

苏州科技大学

软件新技术应用与创新创业开发实践

设计题目 人才管理系统

院 (系) 电子与信息工程

专 业 计算机科学与技术

组 别 3 组 长 孟宇

指导教师 傅启明/程涵婧

2025 年 7 月 1 日

成绩评定表

学号	姓名	小组评分	平时成绩	验收成绩	总评成绩
22200107129	孟宇	100			
22200107130	朱轩	100			
22200107131	姚小威	100			
22200107126	王帅	100			
2220010736	王冬升	100			

注：小组评分为百分制，由组长根据各组员贡献，如实打分。

人才管理系统

摘 要

本人才管理系统旨在搭建个人用户与企业用户之间的双向交互平台,实现人才与岗位的高效匹配。系统支持个人用户注册、简历发布,企业用户注册及招聘信息发布,管理员对企业信息进行审核管理。通过搜索功能,用户可快速获取人才或招聘信息,满足企业招聘与个人求职需求。系统采用 B/S 架构,基于 Java EE 技术栈开发,数据库选用 MySQL,确保系统的稳定性与可扩展性。本文详细阐述系统的需求分析、设计方案及实现细节,为系统开发提供完整的技术文档支持。

关键词 人才管理系统; 简历发布; 招聘信息; 管理员审核; 双向匹配

目 录

成绩评定表.....	IV
第 1 章 绪论.....	1
1.1 引言.....	1
1.2 研究内容.....	1
1.3 本文结构.....	2
第 2 章 需求分析.....	3
2.1 功能需求.....	3
2.1.1 用户注册模块.....	3
2.1.2 简历发布模块.....	3
2.1.3 企业审核模块.....	3
2.1.4 招聘信息发布模块.....	3
2.1.5 系统管理模块.....	3
2.1.6 信息搜索模块.....	3
2.2 非功能需求.....	4
2.3 用例详细说明.....	4
2.3.1 个人用户用例.....	4
2.3.2 企业用户用例.....	7
2.3.3 管理员用例.....	11
第 3 章 概要设计.....	13
3.1 系统架构设计.....	13
3.2 功能模块划分.....	13
第 4 章 数据库设计.....	17
4.1 数据库选型.....	17
4.2 数据库概念设计.....	17
4.3 数据库逻辑设计.....	18
4.3.1 企业用户表（accounts_enteriseprofile）.....	18
4.3.2 个人用户表（accounts_individualprofile）.....	18
4.3.3 用户基础信息表（accounts_user）.....	19
4.3.4 职位申请表（jobs_application）.....	20
4.3.5 职位表（jobs_job）.....	20
4.3.6 教育经历表（resumes_education）.....	21
4.3.7 工作经历表（resumes_experience）.....	21
4.3.8 简历表（resumes_resume）.....	22
第 5 章 详细设计.....	24
5.1 用户管理模块.....	24
用户注册.....	24
用户登录.....	24
5.1.3 用户信息完善.....	24
5.2 企业审核模块.....	24
5.2.1 待审核企业列表.....	24

5.2.2 企业审核操作.....	24
5.3 简历管理模块.....	25
5.3.1 简历列表.....	25
5.3.2 简历创建和编辑.....	25
5.3.3 简历删除.....	25
5.4.1 招聘信息发布.....	25
5.4.2 招聘信息列表.....	25
5.4.3 招聘信息管理.....	25
第 6 章 系统实现.....	26
6.1 用户管理模块实现.....	26
6.1.1 用户注册功能.....	26
6.1.2 用户登录与注销功能.....	27
6.1.3 用户资料完善功能.....	27
6.2 企业审核模块实现.....	28
6.2.1 管理员查看待审核企业用户.....	28
6.2.2 管理员审核企业用户.....	29
6.3 简历管理模块实现.....	30
6.3.1 创建简历功能.....	30
6.3.2 更新简历功能.....	31
6.3.3 删除简历功能.....	32
6.4 招聘信息管理模块实现.....	33
6.4.1 发布招聘信息功能.....	33
6.4.2 更新招聘信息功能.....	33
6.4.3 删除招聘信息功能.....	34
6.4.4 申请职位功能.....	34
6.4.5 查看申请信息功能.....	35
第 7 章 测试报告.....	36
7.1 功能测试.....	36
7.1.1 用户注册.....	36
7.1.2 用户登录.....	37
7.1.3 企业审核.....	38
7.1.4 简历管理.....	38
7.1.5 招聘信息管理.....	40
7.1.6 搜索模块.....	41
结 论.....	44
参 考 文 献.....	45

第1章 绪论

1.1 引言

在当今数字化浪潮席卷全球的时代，信息技术的飞速发展正在深刻地重塑各个行业，人力资源管理领域也不例外。随着企业运营模式的数字化转型加速，传统的人力资源管理方式正逐渐被智能化、数字化的解决方案所取代。招聘管理系统作为企业与求职者高效对接的核心工具，其重要性日益凸显。它不仅是企业筛选和匹配人才的关键平台，更是推动人力资源行业创新发展的重要引擎。

近年来，随着互联网技术的普及和移动设备的广泛使用，在线招聘市场规模呈现出爆发式增长。根据艾瑞咨询的权威报告显示，2023 年中国在线招聘市场规模已经突破千亿元大关，并且预计在未来五年内仍将保持 15%以上的年复合增长率。这一数据充分表明，招聘管理系统的市场前景极为广阔，其在企业人力资源管理中的地位也将进一步提升。

然而，市场的快速发展也带来了新的挑战。一方面，企业对人才质量的要求越来越高，不再仅仅满足于传统的简历筛选方式，而是需要更加精准、高效的人才匹配机制。另一方面，求职者对个性化服务的需求也日益增长，他们希望能够在招聘平台上获得更加便捷、个性化的求职体验。与此同时，随着《个人信息保护法》《网络安全法》等一系列法规的严格实施，招聘管理系统在数据安全和隐私保护方面的责任也愈发重大。这些法规对招聘管理系统提出了更高的要求，促使其在功能完整性、用户体验及数据安全等方面实现全方位的升级。

1.2 研究内容

在绪论部分，本报告将开篇点明行业发展趋势与研究必要性，为后续内容奠定坚实的理论基础。主体章节将依次对 15 个核心功能用例（从 UC1 至 UC15）进行深度剖析。每个用例都将包含功能描述、参与者界定、主成功场景、扩展流程、特别需求等内容，以确保对每个功能模块的全面分析。同时，本报告将结合 PowerDesigner 工具绘制专业的用例图，直观地展示功能流程与系统交互关系，使读者能够更加清晰地理解系统的整体架构和各功能模块之间的关联。

在研究方法上，本报告将采用多种研究手段相结合的方式。首先，通过对行业现状和趋势的深入调研，收集大量一手和二手资料，以确保研究的前沿性和科

学性。其次，结合实际案例分析，对招聘管理系统在不同应用场景下的表现进行剖析，从而总结出系统的优缺点和改进方向。最后，通过与行业专家的交流与合作，获取专业意见和建议，进一步完善研究内容。

本报告的研究意义不仅在于为招聘管理系统的开发提供理论支持和实践指导，还在于推动整个人力资源管理行业的数字化转型。通过构建系统化、规范化的用例设计体系，本报告将为招聘管理系统的设计、开发和优化提供极具参考价值的理论依据。同时，通过对各功能模块的深入分析，本报告将为系统的实际应用提供实践指导，帮助企业更好地应对市场变化，提升人力资源管理效率。

在预期成果方面，本报告将形成一套完整的招聘管理系统用例设计体系，涵盖系统的主要功能模块和关键交互流程。此外，本报告还将提出一系列功能优化与系统迭代的方向，为招聘管理系统的持续演进提供参考。通过这些成果，本报告希望能够为招聘管理系统的设计者、开发者以及使用者提供有价值的参考，推动招聘管理系统在数字化浪潮中不断创新与发展，为企业和求职者创造更大的价值。

