软件架构文档 (SAD)

说明：

SAD，全称为Software Architecture Document（软件架构文档），是软件开发过程中至关重要的一个文档，它详细描述了软件系统的总体结构、组成组件、它们之间的关系、接口以及指导设计和实施的原则。编写SAD的目的是为了让所有项目干系人对软件的高层设计有共同的理解，从而促进团队间的沟通，确保开发工作的一致性和高效性。

**目录**

[1 文档简介 3](#_Toc14182)

[1.1文档目的及范围 3](#_Toc17536)

[1.2 参考资料 3](#_Toc25500)

[2 架构视图阅读指南 3](#_Toc5244)

[3 架构设计目标 4](#_Toc29427)

[3.1 关键功能 4](#_Toc28988)

[3.2 关键质量属性 4](#_Toc8843)

[3.2.1安全性 4](#_Toc24205)

[3.2.2 可用性与响应性 4](#_Toc32100)

[3.2.3 可扩展性 4](#_Toc8249)

[3.2.4 用户界面友好性 4](#_Toc4577)

[3.2.5 兼容性和性能 4](#_Toc26082)

[3.2.6 可维护性和可测试性 5](#_Toc2209)

[3.3业务需求和约束因素 5](#_Toc14100)

[4 架构设计原则 6](#_Toc20186)

[4.1 架构设计原则 6](#_Toc10392)

[4.2 备选架构设计方案及被否原因 7](#_Toc22750)

[4.2.1 备选方案A：微服务架构与单体架构的混合模式 7](#_Toc28031)

[4.2.2 备选方案B：完全基于容器化的微服务架构 8](#_Toc25969)

[4.2.3 备选方案C：基于云原生的Serverless架构 8](#_Toc10637)

[4.3 架构设计对后续工作的限制（详设，部署等） 8](#_Toc12563)

[4.3.1 细节设计约束 8](#_Toc22500)

[4.3.2 部署约束 9](#_Toc16501)

[4.3.3 后续维护与扩展 9](#_Toc28706)

[5 全局数据结构说明 9](#_Toc25516)

[5.1 常量 9](#_Toc1356)

[5.2 变量 10](#_Toc24543)

[5.3 数据结构 11](#_Toc13631)

[6 逻辑架构视图 16](#_Toc23426)

[6.1顶层逻辑架构视图 16](#_Toc13440)

[6.1.1 逻辑视图 16](#_Toc18787)

[6.1.2 数据流程视图 17](#_Toc24984)

[6.2 功能逻辑架构（分解）视图 18](#_Toc12555)

[6.2.1个人用户端功能模块 18](#_Toc25709)

[6.2.2企业用户端功能模块 20](#_Toc26437)

[6.2.3管理员端功能模块 22](#_Toc16687)

[6.2.4 功能逻辑视图 24](#_Toc8141)

[6.3 开发视图 30](#_Toc17357)

[6.4 运行架构视图 31](#_Toc18798)

[6.4.1登陆运行视图 31](#_Toc4122)

[6.4.2个人信息运行视图 32](#_Toc28697)

[6.4.3 简历模板运行视图 34](#_Toc17412)

[6.4.4招聘运行视图 36](#_Toc31935)

[6.4.5系统公告运行视图 37](#_Toc17543)

[6.4.6求职论坛运行视图 38](#_Toc10485)

[6.4.7在线客服运行视图 40](#_Toc21320)

[6.5 实现视图 41](#_Toc27813)

[6.6 物理架构视图 41](#_Toc14845)

[6.6.1物理拓扑 41](#_Toc7770)

[6.6.2软件到硬件的映射 43](#_Toc517)

[6.6.3优化部署 43](#_Toc750)

[7.关键质量属性的设计原理 44](#_Toc8677)

[7.1 性能 44](#_Toc12192)

[7.2 安全性 44](#_Toc6016)

[7.3 可用性 44](#_Toc31195)

[7.4 可维护性 44](#_Toc23880)

[7.5 可扩展性 45](#_Toc18998)

[7.6 兼容性 45](#_Toc22970)

[7.7 可测试性 45](#_Toc5238)

[7.8 用户体验 45](#_Toc9178)

**1 文档简介**

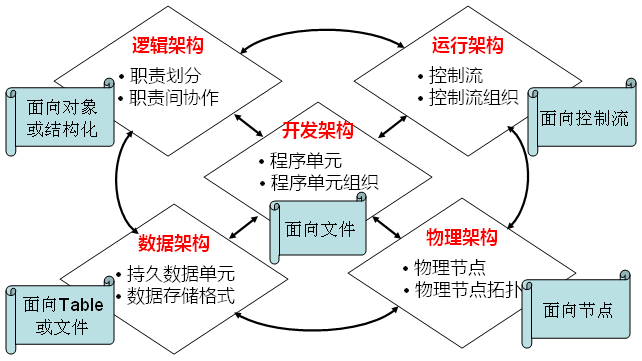
## 1.1文档目的及范围

该架构文档对校园人才招聘系统整体结构进行描述，主要模块为架构描述、数据结构说明、逻辑架构视图。其中架构描述包括架构设计的目标、原则，数据结构说明给出了本系统中所有常量变量以及数据结构，逻辑架构视图包括了顶层架构、功能架构、开发视图、物理架构视图。

## 1.2 参考资料

无

# **2 架构视图阅读指南**



本文详细给出了顶层逻辑架构视图、功能逻辑架构视图、开发视图、运行视图、实现试图和物理架构视图。

其中顶层逻辑架构视图首先介绍了我们整个项目的所有模块，包括注册中心、服务中心、中间件、储存和持续集成和容器集成技术，并给出了图表。此外这个部分还给出了整个项目的数据流程视图，介绍了项目数据的来源和去向。

功能逻辑架构视图方面，根据该项目的功能我们把它分为四个模块：个人用户端功能模块、企业用户端功能模块、管理员端功能模块和功能逻辑视图模块。分别介绍了它们的基本功能。

开发视图方面，给出了我们整个项目的开发视图。

运行架构视图方面，我们给出了登陆运行视图、个人信息运行视图、简历模板运行视图、招聘运行视图、系统公告运行视图、求职论坛运行视图、在线客服运行视图。

实现视图方面，给出了我们项目的总体实现视图，从数据库、数据层、业务层、展示层最后到总体层。

物理架构视图方面，我们从物理拓扑、软件到硬件的映射、优化部署三个方面介绍我们的物理架构。

# **3 架构设计目标**

## 3.1 关键功能

|  |  |  |  |
| --- | --- | --- | --- |
| 用户角色 | 核心功能 | 必做功能 | 独特功能 |
| 用户（应聘者+企业） | 登录（应聘者+企业）、注册 | 简历上传、个人中心（个人中心（信息查看）、我的发布、我的收藏）、后台管理 | 简历模板（点赞、踩一下、收藏）招聘会（报名）求职论坛（发布帖子）、在线客服、系统公告 |
| 管理员 | 用户管理 | 个人中心（修改、查看） | 简历模板管理、招聘会管理、报名招聘管理（审核通过报名、修改等）、我的收藏管理、系统管理（轮播图管理、系统公告、在线客服） |

## 3.2 关键质量属性

**3.2.1安全性**

这是整个平台最为基础且至关重要的质量属性。包括但不限于用户数据的加密存储和传输（如用户登录信息、简历内容等），防止数据泄露或被非法访问。同时，系统应具备防范SQL注入、XSS攻击等常见安全威胁的能力。对于在线客服系统，还需要确保通信内容的安全性。

**3.2.2 可用性与响应性**

系统应提供稳定的访问服务，保证用户在任何时间都能顺利登录、浏览信息、上传简历、参与招聘会等，且操作过程响应迅速。这要求平台具有高可用性设计，比如负载均衡、故障转移机制，以及合理的服务器资源分配，确保用户界面和后台处理的高效执行。

**3.2.3 可扩展性**

随着用户量的增长和功能的增加，系统需能够平滑地扩展，快速适应变化。这涉及到后端服务的微服务化设计、数据库的水平扩展、以及前端界面的模块化开发，确保新功能的快速集成和部署。

**3.2.4 用户界面友好性**

无论是应聘者还是企业用户，都期望有一个直观、易用的界面。这意味着设计上应注重简洁明了的布局，操作逻辑清晰，以及良好的移动端适配，确保跨平台的无缝体验。

**3.2.5 兼容性和性能**

系统需确保在不同浏览器和设备上的兼容性，同时优化前后端性能，减少加载时间，提升用户体验。特别是在简历上传、招聘会报名等可能涉及大量数据交互的功能上，要特别注意性能优化。

**3.2.6 可维护性和可测试性**

为了长期运营和不断升级，系统设计应便于维护和测试。这包括清晰的代码结构、详细的文档记录、自动化测试框架的集成，以及持续集成/持续部署(CI/CD)流程的建立。

## 3.3业务需求和约束因素

|  |  |  |  |
| --- | --- | --- | --- |
|  | 广义功能 | 质量 | 约束 |
| **业务级需求** |  |  |  |
| 用户登录/注册 | 用户身份验证与权限管理 | 高安全性、易用性 | 数据加密传输、符合GDPR等隐私法规 |
| 简历模板 | 提供多样化的简历模板 | 易用性、美观性、个性化 | 定期更新模板库、模板版权合规 |
| 招聘会 | 举办与参与招聘会 | 信息实时性、活动管理便利性 | 时间地点明确、报名截止日期设置 |
| 简历上传 | 用户简历存储与管理 | 数据安全、格式兼容性 | 简历审核机制、存储空间限制 |
| 求职论坛 | 用户交流互动平台 | 内容审核、社区活跃度 | 防止垃圾信息、用户行为规范 |
| 系统公告 | 信息推送与通知 | 及时性、覆盖度 | 多渠道通知、历史公告存档 |
| 个人中心 | 用户个性化信息展示与管理 | 个性化配置、易于导航 | 数据私密性、界面响应速度 |
| 后台管理 | 系统与内容管理 | 功能全面性、操作便捷性 | 权限分级、操作日志记录 |
| 在线客服 | 用户服务与支持 | 快速响应、解决问题能力 | 24小时服务、多语言支持 |
| **开发级需求** |  |  |  |
| 用户身份验证与管理 | 实现OAuth、JWT等认证协议 | 安全性、响应速度、扩展性 | 支持多种登录方式、遵循安全标准 |
| 简历模板设计与管理 | 前后端分离架构、模板渲染引擎 | 用户体验、兼容性、易维护性 | 版权管理、模板更新机制 |
| 招聘会管理 | 数据库存储优化、事件发布/订阅模型 | 数据一致性、实时性、可扩展性 | 数据准确性、活动状态跟踪 |
| 简历上传与审核 | 文件存储服务(S3, Azure Blob等)、格式检查 | 文件处理速度、安全性、存储优化 | 文件大小限制、类型过滤 |
| 求职论坛构建 | 内容审核机制、评论系统设计 | 性能、内容安全、互动性 | 内容过滤、防垃圾信息策略 |
| 系统公告推送 | 消息队列(RabbitMQ, Kafka)、实时推送技术 | 实时性、可扩展性、消息可靠性 | 通知渠道多样性、用户偏好设置 |
| 个人中心与信息管理 | 数据缓存(Redis、Memcached)、权限控制 | 数据一致性、页面加载速度、个性化体验 | 数据隐私保护、个性化推荐算法 |
| 发布与收藏功能 | 数据库事务处理、缓存策略 | 数据一致性、操作效率、用户体验 | 操作日志记录、收藏数量限制 |
| 后台管理系统 | RBAC权限模型、数据可视化工具 | 界面友好、操作便利性、系统稳定性 | 权限细分、操作审计 |
| 在线客服系统 | 聊天API集成、AI聊天机器人 | 响应时间、交互体验、智能程度 | 客服人员管理、工单系统集成 |

