:RJZZ.0.2023-11-045.

[8]陈丽梅.基于微信小程序的高校就业信息平台的设计与实现[J].信息与电脑(理论版),2023,35(07):139-141.DOI:CNKI:SUN:XXDL.0.2023-07-039.