1.3 本文结构

本文依次介绍系统的需求分析、概要设计、数据库设计、详细设计、系统实现及测试方案，最后总结系统的设计成果与未来改进方向。

第2章 需求分析

2.1 功能需求

2.1.1 用户注册模块

个人用户：通过用户名注册，填写信息（姓名、性别等），设置登录密码。

企业用户：提交企业名称、地址、营业执照等资质信息进行注册。

2.1.2 简历发布模块

个人用户注册后可创建、编辑简历，内容包括教育背景、工作经历等。

2.1.3 企业审核模块

管理员对企业提交的注册信息进行审核，审核通过后企业成为正式会员，可发布招聘信息。

2.1.4 招聘信息发布模块

正式会员企业可发布岗位信息，包括职位名称、职责描述、薪资范围等。

2.1.5 系统管理模块

管理员企业审核管理，企业信息管理，个人信息等功能，以保障系统的安全、稳定运行与灵活配置。

2.1.6 信息搜索模块

支持按关键词搜索个人简历和企业招聘信息

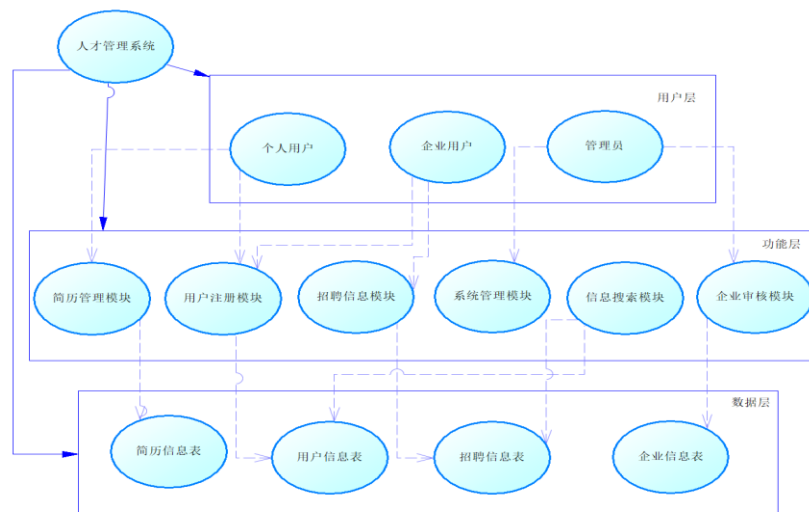


图 2.1 系统结构图

结构说明：

(1) 用户层：

个人用户：通过注册模块创建账号，使用简历管理模块维护个人信息，通过

搜索模块查询招聘信息。

企业用户：注册后需通过审核成为正式会员，使用招聘信息模块发布岗位，通过搜索模块查询人才简历。

管理员：通过企业审核模块管理企业资质，通过系统管理模块维护后台数据。

（2）功能层：

用户注册模块：支持个人 / 企业用户注册，验证账号信息（如手机号、企业资质）。

简历管理模块：个人用户创建、编辑简历，设置隐私权限。

企业审核模块：管理员对企业提交的注册信息进行审核（如营业执照、信用代码），更新企业状态（未审核 / 通过 / 驳回）。

招聘信息模块：企业用户发布、修改招聘岗位，查看应聘简历；个人用户查看岗位详情并投递简历。

信息搜索模块：提供多条件搜索（如关键词、岗位类型、技能标签），关联用户信息表和招聘信息表数据。

系统管理模块：管理员维护用户账号、数据备份、日志查看等系统级操作。

（3）数据层：

存储用户基础信息（用户信息表）、个人简历详情（简历信息表）、企业资质及审核状态（企业信息表）、招聘岗位详情（招聘信息表）。

模块通过数据库接口实现数据的增删改查（如用户注册时向用户信息表插入数据，搜索时关联多表查询）。

2.2 非功能需求

安全性：用户密码加密存储，企业资质信息审核确保数据真实性；

性能：搜索响应时间 ≤ 1 秒；

易用性：界面简洁直观，操作流程清晰，提供注册引导与帮助文档；

2.3 用例详细说明

2.3.1 个人用户用例

（1）注册账号（UC1）

用例 ID	UC1	用例名	个人用户注册账号
-------	-----	-----	----------

功能描述	个人用户通过手机号 / 邮箱注册，填写姓名、年龄等基本信息，设置密码并验证验证码。
用例体	
主执行者	个人用户
输入/前置条件	用户已访问注册页面，输入必填信息（用户名、密码）且系统服务正常。
主成功场景	用户填写信息→系统校验通过→加密存储数据→跳转登录页面。
特别需求	密码加密存储、HTTPS 传输、2 秒内完成验证码发送。

（2）发布 / 管理简历（UC2）

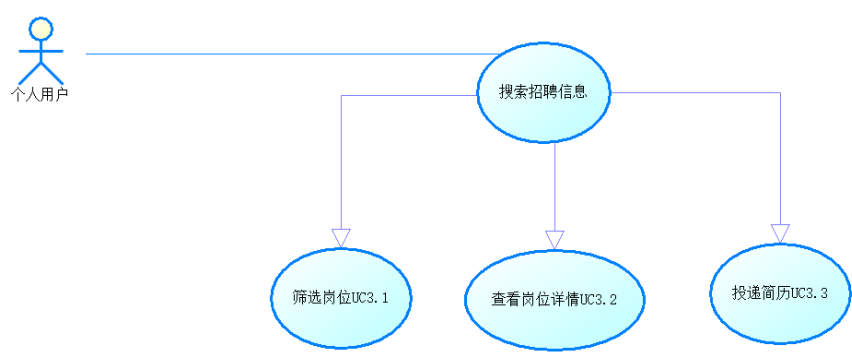
描述：用户创建简历，填写教育背景、工作经历、技能标签，支持编辑、删除

用例 ID	UC2	用例名	发布/管理简历
功能描述	用户创建简历，填写教育背景、工作经历、技能标签，支持编辑、删除及隐私设置（公开 / 仅企业可见 / 仅自己可见）。		
用例体			
主执行者	个人用户		
输入/前置条件	用户已登录系统，进入“我的简历”页面，需至少填写简历		

	基本信息（如姓名、教育背景）。
主成功场景	用户点击“新建简历”→填写教育 / 工作经历 / 技能→保存发布，或对已有简历执行编辑、删除操作。
扩展	异常处理：简历内容为空时提示“请填写完整信息”；删除时二次确认防止误操作； 权限控制：仅允许编辑 / 删除本人简历。
特别需求	支持富文本编辑（如加粗 / 列表）、附件上传（证书扫描件，≤20MB），隐私设置需符合数据安全规范。

（3）搜索招聘信息（UC3）

描述：按关键词（如职位名称、企业名称）、薪资范围、地区筛选岗位，查看岗位详情并投递简历。

用例 ID	UC3	用例名	搜索招聘信息
功能描述	按关键词（如职位名称）、地区筛选岗位，查看岗位详情并投递简历。		
用例体	 <pre> graph TD User((个人用户)) --- Search((搜索招聘信息)) Search --> Filter((筛选岗位UC3.1)) Search --> Details((查看岗位详情UC3.2)) Search --> Apply((投递简历UC3.3)) </pre>		
主执行者	个人用户、企业用户（企业用户可搜索人才简历，此处聚焦招聘信息搜索）		
前置条件	用户已登录系统，系统存在有效招聘信息数据。		
主成功场景	用户输入关键词→选择筛选条件（地区）→查看岗位列表→点击详情→选择已发布简历投递→接收投递成功通知。		
扩展	无结果处理：搜索无匹配岗位时提示“暂无相关职位，试试其他关键词”；		

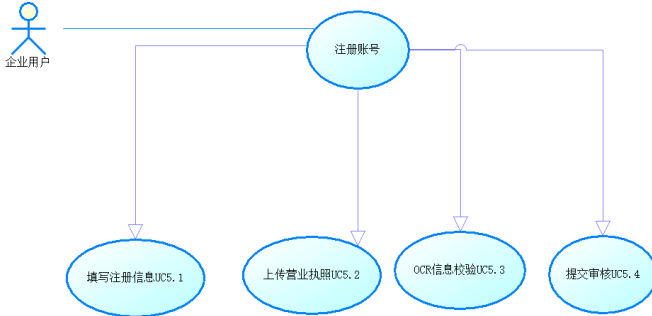
特别需求	支持模糊匹配(如关键词“开发”匹配“Java 开发”“前端开发”), 搜索响应时间≤3 秒, 投递记录可在“我的申请”中查看。
------	--

2.3.2 企业用户用例

(1) 注册账号 (UC4)