# **4 架构设计原则**

## 4.1 架构设计原则

本系统在设计时遵循的是以下六个原则：模块化(modularity)、接口(interface)、信息隐藏(information hiding)、增量式开发(incremental development)、抽象(abstraction)、通用性(generality)，希望融合使用这些原则帮助我们做出高效和健壮的设计。

|  |  |
| --- | --- |
| 原则 | 内涵 |
| 模块化(modularity) | 模块化，也称关注点分离，是一种**把系统中各不相关的部分进行分离**的原则，以便于各部分能够独立研究。其中，关注点可以是功能、数据、特征、任务、性质或想要定义或详细理解的需求以及设计的任何部分。如果该原则应用得当。每个模块都有自己唯一的目的，并且相对独立于其它模块。每个模块的理解和开发将会更简单。模块独立也使得故障的定位和系统的修改更加简单。 |
| 接口(interface) | 接口**为系统其余部分定义了该软件单元提供的服务，以及如何获取这些服务**。一个对象的接口是该对象所有公共操作以及这些操作签名的集合，指定了操作名称、参数和可能的返回值。更全面的讲，依据服务或假设，接口还必须定义该单元所必须的信息，以确保该单元能够正确工作。 |
| 信息隐藏(information hiding) | 信息隐藏的**目标是使得软件系统更加易于维护**。它以系统分解为特征。每个软件单元都封装了一个将来可以改变的独立的设计决策，然后我们根据外部可见的性质，在接口和接口规格说明的帮助下描述了各个软件单元。信息隐藏的一个好处是使得软件具有低耦合度。每个单元的接口列出了该单元提供的访问函数和需要使用的其它访问函数的集合。这个特征使得软件易于理解、维护和定位。 |
| 增量式开发(incremental development) | 使用**单元之间的依赖关系**来设计一个增量式设计开发的进度表。首先，指定单元之间的使用关系它为各个软件单元和它依赖的单元之间建立关联。将系统的这种使用关系表述成使用图，图中节点代表软件单元，有向边从使用其它单元的软件单元出发，指向被使用的单元。使用图可以帮助我们逐步确定更大的系统子集，对此我们可以增量的进行实现和测试。扇入指代使用某个软件单元的软件单元数量，扇出指代某个软件单元使用其它软件单元的数量。设计一个系统的最终目标之一是创建一个有着高扇入、低扇出的软件单元。使用夹层法消除循环。 |
| 抽象(abstraction) | 抽象是一种**忽略一些细节来关注其它细节**的模型或表示。目标不同，忽略的细节也不同。对于一个特定的模型，一个好的抽象的关键是决定哪些细节是不相关的，进而可以被忽略。抽象的性质取决于开始时我们建立这个模型的初衷，我们想交互哪些信息，或者我们想展示哪个分析过程。 |
| 通用性(generality) | 在开发软件单元时，使它尽可能的能够称为通用的软件，来加强它在将来某个系统中能够被使用的可能性。我们通过**增加软件单元使用的上下文环境的数量来开发更加通用的软件单元**，下面是几条实现规则：将特定的上下文环境信息参数化：通过把软件单元所操作的数据参数化，我们可以开发出更加通用的软件去除前置条件：使得软件在那些我们之前假设不可能发生的条件下工作简化后置条件：把一个复杂的软件单元分解成若干个具有不同后置条件的单元，再将他们集中起来解决原来需要解决的问题，或者当只需要其中一部分后置条件时单独使用。 |

## 4.2 备选架构设计方案及被否原因

在设计智慧人才招聘系统的过程中，我们探讨了多种架构方案以寻找最合适的系统设计路径。以下是几个主要备选架构方案及其未被采纳的原因：

**4.2.1 备选方案A：微服务架构与单体架构的混合模式**

**（1）描述**

此方案提议采用微服务架构处理核心业务功能（如简历管理、职位发布等），同时保留部分非核心功能（如系统公告、在线客服）在单体架构中，旨在平衡开发速度与系统解耦。

**（2）未采纳原因**

经过评估，发现混合模式可能导致系统间通讯复杂度上升，运维难度增加，且难以充分利用微服务带来的扩展优势。此外，随着系统的发展，混合架构可能会逐渐暴露出难以维护和扩展的问题。

**4.2.2 备选方案B：完全基于容器化的微服务架构**

**（1）描述**

提议所有系统功能均以微服务形式部署，每个服务独立运行在Docker容器中，通过Kubernetes进行集群管理。

**（2）未采纳原因**

尽管容器化和微服务能够带来高度的灵活性与可扩展性，但对于本项目初期，考虑到成本投入、团队对技术栈的熟悉度以及系统初期规模，此方案被认为过度设计，增加了不必要的复杂性和初期部署成本。

**4.2.3 备选方案C：基于云原生的Serverless架构**

**（1）描述**

该方案主张利用AWS Lambda或Azure Functions等Serverless平台，让系统功能按需执行，无需直接管理服务器。

**（2）未采纳原因**

虽然Serverless架构极大地降低了运维成本，提高了系统的可伸缩性，但鉴于人才招聘系统对数据处理的即时性要求较高，Serverless架构可能引入额外的延迟，且在处理复杂业务逻辑和状态管理方面存在局限性，最终因性能和成本效益比考量而被放弃。

## 4.3 架构设计对后续工作的限制（详设，部署等）

**4.3.1 细节设计约束**

**（1）模块化开发约束**

详尽设计阶段应严格遵循模块化原则，确保每个模块职责单一，接口清晰。模块设计需遵循低耦合、高内聚的原则，使得模块间依赖关系最小化，便于独立开发、测试和维护。任何模块的修改或扩展不应影响到其他模块的正常运行。

**（2）技术栈和标准遵守**

在详细设计中，开发团队必须遵循已选定的技术栈（如Spring Boot, MySQL, Tomcat等）和编码规范（如Google Java Style Guide），确保代码质量和兼容性。任何新引入的技术或库需经过架构评审，确保与现有技术栈的兼容性和长期支持性。

**（3）接口标准化**

接口设计应严格遵守RESTful API设计原则，确保数据交换格式（如JSON, XML）的一致性。对外部系统集成的接口需按照已定义的协议规范实现，保证数据交互的稳定性和安全性。

**4.3.2 部署约束**

**（1）环境一致性**

系统部署需支持多环境（如开发、测试、生产环境）的一致性配置，确保在不同环境间切换时，系统行为和性能保持稳定。配置管理应通过配置文件或环境变量实现，避免硬编码配置。

**（2）部署自动化与容器化**

推荐使用容器化技术（如Docker）和容器编排工具（如Kubernetes）进行部署，以实现快速、可复制的部署流程。部署脚本和配置应纳入版本控制系统，确保部署过程的可追溯性和一致性。

**（3）安全与合规性**

部署过程中，必须实施数据加密、访问控制等安全措施，确保符合既定的安全性需求。系统部署前需通过安全审计，确保没有安全漏洞暴露。

**4.3.3 后续维护与扩展**

**（1）可扩展性预留**

在设计阶段，应为系统留有合理的扩展接口和机制，如使用微服务架构的，需考虑服务的动态扩展和负载均衡。系统设计应避免单点故障，支持水平扩展以应对未来业务增长。

**（2）版本兼容与升级路径**

详细设计需考虑未来版本的兼容性，包括数据迁移策略和向后兼容性设计，确保系统升级时不影响现有功能和服务的连续性。

**（3）监控与日志**

系统需内置监控与日志记录机制，为运维提供实时的系统健康状况反馈和故障追踪能力。日志应遵循统一格式，并支持集中式日志管理系统。

# **5 全局数据结构说明**

本章说明本人才招聘程序系统中使用的全局数据常量、变量和数据结构。

## 5.1 常量

常量是在程序执行期间其值保持不变的数据项，本系统中使用的部分关键常量说明所下表所示。

**表 1 常量列表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 常量名称 | 数据类型 | 默认值 | 所在目录 | 功能说明 |
| DEFAULT\_CONTEXT\_CLASS | String | "org.springframework.context.annotation.AnnotationConfigApplicationContext" | E:\rencaizhaopinxitong\maven-3.3.9\repository\org\springframework\boot\spring-boot\2.2.2.RELEASE\spring-boot-2.2.2.RELEASE.jar!\org\springframework\boot\SpringApplication.class | Spring Boot应用程序的默认上下文类，用于基于Java配置的应用上下文。 |
| DEFAULT\_SERVLET\_WEB\_CONTEXT\_CLASS | String | "org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationContext" | Spring Boot的基于Servlet的Web应用程序的默认上下文类。 |
| DEFAULT\_REACTIVE\_WEB\_CONTEXT\_CLASS | String | "org.springframework.boot.web.reactive.context.AnnotationConfigReactiveWebServerApplicationContext" | Spring Boot的基于响应式的Web应用程序的默认上下文类。 |
| BANNER\_LOCATION\_PROPERTY\_VALUE | String | "banner.txt" | 默认的banner文件名称。 |
| BANNER\_LOCATION\_PROPERTY | String | "spring.banner.location" | 用于配置banner文件位置的属性名。 |
| SYSTEM\_PROPERTY\_JAVA\_AWT\_HEADLESS | String | "java.awt.headless" | Java系统属性，用于配置是否在没有显示或图形用户界面的情况下运行。 |

## 5.2 变量

本系统中使用的部分关键常量说明如下表所示。

**表 2 变量列表**

|  |  |  |  |
| --- | --- | --- | --- |
| 变量名称 | 数据类型 | 所在目录 | 功能说明 |
| primarySources | Set<Class<?>> | E:\rencaizhaopinxitong\maven-3.3.9\repository\org\springframework\boot\spring-boot\2.2.2.RELEASE\spring-boot-2.2.2.RELEASE.jar!\org\springframework\boot\SpringApplication.class | 包含主应用源（通常是主配置类）的Set集合。 |
| sources | Set<String> | 包含应用源（通常是Spring的@Configuration类）的Set集合的字符串表示（类的全限定名）。 |
| mainApplicationClass | Class<?> | 主应用的Class对象。 |
| bannerMode | Banner.Mode | 用于设置banner的显示模式。 |
| logStartupInfo | boolean | 表示是否在启动时记录日志信息。 |
| addCommandLineProperties | boolean | 表示是否将命令行参数添加到Spring环境中。 |
| addConversionService | boolean | 表示是否添加一个类型转换服务到应用上下文中。 |
| allowBeanDefinitionOverriding | boolean | 表示是否允许重新定义已存在的Bean定义。 |
| isCustomEnvironment | boolean | 表示环境是否已被自定义。 |
| lazyInitialization | boolean | 表示是否启用Bean的延迟初始化。 |

## 5.3 数据结构

本系统所采用的部分关键数据结构说明如下表所示。

**表 3 报名招聘（baomingzhaopin）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| gongsi | varchar(200) | 公司 | 存储公司名称 | 默认值为NULL |
| fabushijian | datetime | 发布时间 | 存储发布时间 | 默认值为NULL |
| jiezhishijian | datetime | 截止时间 | 存储截止时间 | 默认值为NULL |
| zhaopian | varchar(200) | 照片 | 存储照片的  URL或路径 | 默认值为NULL |
| zhanghao | varchar(200) | 账号 | 存储用户或  公司的账号 | 默认值为NULL |
| xingming | varchar(200) | 姓名 | 存储姓名 | 默认值为NULL |
| jianli | varchar(200) | 简历 | 存储简历 | 默认值为NULL |
| sfsh | varchar(200) | 是否审核 | 存储是否审核信息 | 默认值为否 |
| shhf | longtext | 审核回复 | 存储审核后  的回复内容 | 无 |

**表 4 在线客服（chat）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| userid | bigint(20) | 用户id | 存储用户id | 非空 |
| adminid | bigint(20) | 管理员id | 存储管理员id | 默认值为NULL |
| ask | longtext | 提问 | 存储提问信息 | 无 |
| reply | longtext | 回复 | 存储回复信息 | 无 |
| isreply | int(11) | 是否回复 | 存储是否回复信息 | 默认值为NULL |

**表 5 求职论坛（forum）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| title | varchar(200) | 帖子标题 | 存储帖子标题 | 默认值为NULL |
| content | longtext | 帖子内容 | 存储帖子内容 | 非空 |
| parentid | bigint(20) | 父节点id | 存储父节点id | 默认值为NULL |
| userid | bigint(20) | 用户id | 存储用户id | 非空 |
| username | varchar(200) | 用户名 | 存储用户名 | 默认值为NULL |
| isdone | varchar(200) | 状态 | 存储状态  （如“开放”） | 默认值为NULL |

**表 6 简历模板（jianlimoban）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| mobanmingcheng | varchar(200) | 模板名称 | 存储模板名称 | 默认值为NULL |
| mobanleixing | varchar(200) | 模板类型 | 存储模板类型 | 默认值为NULL |
| jianjie | longtext | 简介 | 存储简介 | 无 |
| zhaopian | varchar(200) | 照片 | 存储照片 | 默认值为NULL |
| thumbsupnum | int(11) | 赞 | 存储点赞情况 | 默认值为“0” |
| crazilynum | int(11) | 踩 | 存储踩一下的情况 | 默认值为“0” |
| clicktime | datetime | 最近点击时间 | 存储最近点击时间 | 默认值为NULL |
| clicknum | int(11) | 点击次数 | 存储点击次数 | 默认值为“0” |

**表 7 简历上传（jianlishangchuan）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| zhanghao | varchar(200) | 账号 | 存储账号信息 | 默认值为NULL |
| xingming | varchar(200) | 姓名 | 存储姓名信息 | 默认值为NULL |
| jianli | varchar(200) | 简历 | 存储简历信息 | 默认值为NULL |
| fankui | varchar(200) | 反馈 | 存储反馈信息 | 默认值为NULL |
| shangchuanshijian | datetime | 上传时间 | 存储上传时间 | 默认值为NULL |
| sfsh | varchar(200) | 是否审核 | 存储是否审核信息 | 默认值为“否” |
| shhf | longtext | 审核回复 | 存储审核回复内容 | 无 |

**表 8 模板类型（mobanleixing）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| leixing | varchar(200) | 类型 | 存储模板类型信息 | 默认值为NULL |

**表 9 系统公告（news）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| title | varchar(200) | 标题 | 存储系统公告标题 | 非空 |
| introduction | longtext | 简介 | 存储系统公告简介 | 无 |
| picture | varchar(200) | 图片 | 存储系统公告图片 | 非空 |
| content | longtext | 内容 | 存储系统公告内容 | 非空 |

**表 10 收藏表（storeup）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| userid | bigint(20) | 用户id | 存储用户id | 非空 |
| refid | bigint(20) | 收藏id | 存储收藏id | 默认值为NULL |
| tablename | varchar(200) | 表名 | 存储表名  （如“jianlimoban”） | 默认值为NULL |
| name | varchar(200) | 收藏名称 | 存储收藏名称  （如“商务”） | 非空 |
| picture | varchar(200) | 收藏图片 | 存储收藏图片 | 非空 |

**表 11 token表（token）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| userid | bigint(20) | 用户id | 存储用户id | 非空 |
| username | varchar(100) | 用户名 | 存储用户名 | 非空 |
| tablename | varchar(100) | 表名 | 存储表名 | 默认值为NULL |
| role | varchar(100) | 角色 | 存储角色信息  （如“管理员”  “用户”） | 默认值为NULL |
| token | varchar(200) | 密码 | 存储密码 | 非空 |
| addtime | timestamp | 新增时间 | 存储新增时间 | 非空；  默认值为当前时间戳 |
| expiratedtime | timestamp | 过期时间 | 存储过期时间 | 非空；  默认值为当前时间戳 |

**表 12 用户表（users）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| username | varchar(100) | 用户名 | 存储用户名 | 非空 |
| password | varchar(100) | 密码 | 存储密码 | 非空 |
| role | varchar(100) | 角色 | 存储角色信息  （如“管理员”  “用户”） | 默认值为“管理员” |
| addtime | timestamp | 新增时间 | 存储新增时间 | 非空；  默认值为当前时间戳 |

**表 13 用户（yonghu）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| zhanghao | varchar(200) | 账号 | 存储账号信息 | 非空；唯一键 |
| mima | varchar(200) | 密码 | 存储用户密码 | 非空 |
| xingming | varchar(200) | 姓名 | 存储用户姓名 | 非空 |
| xingbie | varchar(200) | 性别 | 存储用户性别 | 默认值为NULL |
| nianling | int(11) | 年龄 | 存储用户年龄 | 默认值为NULL |
| shouji | varchar(200) | 手机 | 存储用户手机号 | 默认值为NULL |
| youxiang | varchar(200) | 邮箱 | 存储用户邮箱 | 默认值为NULL |
| shenfenzheng | varchar(200) | 身份证 | 存储用户身份证 | 默认值为NULL |
| touxiang | varchar(200) | 头像 | 存储用户头像 | 默认值为NULL |

**表 14 招聘会（zhaopinhui）数据结构表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **数据类型** | **注释** | **功能说明** | **约束** |
| id | bigint(20) | 主键 | 唯一标识表中的  每一行记录 | 非空；自增 |
| addtime | timestamp | 创建时间 | 存储创建时间 | 非空；  默认值为当前时间戳 |
| gongsi | varchar(200) | 公司 | 存储公司信息 | 默认值为NULL |
| fabushijian | datetime | 发布时间 | 存储发布时间 | 默认值为NULL |
| jiezhishijian | datetime | 截止时间 | 存储截止时间 | 默认值为NULL |
| didian | varchar(200) | 地点 | 存储地点 | 默认值为NULL |
| neirong | longtext | 内容 | 存储内容 | 无 |
| zhaopian | varchar(200) | 照片 | 存储照片 | 默认值为NULL |
| clicktime | datetime | 最近点击时间 | 存储最近点击时间 | 默认值为NULL |
| clicknum | int(11) | 点击次数 | 存储点击次数 | 默认值为“0” |

**6 逻辑架构视图**

## 6.1顶层逻辑架构视图

**6.1.1 逻辑视图**

**（1）注册中心**

组件：使用Eureka或Consul作为服务注册与发现中心，确保微服务实例能够动态注册与注销，实现服务的自动发现与负载均衡。逻辑视图中，注册中心作为一个核心组件，负责维护所有微服务实例的地址信息，为服务间通信提供基础。

逻辑连接：在逻辑视图中，所有微服务启动时会向注册中心注册自身，并定期发送心跳保持活跃状态。客户端或服务间调用时，首先查询注册中心获取服务实例列表，之后通过负载均衡策略选择实例进行通信。

**（2）服务中心**

组件：服务中心包括API Gateway和微服务治理组件（如Spring Cloud Gateway、Zuul）。API Gateway作为系统入口，负责路由、认证、限流等，而微服务治理组件则提供服务熔断、服务降级、服务跟踪等高级功能。

逻辑布局：在逻辑视图中，API Gateway位于前端应用与后端微服务之间，实现请求的初步处理与路由。微服务治理组件与微服务紧密集成，监控并管理服务间的交互，确保高可用性与高性能。

**（3）中间件**

技术栈：采用消息队列（如RabbitMQ、Kafka）处理异步通信与解耦，Redis作为缓存服务提高数据读取速度，Spring Cloud Config作为配置中心管理微服务的配置信息。

逻辑架构：在逻辑视图里，消息队列作为消息传递的基础设施，连接不同的微服务，实现异步处理与事件驱动。Redis作为高性能缓存，缓存频繁访问的数据，减轻数据库压力。Spring Cloud Config确保配置的一致性和可管理性。

**（4）存储**

设计：MySQL 5.7作为主数据库存储核心业务数据，使用MyBatis或JPA进行ORM映射。考虑使用NoSQL数据库（如MongoDB）处理非结构化数据，如简历附件等。数据备份与恢复策略确保数据安全。