描述: 企业用户提交企业名称、信用代码、联系方式及营业执照图片, 申请注册。

扩展点: 系统自动识别营业执照文字信息 (OCR 技术), 与填写内容校验一致性。

用例 ID	UC4	用例名	企业用户注册账号
功能描述	企业用户提交企业名称、地址、注册时间及营业执照图片, 申请注册。 扩展点: 系统自动识别营业执照文字信息 (OCR 技术), 与填写内容校验一致性。		
用例体	 <pre> graph TD User((企业用户)) --- UC4((注册账号)) UC4 --> UC5.1((填写注册信息UC5.1)) UC4 --> UC5.2((上传营业执照UC5.2)) UC4 --> UC5.3((OCR信息校验UC5.3)) UC4 --> UC5.4((提交审核UC5.4)) </pre>		
主执行者	企业用户		
输入/前置条件	企业用户已访问企业注册页面, 准备好企业名称、地址、营业执照图片 (JPG/PNG, ≤5MB)。		
主成功场景	企业用户填写注册信息→上传营业执照图片→系统调用 OCR 接口识别文字→自动校验信息一致性 (如企业名称) →校验通过后提交审核, 生成待审核记录。		
特别需求	营业执照 OCR 识别准确率≥95%, 支持多字体、多版式识别; 注册信息加密传输 (HTTPS), 营业执照图片存储符合数据合规要求 (如脱敏处理)。		

(2) 提交企业审核 (UC5)

描述：注册后向管理员提交资质审核，等待审核结果（通过 / 驳回），驳回时可查看审核意见。

用例 ID	UC5	用例名	提交企业审核
功能描述	企业用户提交企业名称、地址及营业执照图片，申请注册。 扩展点：系统自动识别营业执照文字信息，与填写内容校验一致性。		
用例体	<pre> graph TD User((企业用户)) --> UC6.1((发起审核请求UC6.1)) UC6.1 --> UC5((提交企业审核)) UC5 --> UC6.2((管理员审核UC6.2)) Admin((管理员)) --> UC6.2 UC6.2 --> UC6.3((通知审核结果UC6.3)) UC6.3 -.-> System((企业审核系统)) System -.-> User </pre>		
主执行者	企业用户		
输入/前置条件	企业用户已完成注册 (UC4)，且注册信息状态为“待审核”； 提交审核前系统已通过 OCR 校验注册信息与营业执照一致性。		
主成功场景	企业用户在注册流程中或“企业中心”点击“提交审核”； 系统生成审核请求，状态更新为“审核中”； 管理员收到审核通知，查看企业资质文件； 管理员操作“通过”或“驳回”： 通过：企业状态变为“正式会员”，可发布招聘信息； 驳回：企业收到通知，可查看审核意见（如“营业执照过		

	期”)。
特别需求	审核结果需在 24 小时内通知企业（通过站内信 + 短信）； 驳回意见需具体明确（如“营业执照图片不清晰，请重新上传”）； 审核记录可追溯，管理员操作需留痕。

（3）发布 / 管理招聘信息（UC6）

描述：发布岗位名称、职责、薪资等信息，支持修改、下架招聘信息，查看应聘简历并筛选候选人。

用例 ID	UC6	用例名	发布/管理招聘信息
功能描述	发布岗位名称、职责、薪资等信息，支持修改、下架招聘信息，查看应聘简历并筛选候选人。		
用例体	<pre> graph TD Actor[企业用户] UC7.1((发布岗位信息UC7.1)) UC7.2((修改/下架岗位UC7.2)) UC7.3((查看应聘简历UC7.3)) UC7.4((筛选候选人UC7.4)) Actor --- UC7.1 Actor --- UC7.2 Actor --- UC7.3 Actor --- UC7.4 UC7.1 -.- UC7.4 </pre>		
主执行者	企业用户		
输入/前置条件	企业用户已通过资质审核（状态为“正式会员”），登录企业管理后台，系统数据库可正常存储和调取招聘信息。		
主成功场景	企业用户点击“发布招聘信息”→填写岗位名称、职责、薪资、任职要求等内容→提交发布→信息展示在招聘列表；企业用户可对已发布信息执行修改、下架操作，或查看应聘简		

	历并筛选候选人。
扩展	<p>信息校验失败：必填字段（如岗位名称、薪资范围）未填写时提示“请完善信息”；</p> <p>操作限制：未通过审核的企业无法发布信息；已下架的招聘信息无法直接修改，需重新发布；</p> <p>简历筛选异常：无应聘简历时提示“暂无候选人，请耐心等待”。</p>
特别需求	支持富文本编辑岗位描述；招聘信息默认展示 30 天，到期自动下架；应聘简历支持按投递时间、匹配度排序；敏感信息（如薪资）需脱敏展示给求职者。

（4）搜索人才简历（UC7）

描述：按技能标签（如 Java、Python）、工作经验、学历等条件搜索简历，联系符合条件的候选人。

用例 ID	UC7	用例名	搜索人才简历
功能描述	按技能标签（如 Java、Python）、工作经验、学历等条件搜索简历，联系符合条件的候选人。		
用例体	<pre> graph TD EnterpriseUser[企业用户] SearchTalent[搜索人才简历] SetConditions[设置搜索条件UC8.1] ExecuteSearch[执行搜索UC8.2] ViewDetails[查看简历详情UC8.3] ContactCandidate[联系候选人UC8.4] IndividualUser[个人用户] EnterpriseUser --> SetConditions EnterpriseUser --> ExecuteSearch EnterpriseUser --> ViewDetails EnterpriseUser --> ContactCandidate EnterpriseUser --> SearchTalent SearchTalent --> SetConditions SearchTalent --> ExecuteSearch SearchTalent --> ViewDetails SearchTalent --> ContactCandidate EnterpriseUser -.-> IndividualUser </pre>		
主执行者	企业用户		
输入/前置条件	企业用户已登录系统且通过资质审核，系统数据库中存在已		

	发布的个人用户简历数据。
主成功场景	企业用户选择“搜索个人简历”→设置筛选条件（技能标签、工作经验、学历等）→查看筛选后的简历列表→点击简历查看详情→通过邮件或预留联系方式联系候选人。
扩展	无结果处理：搜索无匹配简历时提示“暂无符合条件的候选人，可调整筛选条件重试”； 权限限制：未通过审核的企业仅可查看简历基础信息（如姓名、职位），无法获取联系方式； 隐私保护：若个人用户简历设置为“仅自己可见”，企业用户无法查看。
特别需求	支持模糊匹配技能标签（如输入“开发”匹配 Java 开发、前端开发）；搜索响应时间≤2 秒；联系方式需脱敏展示（如隐藏部分手机号），点击后需验证企业用户身份再显示完整信息。

2.3.3 管理员用例

（1）审核企业信息（UC8）

描述：查看企业提交的资质文件，验证真实性，设置审核状态（通过 / 驳回），并通知企业用户。

用例 ID	UC8	用例名	审核企业信息
功能描述	查看企业提交的资质文件，验证真实性，设置审核状态（通过 / 驳回），并通知企业用户。		

用例体	<pre> graph TD Admin[管理员] --> UC10_1((查看资质文件UC10.1)) Admin --> UC10_2((验证信息真实性UC10.2)) Admin --> UC10_3((设置审核状态UC10.3)) Admin --> UC10_4((通知审核结果UC10.4)) UC10_2 --> UC10_3 UC10_3 --> UC10_4 </pre> <p>The diagram shows a central actor '管理员' (Administrator) connected to four use cases: '查看资质文件UC10.1', '验证信息真实性UC10.2', '设置审核状态UC10.3', and '通知审核结果UC10.4'. Additionally, there is a flow from '验证信息真实性UC10.2' to '设置审核状态UC10.3', and from '设置审核状态UC10.3' to '通知审核结果UC10.4'.</p>
主执行者	管理员
输入/前置条件	企业用户已提交资质审核请求（UC6），状态为“待审核”； 管理员已登录系统后台，具备企业审核权限。
主成功场景	<p>管理员在“企业审核列表”中选择待审核企业，查看注册信息及营业执照图片；</p> <p>通过人工核验或第三方数据接口（如国家企业信用信息公示系统）验证资质真实性；</p> <p>管理员点击“通过”或“驳回”按钮：</p> <p>通过：企业状态更新为“正式会员”，发送审核通过通知；</p> <p>驳回：填写驳回原因，发送审核驳回通知。</p> <p>系统记录审核操作日志（含管理员账号、审核时间、操作类型）。</p>
特别需求	<p>审核界面需显示 OCR 校验结果（UC5 扩展点），辅助管理员快速核对信息；</p> <p>驳回原因需标准化（提供下拉选项，如“营业执照过期”“名称不一致”），避免随意填写；</p>