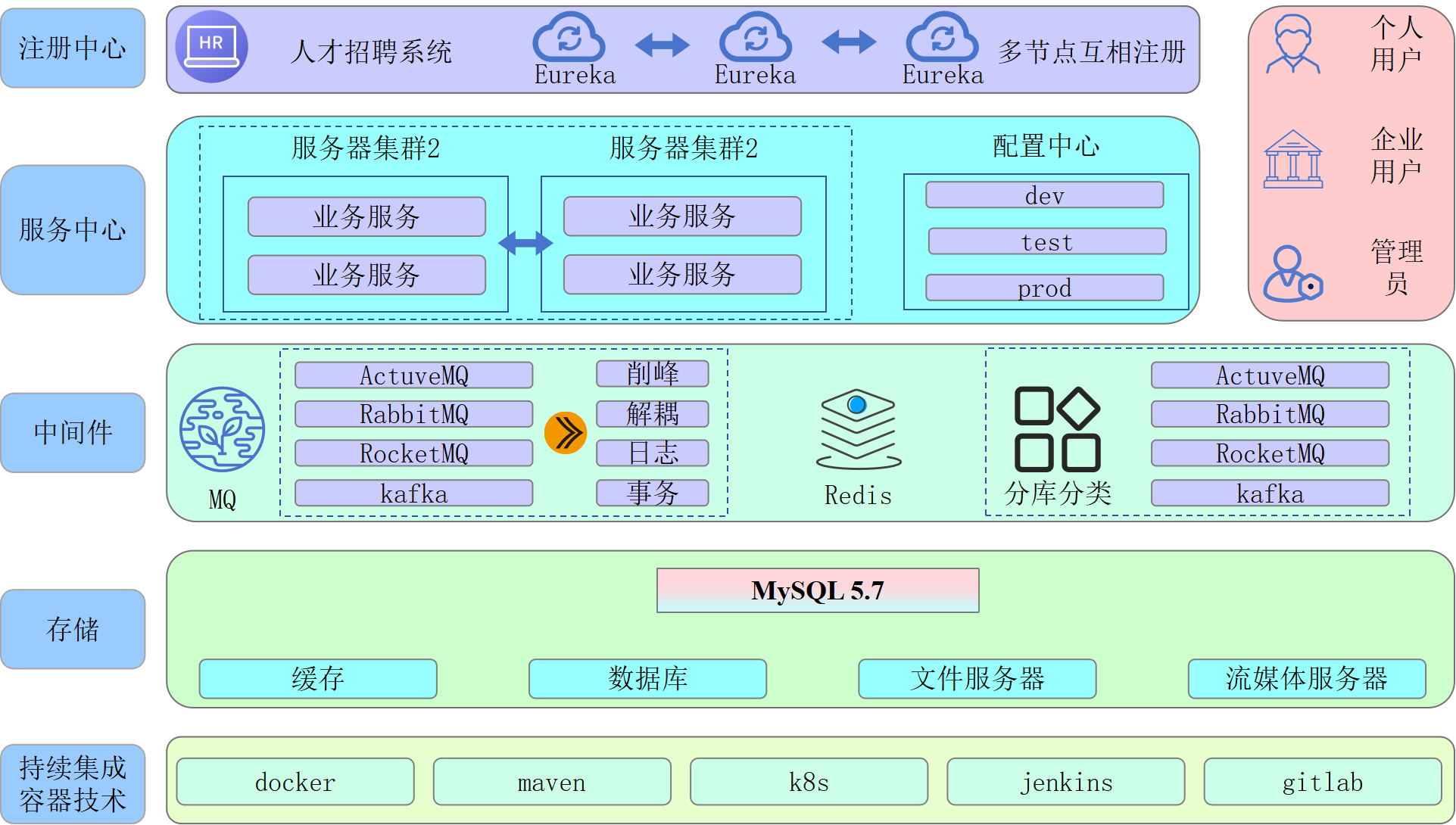
逻辑视图：数据库服务与微服务解耦，通过数据访问层与业务逻辑层交互。数据流图显示数据从录入、处理到存储的全过程，包括数据脱敏处理与加密存储策略。

**（5）持续集成和容器集成技术**

CI/CD：采用Jenkins实现代码的自动化构建、测试与部署，确保快速迭代与高质量交付。

容器化与编排：Docker用于应用的容器化包装，Kubernetes（K8s）作为容器编排平台，实现服务的自动部署、扩展与管理。

逻辑架构融入：在逻辑视图中，CI/CD流程作为开发视图的一部分，展示代码从提交到生产环境的自动化流程。容器服务与微服务架构紧密结合，利用K8s实现服务的弹性伸缩和高可用部署。



**图 1 逻辑视图**

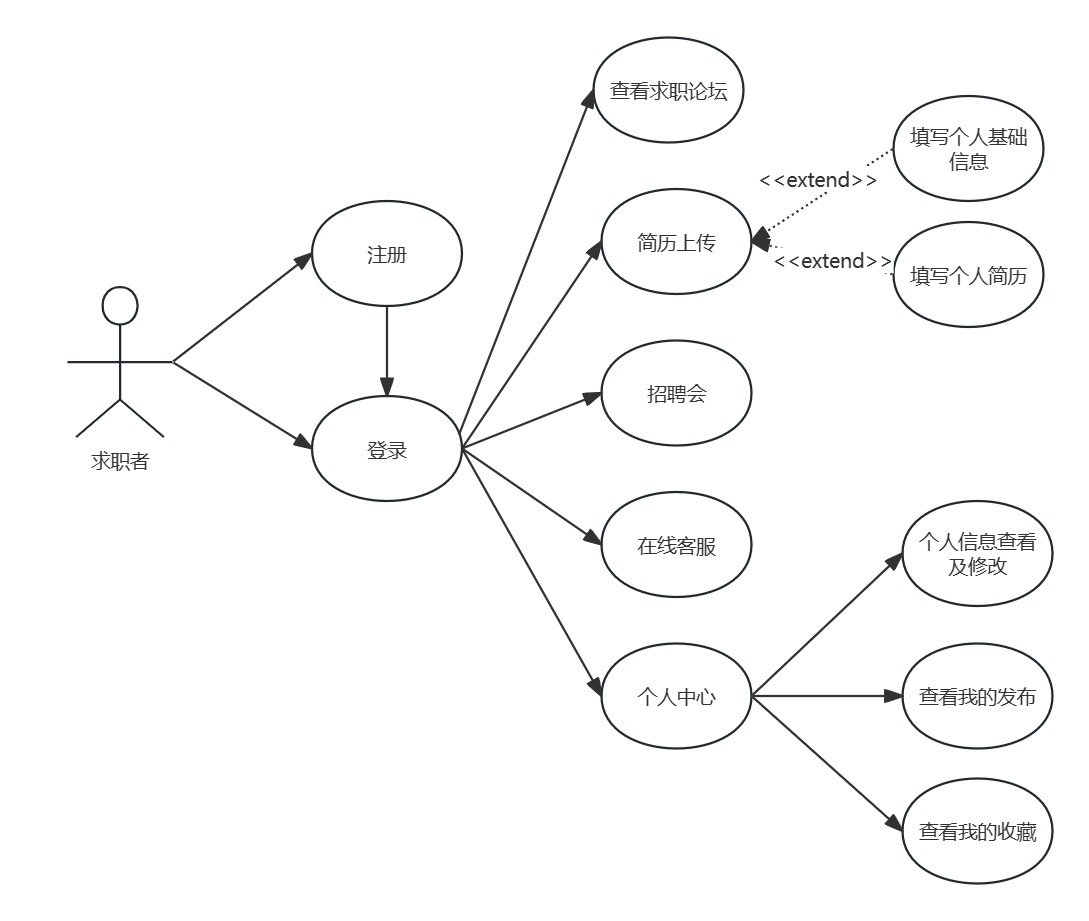
**6.1.2 数据流程视图**



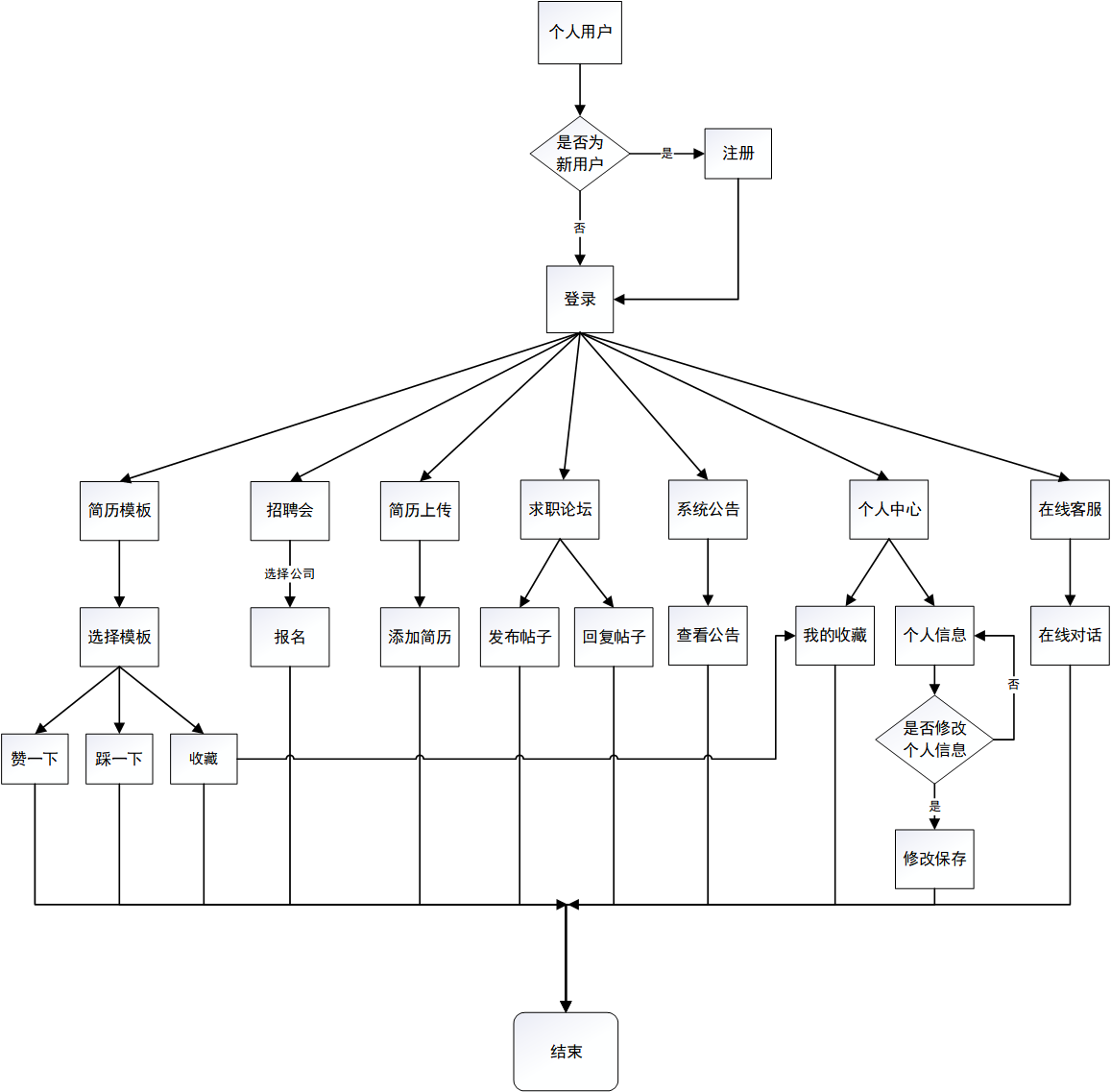
**图 2 数据流程图**

## 6.2 功能逻辑架构（分解）视图

**6.2.1个人用户端功能模块**

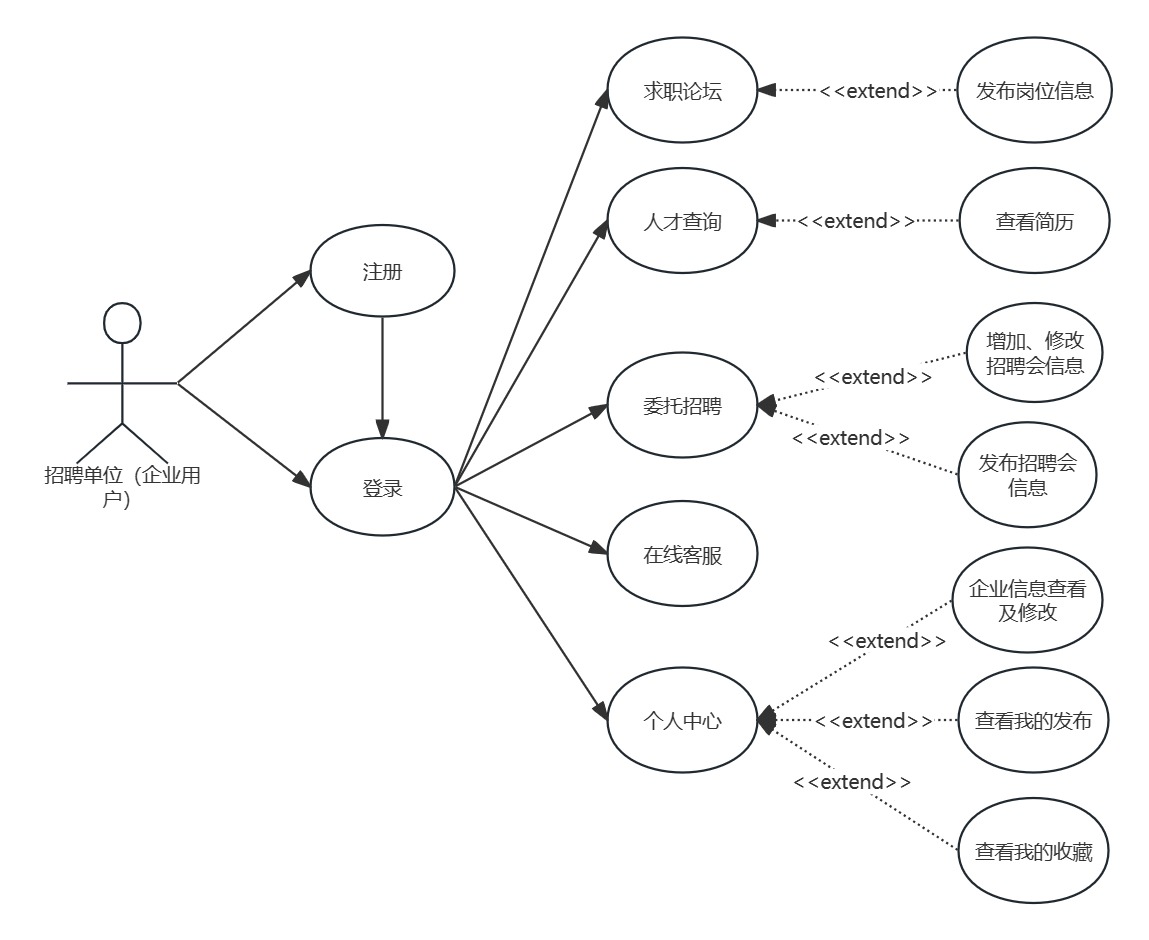