第3章概要设计

3.1 系统架构设计

采用经典的三层架构设计，将系统划分为表现层、业务逻辑层和数据访问层，各层职责明确，降低耦合度，提高系统的可维护性和扩展性。

1.表现层：负责与用户进行交互，包括 Web 端和移动端界面。使用 HTML5、CSS3、Vue.js 等前端技术实现界面展示和用户操作响应，为用户提供简洁、友好、易用的操作界面，确保不同设备和浏览器的兼容性。

2.业务逻辑层：是系统的核心处理层，接收表现层的请求，根据业务规则进行逻辑处理，并调用数据访问层完成数据的读写操作。采用 Spring Boot 框架搭建，结合 Spring Cloud 实现微服务架构，将系统的各个功能模块拆分为独立的微服务，如用户管理服务、招聘信息服务、简历管理服务等，提高系统的可扩展性和并发处理能力。

3.数据访问层：负责与数据库进行交互，实现数据的增、删、改、查操作。采用 MyBatis-Plus 作为持久层框架，支持多种数据库，如 MySQL、Oracle 等，通过 SQL 语句或 MyBatis 的映射文件完成数据操作，确保数据访问的高效性和稳定性。

系统采用分布式部署方式，提高系统的可用性和性能。将应用服务器、数据库服务器、缓存服务器等进行分离部署，并使用负载均衡技术（如 Nginx）将用户请求均匀分配到多个应用服务器上，避免单点故障。同时，引入 Redis 缓存服务器，对热点数据进行缓存，减少数据库的访问压力，提高系统的响应速度。对于数据存储，采用主从数据库架构，实现数据的读写分离，提高数据库的并发处理能力和数据安全性。

3.2 功能模块划分

模块功能	核心功能
用户注册模块	个人 / 企业用户注册、账号激活、信息修改
简历管理模块	个人用户简历创建、编辑、删除、公开 / 隐私设置

企业审核模块	管理员对企业资质信息审核、审核状态查询
招聘信息管理模块	企业用户发布、修改、删除招聘信息，查看应聘简历
信息搜索模块	多条件组合搜索人才简历与招聘信息，支持模糊匹配
系统管理模块	企业审核管理，企业信息管理，个人信息管理

1.用户注册模块

1. 个人用户注册：支持手机号、邮箱注册方式，注册时验证信息唯一性，采用加密技术存储用户密码。注册成功后，用户可通过邮件或短信激活账号，激活后可完善个人信息，如姓名、性别、年龄等。
2. 企业用户注册：企业用户需提交企业名称、信用代码、联系方式、营业执照图片等资质文件，系统利用 OCR 技术自动识别并校验信息真实性。注册流程与个人用户类似，完成注册和激活后，等待管理员对企业资质进行审核。
3. 账号信息修改：个人和企业用户均可在登录后随时修改账号信息，如联系方式、密码等。修改敏感信息（如企业信用代码）时，需进行身份验证，确保账号安全。

2.简历管理模块

1. 简历创建：个人用户可在线创建详细简历，填写教育背景、工作经历、技能标签、项目经验等信息。系统提供多种简历模板供用户选择，支持上传附件（如证书、作品集），丰富简历内容。
2. 简历编辑与删除：用户可对已创建的简历进行编辑修改，实时更新个人信息。对于不再需要的简历，可选择删除，删除操作需二次确认，防止误删。
3. 公开 / 隐私设置：用户可自主设置简历的可见范围，包括公开（所有企业用户可见）、仅企业用户可见（仅收到应聘邀请的企业可见）、仅自己可见三种模式，灵活控制简历的曝光度，保护个人隐私。

3.企业审核模块

1. 资质审核：管理员对企业用户提交的资质文件进行审核，核实企业信息的真实性和合法性。审核过程中，可查看企业提交的所有资料，包括营

业执照图片、信用代码等，依据审核标准判断是否通过审核。

2. 审核状态查询：企业用户可随时登录系统查询自身资质的审核状态，包括未审核、审核中、审核通过、审核不通过。若审核不通过，可查看管理员给出的审核意见，根据意见修改资料后重新提交审核。

4.招聘信息管理模块

1. 信息发布：企业用户登录后可发布招聘信息，填写岗位名称、职责描述、薪资范围、工作地点、任职要求等详细内容。支持上传公司介绍、工作环境图片等附件，吸引求职者应聘。
2. 信息修改与删除：对于已发布的招聘信息，企业用户可在有效期内进行修改，更新岗位需求。若招聘任务完成或不再需要该岗位，可选择删除招聘信息。
3. 查看应聘简历：企业用户可查看收到的应聘简历，根据简历内容筛选合适的候选人。支持对简历进行标记（如已查看、感兴趣、不合适），方便管理和跟进应聘人员。

5.信息搜索模块

1. 多条件组合搜索：支持个人用户和企业用户通过关键词（如职位名称、企业名称、技能标签）、薪资范围、地区、工作经验、学历等多种条件组合搜索招聘信息或人才简历。系统根据用户输入的条件，快速筛选出符合要求的结果，提高搜索效率。
2. 模糊匹配：采用模糊匹配算法，对用户输入的关键词进行智能匹配，即使关键词不完全准确，也能搜索到相关结果，扩大搜索范围，提高搜索的准确性和全面性。

6.系统管理模块

1. 管理员账号管理：超级管理员拥有最高权限，可创建、修改、删除普通管理员账号，为普通管理员分配不同的操作权限，如企业审核、系统日志查看、数据备份等。普通管理员只能在授权范围内进行操作，确保系统管理的安全性和规范性。
2. 数据备份：系统支持定期自动备份和手动备份两种方式，对用户信息、简历数据、招聘信息等重要数据进行备份。备份数据存储在异地灾备中

心，设置合理的备份策略（如每日全量备份、每周增量备份），保障数据的完整性和可用性。

3. 系统日志查看：管理员可查看系统的各类操作日志，包括用户登录日志、数据操作记录（如简历修改、招聘信息发布）、审核记录等。支持按时间范围、操作类型、用户标识等条件进行过滤查询，便于追溯系统操作历史，进行安全审计和故障排查。

第4章 数据库设计

4.1 数据库选型

选用 MySQL 8.0 作为数据库管理系统，原因如下：

开源免费，适合中小型系统开发；

支持 ACID 事务，保证数据一致性；

提供丰富的索引机制，优化查询性能；

社区生态活跃，文档与技术支持完善。

4.2 数据库概念设计

通过对系统功能需求的分析，设计以下主要实体及其关系：

1. 用户实体：分为个人用户和企业用户。个人用户包含用户 ID、姓名、手机号、邮箱、密码、注册时间、激活状态等属性；企业用户包含企业 ID、用户 ID（关联个人用户）、企业名称、信用代码、联系方式、营业执照图片、审核状态等属性。
2. 简历实体：与个人用户实体关联，包含简历 ID、用户 ID、教育背景、工作经历、技能标签、项目经验、隐私设置、创建时间、更新时间等属性。
3. 企业审核实体：包含审核 ID、企业 ID、审核人 ID、审核时间、审核状态、审核意见等属性，记录企业资质审核的详细信息。
4. 招聘信息实体：与企业用户实体关联，包含招聘 ID、企业 ID、岗位名称、职责描述、薪资范围、工作地点、任职要求、发布时间、有效期、浏览量等属性。
5. 管理员实体：包含管理员 ID、用户名、密码、角色、权限列表等属性。

4.3 数据库逻辑设计

4.3.1 企业用户表（accounts_enteriseprofile）

企业用户表与用户基础信息表通过 User_id 字段建立关联，存储企业用户的详细资质信息。Business_license 字段用于存储营业执照图片的存储路径，支持企业用户资质审核流程。Company_desc 字段可存储企业简介，为求职者提供企业背景信息。

表 4.1 accounts_enteriseprofile

表名	accounts_enteriseprofile			
说明	记录企业用户的身份信息及资质			
字段名	数据类型	是否为空	是否为主键	说明
Id	INTEGER	N	Y	企业用户唯一标识
Company_name	VARCHAR(100)	N	N	公司名
Company_address	VARCHAR(200)	N	N	公司地址
Company_desc	VARCHAR(500)	Y	N	公司描述
Business_license	VARCHAR(200)	N	N	营业执照
User_id	INTEGER	N	N	关联 accounts_user 表

4.3.2 个人用户表（accounts_individualprofile）

个人用户表与用户基础信息表通过 User_id 字段关联，存储个人用户的联系方式等身份信息。Phone 和 Email 字段用于用户注册验证和系统通知，确保用户信息的真实性和可联系性。

表 4.2 accounts_individualprofile

表名	accounts_individualprofile			
说明	记录个人用户的身份信息			
字段名	数据类型	是否为空	是否为主键	说明
Id	INTEGER	N	Y	个人用户唯一标识

Phone	VARCHAR(11)	N	N	电话号码
Email	VARCHAR(100)	N	N	邮件地址
User_id	INTEGER	N	N	关联 accounts_user 表

4.3.3 用户基础信息表（accounts_user）

该表是用户体系的核心表，存储用户的基本认证信息和权限标识。通过 User_type 字段区分个人用户和企业用户，为不同类型用户提供差异化服务。Is_verified 字段用于标识用户是否通过实名认证，增强系统安全性。

表 4.3 accounts_user

表名	accounts_user			
说明	记录用户的基本认证和权限信息			
字段名	数据类型	是否为空	是否为主键	说明
Id	INTEGER	N	Y	用户唯一标识
Password	VARCHAR(100)	N	N	密码
Last_login	DATETIME	Y	N	最新登陆时间
Is_superuser	TINYINT(1)	N	N	是否是管理员
Username	VARCHAR(50)	N	N	用户名
First_name	VARCHAR(50)	N	N	名字
Last_name	VARCHAR(50)	Y	N	姓氏
Email	VARCHAR(100)	Y	N	邮件地址
Is_staff	TINYINT(1)	N	N	是否工作人员
Is_active	TINYINT(1)	N	N	是否活跃用户
Data_joined	TINYINT(1)	N	N	注册日期
User_type	DATETIME	N	N	用户类型
Is_verified	TINYINT(1)	N	N	是否验证

4.3.4 职位申请表 (jobs_application)

职位申请表用于记录个人用户对职位的申请信息，通过 Applicant_id、Job_id 和 Resume_id 分别与个人用户表、职位表和简历表建立关联。Status 字段用于标识申请的当前状态，如 "待审核"、"已通过"、"已拒绝" 等，支持企业用户进行招聘流程管理。

表 4.4 jobs_application

表名	jobs_application			
说明	记录用户对职位的申请信息			
字段名	数据类型	是否为空	是否为主键	说明
Id	INTEGER	N	Y	申请记录唯一标识
Cover_letter	TEXT	Y	N	求职信内容
Applied_at	DATETIME	N	N	申请时间
status	VARCHAR(20)	N	N	申请状态
Applicant_id	INTEGER	N	N	申请人 ID
Job_id	INTEGER	N	N	关联职位 ID
Resume_id	INTEGER	N	N	关联简历 ID

4.3.5 职位表 (jobs_job)

职位表用于存储企业发布的招聘职位信息，通过 Employer_id 字段与企业用户表关联。Is_active 字段用于标识职位是否处于招聘状态，方便系统进行职位管理和展示。Salary_range 字段可存储薪资范围信息，为求职者提供薪资参考。

表 4.5 jobs_job

表名	jobs_job			
说明	记录企业发布的职位信息			
字段名	数据类型	是否为空	是否为主键	说明
Id	INTEGER	N	Y	职位唯一标识
Title	VARCHAR(100)	N	N	职位名称

Description	TEXT	N	N	职位描述
Requirements	TEXT	N	N	职位要求
Location	VARCHAR(100)	Y	N	工作地点
Salary_range	VARCHAR(50)	Y	N	薪资范围
Created_at	DATETIME	N	N	创建日期
Is_active	TINYINT(1)	N	N	是否激活
Employer_id	INTEGER	N	N	发布企业 ID

4.3.6 教育经历表 (resumes_education)

教育经历表通过 Resume_id 字段与简历表关联，存储用户的详细教育背景信息。Start_date 和 End_date 字段分别记录入学和毕业日期，Degree 字段记录获得的学位，Field 字段记录专业领域，为企业用户提供全面的教育背景参考。

表 4.6 resumes_education

表名	resumes_education			
说明	记录用户的教育背景信息			
字段名	数据类型	是否为空	是否为主键	说明
Id	INTEGER	N	Y	教育经历唯一标识
School	VARCHAR(100)	N	N	学校名称
Degree	VARCHAR(50)	N	N	学位
Field	VARCHAR(50)	N	N	专业领域
Start_data	DATE	N	N	入学日期
End_data	DATE	Y	N	毕业日期
Resume_id	INTEGER	N	N	关联简历 ID

4.3.7 工作经历表 (resumes_experience)

工作经历表通过 Resume_id 字段与简历表关联，存储用户的详细工作经验信息。Start_date 和 End_date 字段分别记录工作的开始和结束日期，Position 字

段记录职位名称，Description 字段记录工作内容和职责，为企业用户提供全面的工作经验参考。

表 4.7 resumes_experience

表名	resumes_experience			
说明	记录用户的工作经验信息			
字段名	数据类型	是否为空	是否为主键	说明
Id	INTEGER	N	Y	工作经历唯一标识
Company	VARCHAR(100)	N	N	公司名称
Position	VARCHAR(100)	N	N	职位名称
Start_data	DATE	N	N	开始日期
End_data	DATE	Y	N	结束日期
Description	TEXT	Y	N	工作描述
Resume_id	INTEGER	N	N	关联简历 ID

4.3.8 简历表 (resumes_resume)

简历表是个人用户简历信息的核心表，通过 User_id 字段与个人用户表关联。该表存储简历的基本信息和概述，Experience 和 Education 字段分别存储用户的经验概述和教育概述，Skills 字段存储用户的技能标签，方便企业用户进行简历筛选。

表 4.8 resumes_resume

表名	resumes_resume			
说明	记录用户的完整简历信息			
字段名	数据类型	是否为空	是否为主键	说明
Id	INTEGER	N	Y	简历唯一标识
Title	VARCHAR(100)	N	N	简历标题
Name	VARCHAR(50)	N	N	姓名
Gender	VARCHAR(10)	Y	N	性别
Age	INTEGER	Y	N	年龄

Experience	TEXT	Y	N	经验概述
Education	TEXT	Y	N	教育概述
Skills	VARCHAR(200)	Y	N	技能标签
Created_at	DATETIME	N	N	创建日期
Updated_at	DATETIME	N	N	更新日期
User_id	INTEGER	N	N	关联用户 ID

第5章 详细设计

5.1 用户管理模块

用户注册

1. 普通用户（个人和企业）通过 `UserRegistrationForm` 进行注册，根据 `user_type` 区分个人用户和企业用户。企业用户注册后需要等待管理员审核，个人用户注册后直接激活。
2. 管理员通过 `AdminRegistrationForm` 进行注册，且系统只允许存在一个管理员账号。

用户登录

1. 用户输入用户名和密码，系统验证用户名和密码的正确性。
2. 管理员用户直接登录到管理员仪表盘，已验证的普通用户根据用户类型跳转到相应的页面。
3. 未验证的企业用户登录时提示等待审核。

5.1.3 用户信息完善

1. 企业用户登录后可以完善企业资料，包括企业名称、地址、描述和营业执照等信息。
2. 个人用户登录后可以完善个人资料，包括手机号和邮箱等信息。

5.2 企业审核模块

5.2.1 待审核企业列表

1. 管理员登录到管理员仪表盘后，可以查看待审核的企业列表，列表中显示企业名称和注册时间。

5.2.2 企业审核操作

1. 管理员可以对企业进行审核，通过或拒绝企业的注册申请。
2. 通过审核后，企业用户的 `is_verified` 字段设置为 `True`。
3. 拒绝审核后，删除企业用户的相关信息