**图 3 个人用户用例图**

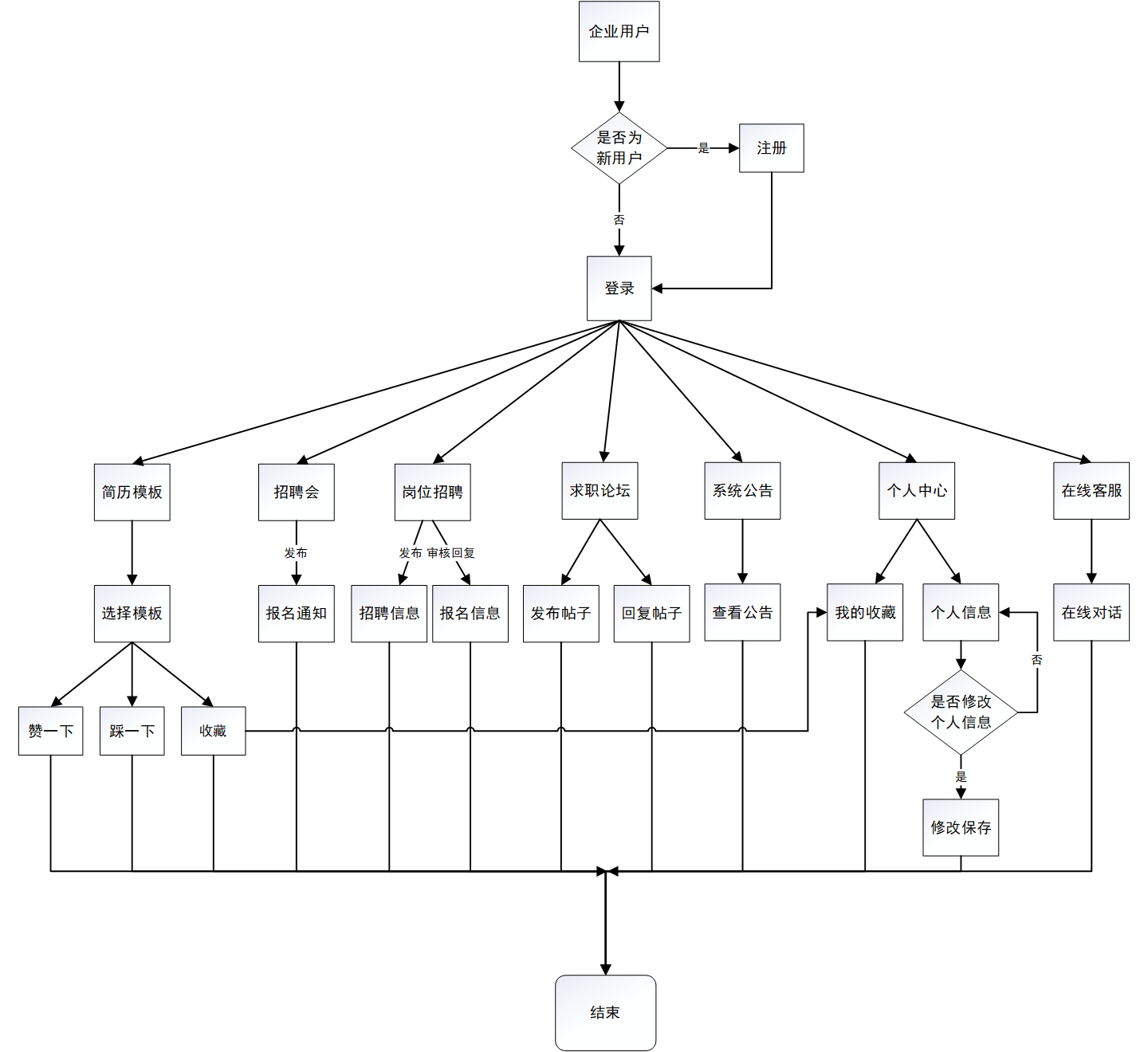


**图 4 个人用户功能逻辑视图**

**6.2.2企业用户端功能模块**

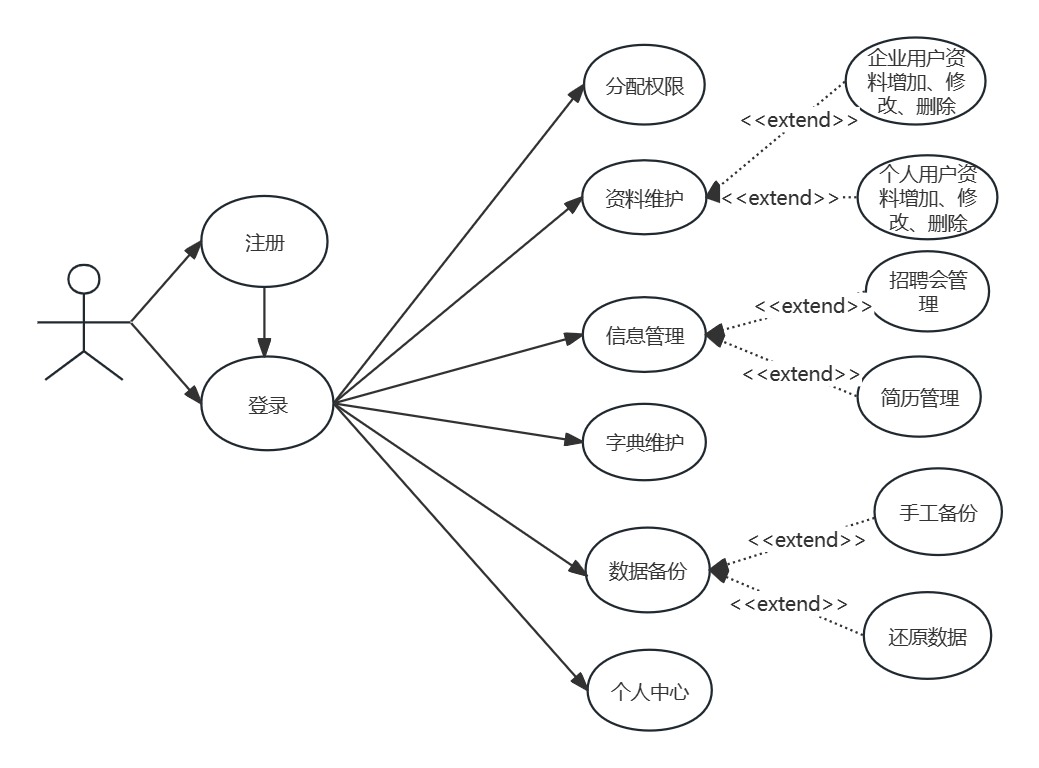


**图 5 企业用户用例图**

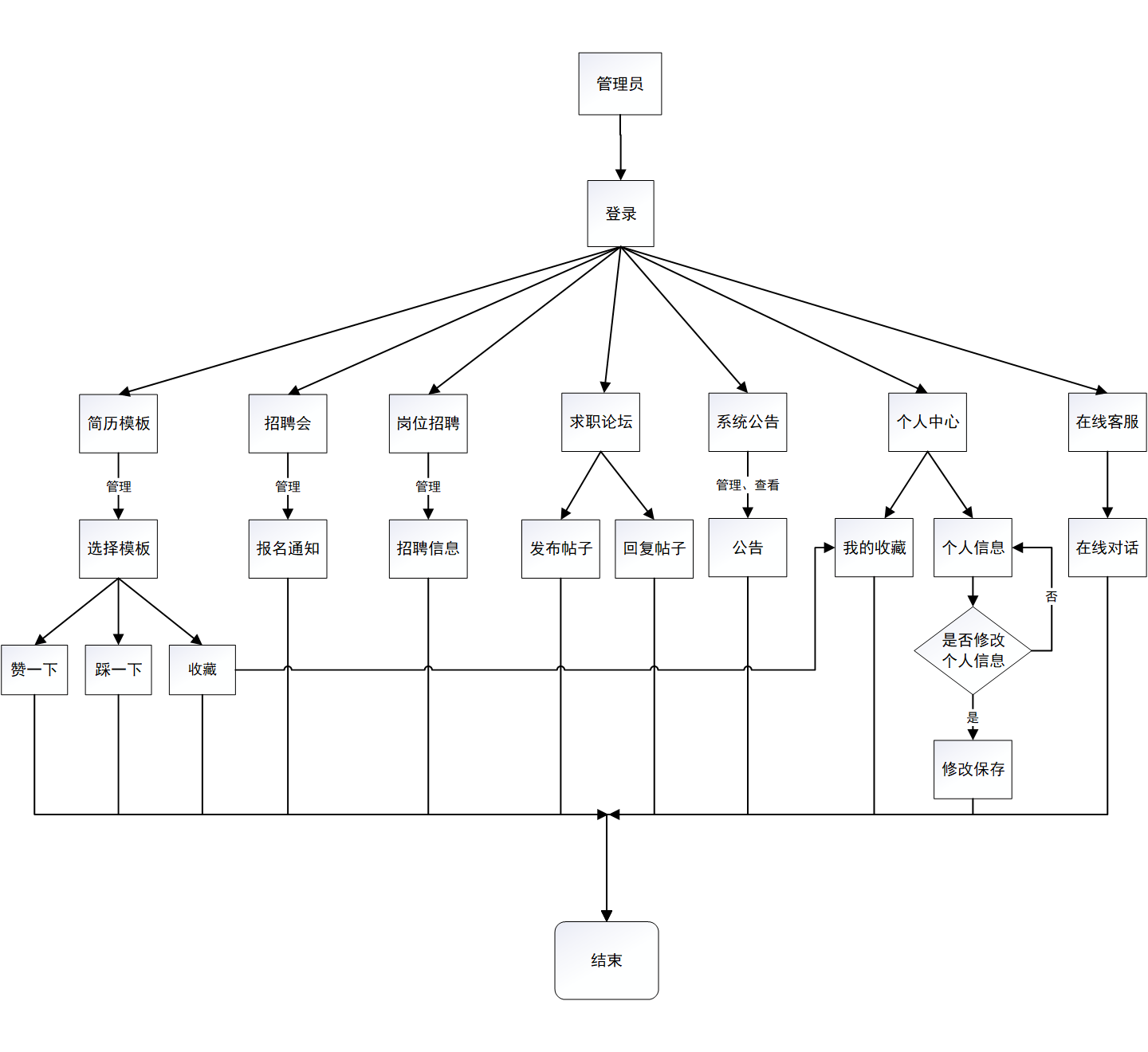


**图 6 企业用户功能逻辑视图**

**6.2.3管理员端功能模块**



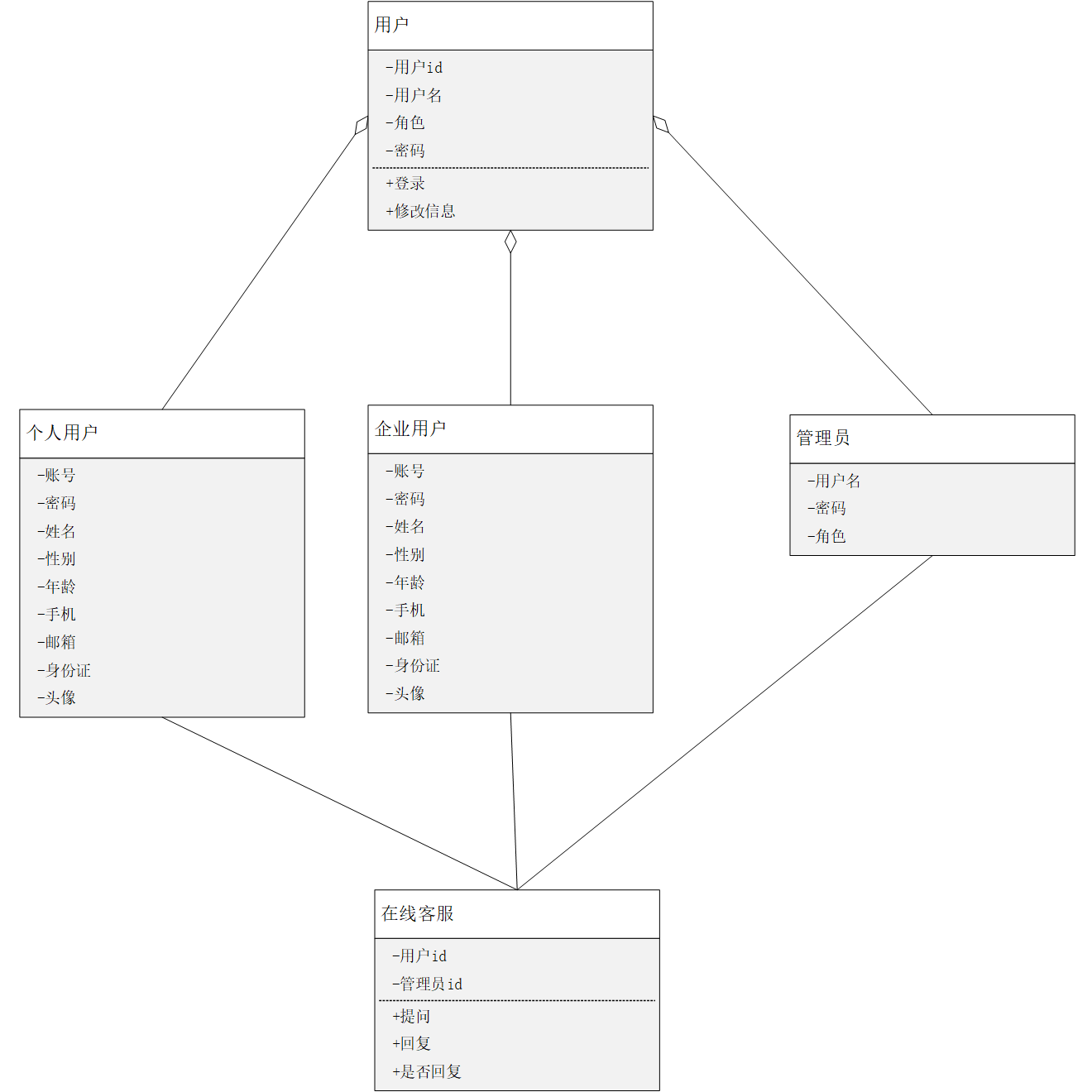