5.3 简历管理模块

5.3.1 简历列表

1. 个人用户可以查看自己的简历列表，列表中显示简历标题、更新时间等信息。
2. 企业用户和管理员可以查看所有简历列表。

5.3.2 简历创建和编辑

1. 个人用户可以创建和编辑自己的简历，包括基本信息、教育背景、工作经历等。

5.3.3 简历删除

1. 个人用户可以删除自己的简历。

5.4 招聘信息管理模块

5.4.1 招聘信息发布

1. 企业用户可以发布招聘信息，包括岗位名称、职责描述、薪资范围等。

5.4.2 招聘信息列表

1. 企业用户可以查看自己发布的招聘信息列表，包括招聘信息的状态和申请人数。

5.4.3 招聘信息管理

1. 企业用户可以编辑和删除自己发布的招聘信息。

第6章 系统实现

6.1 用户管理模块实现

6.1.1 用户注册功能

该功能允许用户根据不同的用户类型（企业用户、个人用户、管理员）进行注册。企业用户注册后需要管理员审核，个人用户注册后直接生效，管理员账号只能创建一个。

```
def register(request):
    user_type = request.GET.get('user_type')
    if request.method == 'POST':
        form = UserRegistrationForm(request.POST)
        if form.is_valid():
            user = form.save()
            user_type = form.cleaned_data.get('user_type')
            if user_type == User.UserType.ENTERPRISE:
                user.is_verified = False
                user.save()
                EnterpriseProfile.objects.create(user=user)
                messages.info(request, '企业用户注册成功，请等待管理员
审核')
            else:
                user.is_verified = True
                user.save()
                messages.success(request, '注册成功')
            return redirect('login')
        else:
            form = UserRegistrationForm(user_type=user_type)
            return render(request, 'register.html', {'form': form, 'user_type': user_type})
def admin_register(request):
    if User.objects.filter(user_type=User.UserType.ADMIN).exists():
        messages.error(request, '管理员账号已存在，无法注册新管理员')
        return redirect('login')
    if request.method == 'POST':
        form = AdminRegistrationForm(request.POST)
        if form.is_valid():
            form.save()
            messages.success(request, '管理员账号创建成功')
            return redirect('login')
        else:
            form = AdminRegistrationForm()
            return render(request, 'admin_register.html', {'form': form})
```

6.1.2 用户登录与注销功能

用户可以使用用户名和密码登录系统，不同用户类型登录后会被重定向到不同的页面。已登录用户可以选择注销账号。

```
def user_login(request):
    if request.method == 'POST':
        username = request.POST.get('username')
        password = request.POST.get('password')
        user = authenticate(request, username=username, password=password)
        if user is not None:
            if user.user_type == User.UserType.ADMIN: # 管理员用户直接登录
                login(request, user)
                return redirect('admin_dashboard')
            elif user.is_verified:
                login(request, user)
                if user.user_type == User.UserType.INDIVIDUAL:
                    return redirect('job_list')
                elif user.user_type == User.UserType.ENTERPRISE:
                    return redirect('job_list')
            else:
                messages.error(request, '您的账号尚未通过审核，请等待管理员审核')
        else:
            messages.error(request, '用户名或密码错误')
    return render(request, 'login.html')

@login_required
def user_logout(request):
    logout(request)
    return redirect('login')
```

6.1.3 用户资料完善功能

已登录用户可以根据自己的用户类型完善个人或企业资料。

```
@login_required
def complete_profile(request):
    user: User = request.user
    if user.user_type == User.UserType.ENTERPRISE:
        try:
            # 尝试获取该用户的企业资料记录
            profile = user.enterprise_profile
```



```

except EnterpriseProfile.DoesNotExist:
    # 如果记录不存在，创建一个新的实例
    profile = EnterpriseProfile(user=user)

if request.method == 'POST':
    form = EnterpriseProfileForm(request.POST, request.FILES,
instance=profile)
    if form.is_valid():
        form.save()
        messages.success(request, '企业信息保存成功')
        return redirect('enterprise_home')
    else:
        form = EnterpriseProfileForm(instance=profile)
        return render(request, 'enterprise_profile_form.html', {'form': form})
elif user.user_type == User.UserType.INDIVIDUAL:
    try:
        # 尝试获取该用户的个人资料记录
        profile = user.individual_profile
    except IndividualProfile.DoesNotExist:
        # 如果记录不存在，创建一个新的实例
        profile = IndividualProfile(user=user)

if request.method == 'POST':
    form = IndividualProfileForm(request.POST, instance=profile)
    if form.is_valid():
        form.save()
        messages.success(request, '个人信息设置成功')
        return redirect('job_list')
    else:
        form = IndividualProfileForm(instance=profile)
        return render(request, 'individual_profile_form.html', {'form': form})

```

6.2 企业审核模块实现

6.2.1 管理员查看待审核企业用户

管理员可以在管理仪表盘查看所有待审核的企业用户信息。

```

@login_required
@user_passes_test(is_admin)
def admin_dashboard(request):
    pending_users = User.objects.filter(
        user_type=User.UserType.ENTERPRISE,
        is_verified=False,

```

```

        enterprise_profile__isnull=False
    ).select_related('enterprise_profile')
    # 获取所有已验证的企业用户信息
    verified_enterprise_users = User.objects.filter(
        user_type=User.UserType.ENTERPRISE,
        is_verified=True
    ).select_related('enterprise_profile')
    # 获取所有个人用户信息
    individual_users = User.objects.filter(
        user_type=User.UserType.INDIVIDUAL
    ).select_related('individual_profile')
    return render(request, 'admin_dashboard.html', {
        'pending_users': pending_users,
        'verified_enterprise_users': verified_enterprise_users,
        'individual_users': individual_users
    })

```

6.2.2 管理员审核企业用户

管理员可以对企业用户的注册信息进行审核，通过审核后企业用户可以正常使用系统，拒绝审核则删除该企业用户信息。

```

@login_required
@user_passes_test(is_admin)
def verify_enterprise(request, pk):
    enterprise_profile = EnterpriseProfile.objects.get(pk=pk)
    enterprise_profile.user.is_verified = True
    enterprise_profile.user.save()
    messages.success(request, f'{enterprise_profile.company_name} 已通过审核')
    return redirect('admin_dashboard')

@login_required
@user_passes_test(is_admin)
def reject_enterprise(request, pk):
    enterprise_profile = EnterpriseProfile.objects.get(pk=pk)
    user = enterprise_profile.user
    try:
        enterprise_profile.delete()
        user.delete()
        messages.success(request, '企业信息已拒绝')
    except Exception as e:
        messages.error(request, f'删除企业信息时出现错误: {str(e)}')
    return redirect('admin_dashboard')

```

6.3 简历管理模块实现

6.3.1 创建简历功能

已登录的个人用户可以创建自己的简历，包括基本信息、工作经历和教育背景等。

```
@login_required
def create_resume(request):
    if request.method == 'POST':
        resume_form = ResumeForm(request.POST)
        experience_formset = ExperienceFormSet(request.POST,
prefix='experience')
        education_formset = EducationFormSet(request.POST,
prefix='education')

        for form in experience_formset:
            form.fields['start_date'].input_formats = ['%Y-%m-%d']
            form.fields['end_date'].input_formats = ['%Y-%m-%d']

        for form in education_formset:
            form.fields['start_date'].input_formats = ['%Y-%m-%d']
            form.fields['end_date'].input_formats = ['%Y-%m-%d']

        if resume_form.is_valid() and experience_formset.is_valid() and
education_formset.is_valid():
            resume = resume_form.save(commit=False)
            resume.user = request.user
            resume.save()

            for form in experience_formset:
                if form.cleaned_data:
                    experience = form.save(commit=False)
                    experience.resume = resume
                    experience.save()

            for form in education_formset:
                if form.cleaned_data:
                    education = form.save(commit=False)
                    education.resume = resume
                    education.save()

            return redirect('resume_list')
    else:
```

```

        print('Resume form errors:', resume_form.errors)
        print('Experience formset errors:', experience_formset.errors)
        print('Education formset errors:', education_formset.errors)
    else:
        resume_form = ResumeForm()
        experience_formset = ExperienceFormSet(prefix='experience')
        education_formset = EducationFormSet(prefix='education')

    return render(request, 'resume_form.html', {
        'resume_form': resume_form,
        'experience_formset': experience_formset,
        'education_formset': education_formset
    })