**图 7 管理员用例图**



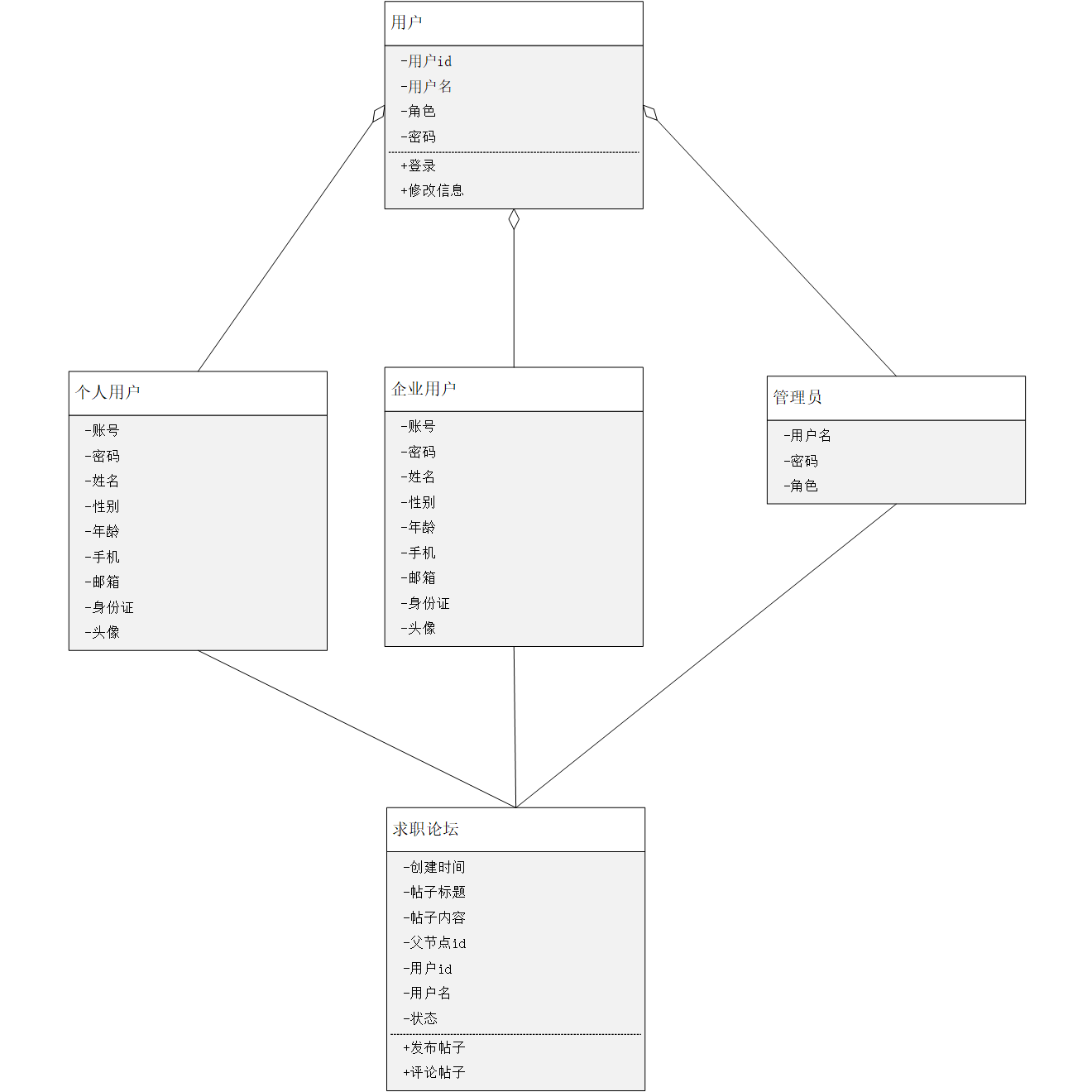
**图 8 管理员功能逻辑视图**

**6.2.4 功能逻辑视图**

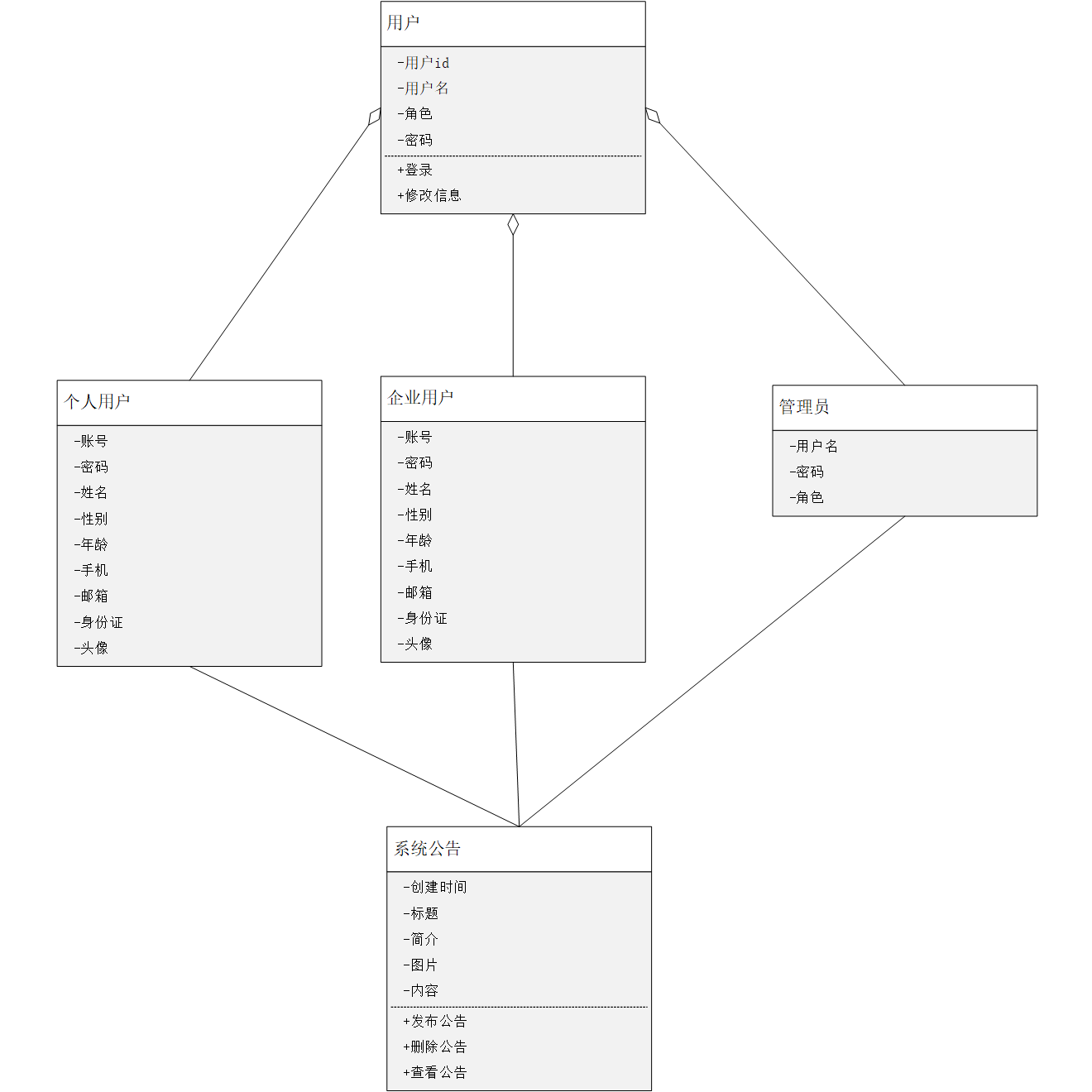
本部分以类图形式展现人才招聘系统各功能逻辑视图。



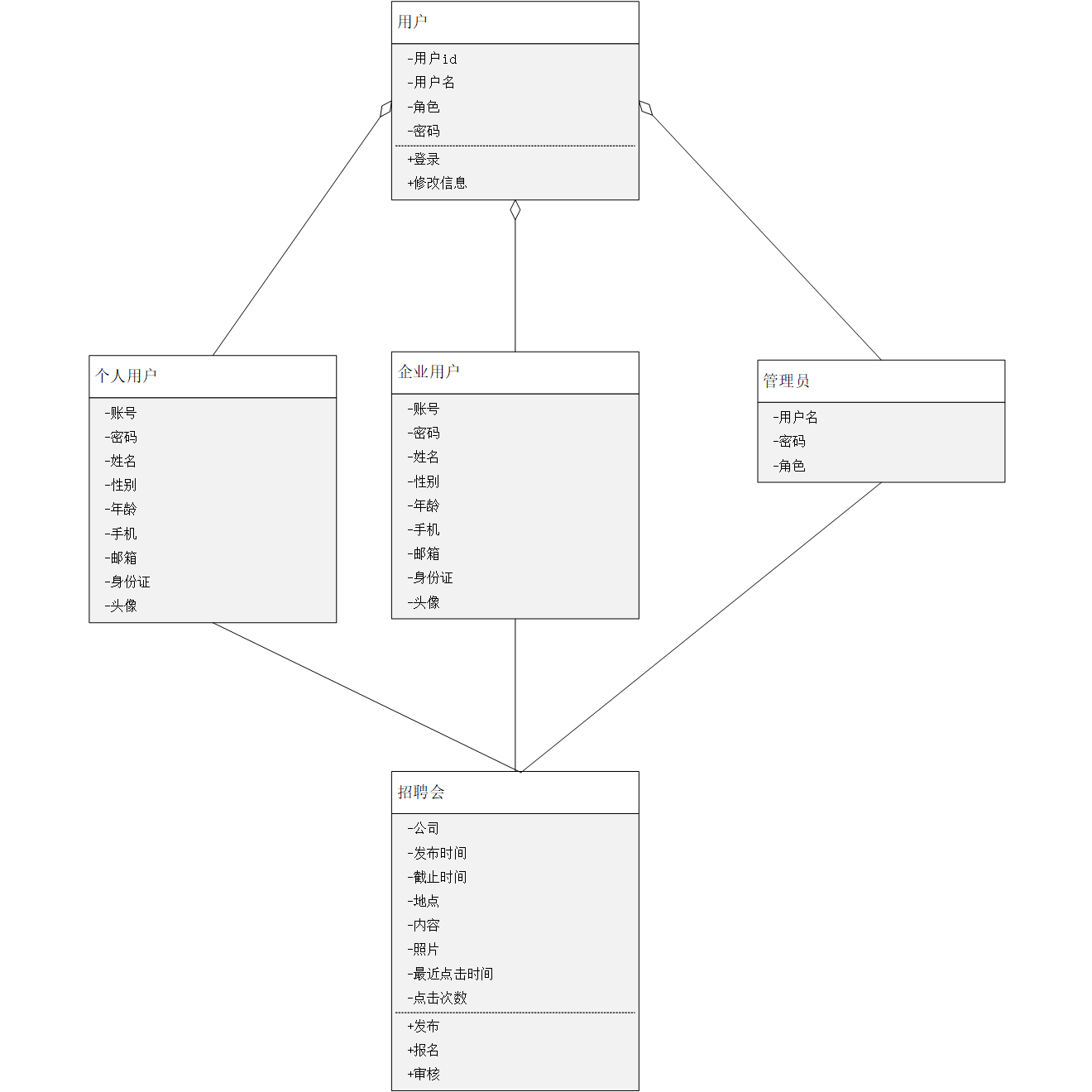