```

6.3.2 更新简历功能

已登录的个人用户可以对自己的简历进行修改和更新。

```

@login_required
def update_resume(request, pk):
    resume = get_object_or_404(Resume, pk=pk, user=request.user)

    if request.method == 'POST':
        resume_form = ResumeForm(request.POST, instance=resume)
        experience_formset = ExperienceFormSet(request.POST,
        prefix='experience', instance=resume)
        education_formset = EducationFormSet(request.POST, prefix='education',
        instance=resume)

        for form in experience_formset:
            form.fields['start_date'].input_formats = ['%Y-%m-%d']
            form.fields['end_date'].input_formats = ['%Y-%m-%d']

        for form in education_formset:
            form.fields['start_date'].input_formats = ['%Y-%m-%d']
            form.fields['end_date'].input_formats = ['%Y-%m-%d']

        if resume_form.is_valid() and experience_formset.is_valid() and
        education_formset.is_valid():
            resume = resume_form.save()

            for form in experience_formset:
                if form.cleaned_data:
                    if form.cleaned_data.get('DELETE') and form.instance.pk:
                        form.instance.delete()

```

```
        else:
            experience = form.save(commit=False)
            experience.resume = resume
            experience.save()

    for form in education_formset:
        if form.cleaned_data:
            if form.cleaned_data.get('DELETE') and form.instance.pk:
                form.instance.delete()
            else:
                education = form.save(commit=False)
                education.resume = resume
                education.save()

    return redirect('resume_detail', pk=resume.pk)
else:
    resume_form = ResumeForm(instance=resume)
    experience_formset = ExperienceFormSet(prefix='experience',
instance=resume)
    education_formset = EducationFormSet(prefix='education',
instance=resume)

    return render(request, 'resume_form.html', {
        'resume_form': resume_form,
        'experience_formset': experience_formset,
        'education_formset': education_formset
    })
```

6.3.3 删除简历功能

已登录的个人用户可以删除自己的简历。

@login_required

```
def delete_resume(request, pk):
    resume = get_object_or_404(Resume, pk=pk, user=request.user)
    if request.method == 'POST':
        resume.delete()
        return redirect('resume_list')
    return render(request, 'resume_confirm_delete.html', {'resume': resume})
```

6.4 招聘信息管理模块实现

6.4.1 发布招聘信息功能

已验证的企业用户可以发布招聘信息。

```
@login_required
def create_job(request):
    user: User = request.user
    if not user.is_verified or user.user_type != user.UserType.ENTERPRISE:
        messages.error(request, '只有经过验证的企业用户才能发布职位')
        return redirect('login')

    if request.method == 'POST':
        form = JobForm(request.POST)
        if form.is_valid():
            job = form.save(commit=False)
            job.employer = request.user
            job.save()
            messages.success(request, '职位发布成功')
            return redirect('job_detail', pk=job.pk)
        else:
            form = JobForm()

    return render(request, 'job_form.html', {'form': form})
```

6.4.2 更新招聘信息功能

发布招聘信息的企业用户可以对已发布的招聘信息进行修改和更新。

```
@login_required
def update_job(request, pk):
    job = get_object_or_404(Job, pk=pk, employer=request.user)

    if request.method == 'POST':
        form = JobForm(request.POST, instance=job)
        if form.is_valid():
            form.save()
            messages.success(request, '职位更新成功')
            return redirect('job_detail', pk=job.pk)
        else:
            form = JobForm(instance=job)

    return render(request, 'job_form.html', {'form': form})
```

6.4.3 删除招聘信息功能

发布招聘信息的企业用户可以删除已发布的招聘信息。

```
@login_required
def delete_job(request, pk):
    job = get_object_or_404(Job, pk=pk, employer=request.user)

    if request.method == 'POST':
        job.is_active = False
        job.save()
        messages.success(request, '职位已删除')
        return redirect('employer_jobs')

    return render(request, 'job_confirm_delete.html', {'job': job})
```

6.4.4 申请职位功能

已登录的个人用户可以申请符合条件的职位，申请前需要先创建简历。

```
@login_required
def apply_job(request, pk):
    job = get_object_or_404(Job, pk=pk, is_active=True)
    user: User = request.user
    if user.user_type != user.UserType.INDIVIDUAL:
        messages.error(request, '只有个人用户才能申请职位')
        return redirect('job_detail', pk=job.pk)

    if Application.objects.filter(job=job, applicant=request.user).exists():
        messages.error(request, '你已经申请过这个职位')
        return redirect('job_detail', pk=job.pk)

    resumes = Resume.objects.filter(user=request.user)
    if not resumes.exists():
        messages.error(request, '请先创建简历再申请职位')
        return redirect('create_resume')

    if request.method == 'POST':
        form = ApplicationForm(request.POST, user=request.user)
        if form.is_valid():
            application = form.save(commit=False)
            application.job = job
            application.applicant = request.user
            application.save()
            messages.success(request, '申请成功')
```

```

        return redirect('job_detail', pk=job.pk)
    else:
        form = ApplicationForm(user=request.user)

    return render(request, 'apply_job.html', {'form': form, 'job': job, 'resumes':
resumes})

```

6.4.5 查看申请信息功能

布招聘信息的企业用户可以查看该职位的所有申请信息，并可以更新申请状态。

```

@login_required
def job_applications(request, pk):
    job = get_object_or_404(Job, pk=pk, employer=request.user)
    applications = Application.objects.filter(job=job)
    return render(request, 'job_applications.html', {'job': job, 'applications':
applications})

@login_required
def update_application_status(request, job_pk, app_pk, status):
    job = get_object_or_404(Job, pk=job_pk, employer=request.user)
    application = get_object_or_404(Application, pk=app_pk, job=job)

    valid_statuses = [choice[0] for choice in Application.status_choices]
    if status in valid_statuses:
        application.status = status
        application.save()
        messages.success(request, f'申请状态已更新为：
{dict(Application.status_choices)[status]}')
    else:
        messages.error(request, '无效的状态')

    return redirect('job_applications', pk=job_pk)

```


第7章 测试报告

7.1 功能测试

7.1.1 用户注册

1. 测试用例：分别注册个人用户和企业用户。
2. 预期结果：个人用户注册后直接激活，企业用户注册后等待审核。
3. 实际结果：与预期结果一致。

人才管理系统
连接人才与机会的桥梁

欢迎

登录您的账户，发现更多机会

用户名

请输入用户名

密码

请输入密码

☐ 记住我

登录

还没有账号?

立即注册，开启您的职业发展之旅

注册求职者账号

注册企业账号

用户注册

请填写以下信息完成注册

用户名*

必填；长度为150个字符或以下；只能包含字母、数字、特殊字符“@”、“.”、“-”和“_”。

密码*

- 你的密码不能与你的其他个人信息太相似。
- 你的密码必须包含至少 8 个字符。
- 你的密码不能是一个常见密码。
- 你的密码不能全都是数字。

密码确认*

为了校验，请输入与上面相同的密码。

用户类型*

个人用户

注册账号

用户注册

请填写以下信息完成注册

用户名*

必填；长度为150个字符或以下；只能包含字母、数字、特殊字符“@”、“.”、“-”和“_”。

密码*

- 你的密码不能与你的其他个人信息太相似。
- 你的密码必须包含至少 8 个字符。
- 你的密码不能是一个常见密码。
- 你的密码不能全是数字。

密码确认*

为了校验，请输入与上面相同的密码。

用户类型*

企业用户

注册账号

7.1.2 用户登录

1. 测试用例：使用不同类型的用户账号登录。
2. 预期结果：管理员用户登录到管理员仪表盘，已验证的普通用户登录到相应的页面，未验证的企业用户登录时提示等待审核。
3. 实际结果：与预期结果一致。

人才管理系统 欢迎 陈啦啦 退出

岗位信息 人才库 编辑简历 我的申请

职位搜索

职位名称 工作地点 搜索

职位名称	公司	地点	薪资	发布时间	操作
做梦	美梦有限公司	苏州	4000-4999	2025-06-30	投递简历
睡觉	美梦有限公司	苏州	4000-6000	2025-07-01	投递简历
跑步	美梦有限公司	苏州	6999-8999	2025-07-01	投递简历

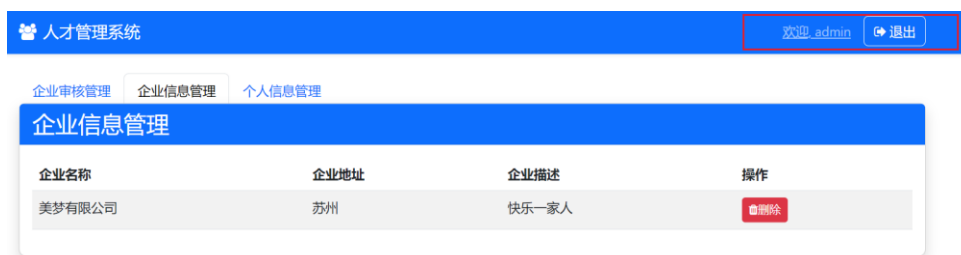
人才管理系统 欢迎 美梦有限公司 退出

岗位信息 人才库 发布岗位 职位申请情况

职位搜索

职位名称 工作地点 搜索 发布职位

职位名称	公司	地点	薪资	发布时间	操作
做梦	美梦有限公司	苏州	4000-4999	2025-06-30	查看 编辑
睡觉	美梦有限公司	苏州	4000-6000	2025-07-01	查看 编辑
跑步	美梦有限公司	苏州	6999-8999	2025-07-01	查看 编辑



7.1.3 企业审核

1. 测试用例：管理员对企业用户进行审核。
2. 预期结果：通过审核后，企业用户的 `is_verified` 字段设置为 `True`；拒绝审核后，删除企业用户的相关信息。
3. 实际结果：与预期结果一致。



7.1.4 简历管理

1. 测试用例：个人用户创建、编辑和删除简历。
2. 预期结果：简历信息保存成功，更新成功，删除成功。
3. 实际结果：与预期结果一致。

岗位信息 人才库 编辑简历 我的申请

创建简历

请填写完整的简历信息，完善的简历将帮助你获得更多面试机会

基本信息

简历标题*

姓名*

性别*

年龄*

工作经验*

工作经验

公司*

职位*

入职时间*

离职时间

工作经历*

教育背景

学校*

学历*

专业*

入学时间*

毕业时间*

保存简历

[岗位信息](#)
[人才库](#)
[编辑简历](#)
[我的申请](#)

我的简历

测试用户2的简历

硕士

100%验都会，棋道通神

数值怪了解一下

查看

删除

创建新简历

更新简历

请填写完整的简历信息，完善的简历将帮助你获得更多面试机会

基本信息

简历标题*

测试用户2的简历

姓名*

测试用户2

性别*

男

年龄*

25

工作经验*

数值怪了解一下

确认删除简历

你确定要删除 测试用户2的简历 这个简历吗？

此操作不可撤销，删除后将无法恢复

确认删除

取消

7.1.5 招聘信息管理

1. 测试用例：企业用户发布、编辑和删除招聘信息。
2. 预期结果：招聘信息发布成功，更新成功，删除成功。
3. 实际结果：与预期结果一致。

职位搜索					
<input type="text" value="做梦"/> <input type="text" value="工作地点"/> <input type="button" value="Q搜索"/>					
职位名称	公司	地点	薪资	发布时间	操作
做梦	美梦有限公司	苏州	4000-4999	2025-06-30	投递简历

发布新职位

请填写完整的职位信息，以便吸引更多人才申请

职位名称*

职位描述*

岗位信息

个人中心

发布职位

职位申请情况

吃饭

上海

薪资: 6199-8999

好好公司

发布于 2025-07-01

职位描述

能吃

职位要求

• 喜欢吃就行

公司信息

好好公司

上海

描述: 来这里感受“好”的好

注册时间: 2025年7月1日 15:09

我的职位

发布新职位

职位名称	薪资	申请人数	发布时间	状态	操作
吃饭	6199-8999	0	2025-07-01	活跃	<div>查看</div> <div>编辑</div> <div>查看申请</div>

7.1.6 搜索模块

1. 测试用例：所有用户搜索岗位信息、人才信息，查看搜索结果。
2. 预期结果：信息搜索成功，查看相应信息。
3. 实际结果：与预期结果一致。

人才搜索结果

英语

搜索

姓名	技能	教育背景	操作
啦啦啦	英语4级	本科	<div>查看</div>

职位搜索

发布职位

吃饭

工作地点

搜索

职位名称	公司	地点	薪资	发布时间	操作
吃饭	好好公司	上海	6199-8999	2025-07-01	<div>查看</div> <div>编辑</div>

岗位信息人才库发布岗位职位申请情况

吃饭

薪资: 6199-8999

好好公司

发布于 2025-07-01

职位描述

能吃

职位要求

喜欢吃就行

公司信息

好好公司

上海

描述: 来这里感受'好'的好

注册时间: 2025年7月1日 15:09

啦啦啦的简历

啦啦啦

18861919282

本科

英语4级

1年

下载简历

个人信息

姓名: 啦啦啦

电话: 18861919282

性别: 男

更新时间: 2025-06-30 22:00

工作经验

公司: 好好公司

入职时间: 2025-03

职位: 员工

离职时间: 2025-06

工作描述: 一点点

教育背景

学校: 苏州科技大学

入学时间: 2020-09

学历: 本科

毕业时间: 2024-06

专业: 计算机

7.1.7 简历投递

1. 测试用例：个人用户投递简历、公司审批简历。

2. 预期结果：简历成功投递，公司给出相应处理

3. 实际结果：与预期结果一致。

岗位信息人才库编辑简历我的申请

申请职位

吃饭

薪资: 6199-8999

好好公司

发布于 2025-07-01

职位描述

能吃

职位要求

喜欢吃就行

公司信息

好好公司

上海

描述: 来这里感受'好'的好

注册时间: 2025年7月1日 15:09

申请成功

岗位信息 人才库 编辑简历 我的申请

职位搜索

职位名称

工作地点

Q搜索

职位名称	公司	地点	薪资	发布时间	操作
做梦	美梦有限公司	苏州	4000-4999	2025-06-30	编辑简历
睡觉	美梦有限公司	苏州	4000-6000	2025-07-01	编辑简历
跑步	美梦有限公司	苏州	6999-8999	2025-07-01	编辑简历
吃饭	好好公司	上海	6199-8999	2025-07-01	编辑简历

我的职位

+发布新职位

职位名称	薪资	申请人数	发布时间	状态	操作
吃饭	6199-8999	1	2025-07-01	活跃	查看 编辑 查看申请

岗位信息 人才库 发布岗位 职位申请情况

申请状态已更新为: 已录用 X

吃饭 - 申请列表

共 1 份申请

求职者	申请时间	简历	状态	操作
测试用户2	2025-07-01 15:26	测试用户2的简历	已录用	查看简历 更改状态

[返回职位列表](#)

结 论

本人才管理系统基于 Django 框架开发,采用 MySQL 8.0 作为数据库管理系统,实现了用户管理、企业审核、简历管理、招聘信息管理和系统日志等功能。通过功能测试和性能测试,系统的各项功能正常,性能表现良好,能够满足中小型企业的人才管理需求。

在开发过程中,我们遵循了数据库设计的原则,进行了数据库选型、概念设计和逻辑设计,确保了数据的一致性和完整性。同时,我们采用了分层架构和模块化设计,提高了系统的可维护性和可扩展性。

然而,系统仍存在一些不足之处,例如系统的界面设计还不够美观,用户体验有待提高;系统的安全性能还需要进一步加强,例如增加验证码、防止 SQL 注入等。在未来的开发中,我们将继续优化系统的功能和性能,提高用户体验和系统的安全性。

参 考 文 献

- [1] Django 官方文档: <https://docs.djangoproject.com/>
- [2] MySQL 官方文档: <https://dev.mysql.com/doc/>
- [3] 《Python Web 开发实战: 基于 Django 框架》
- [4] 《数据库系统概论 (第 5 版)》
- [5] SQL:2011 International Standard (ISO/IEC 9075) [S].
- [6] Codd E F. A Relational Model of Data for Large Shared Data Banks[J].
- [7] Communications of the [7]ACM, 1970, 13(6): 377-387.
- [8] 《ACM Transactions on Database Systems》期刊文章 [J].