**图 9 在线客服功能类图**



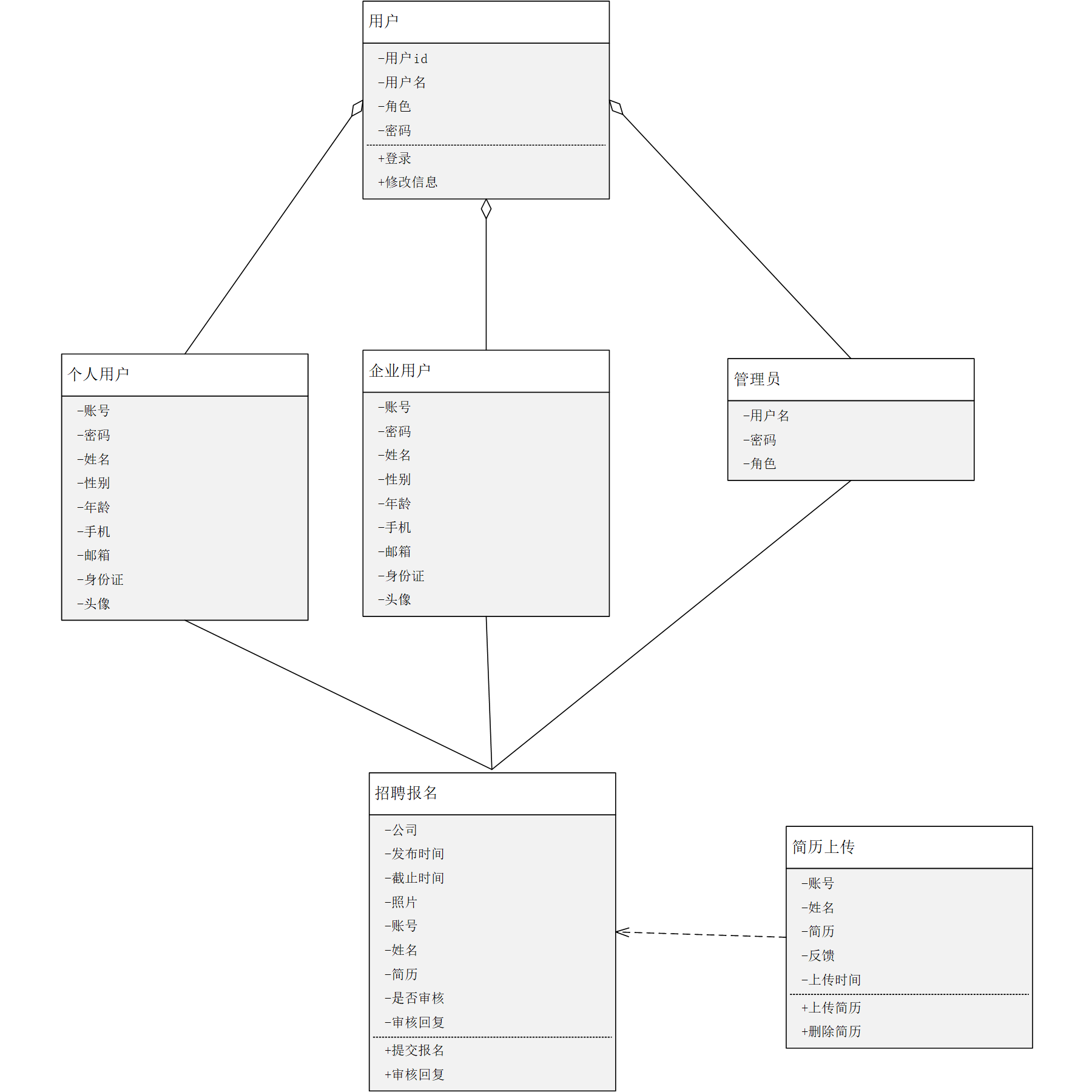
**图 10 求职论坛功能类图**



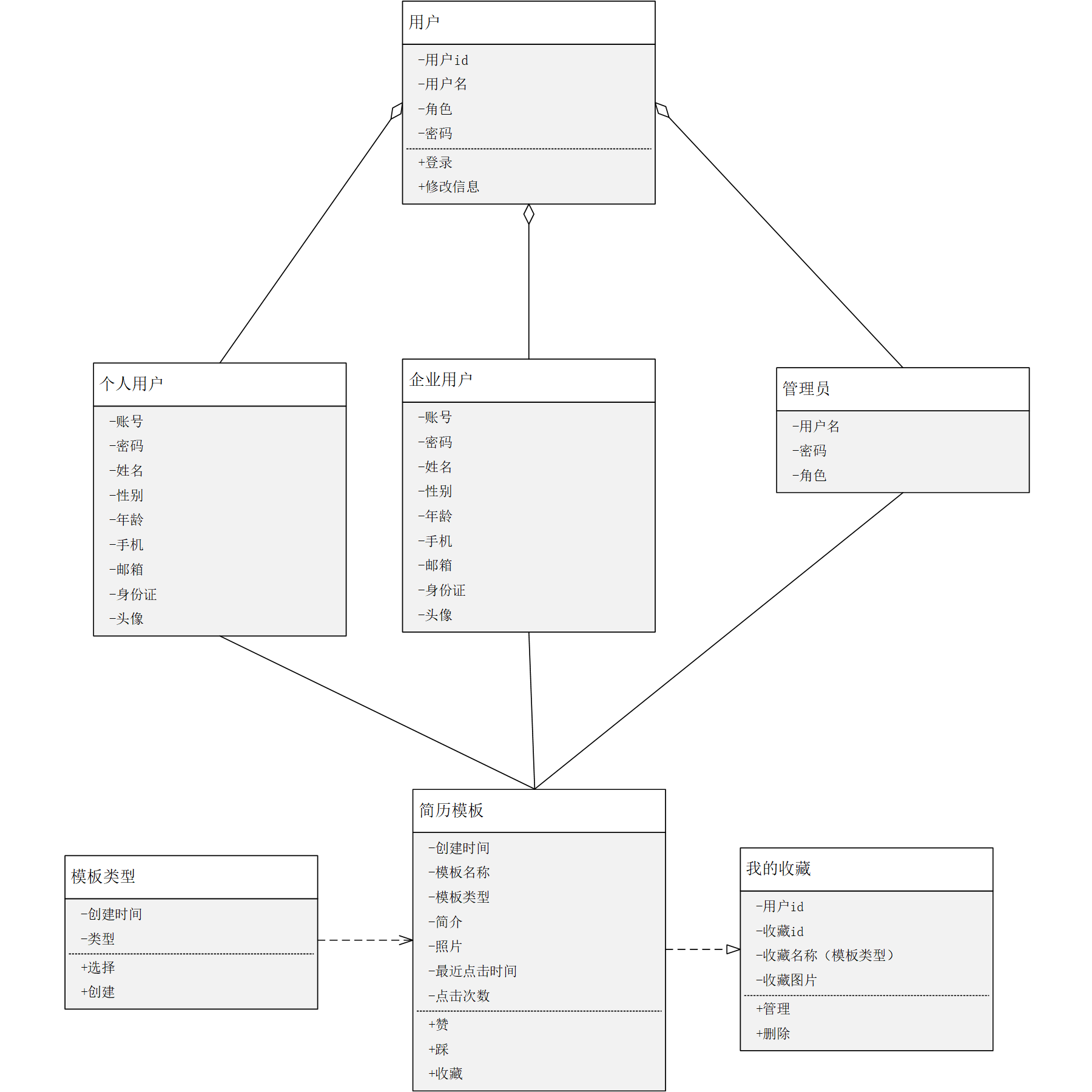
**图 11 系统公告功能类图**

****

**图 12 招聘会功能类图**

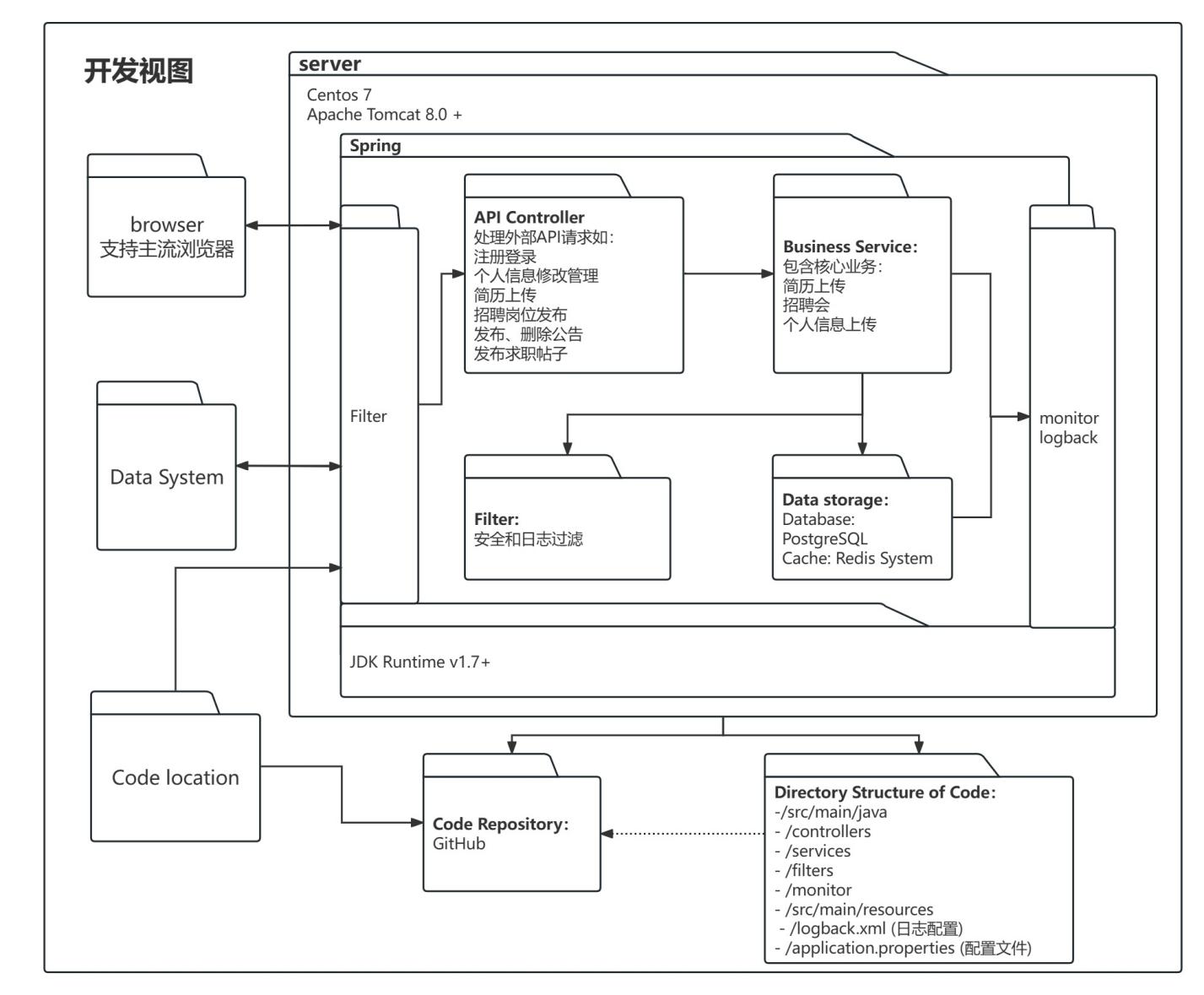


**图 13 招聘报名+简历上传功能类图**

****

**图 14 简历模板+收藏功能类图**

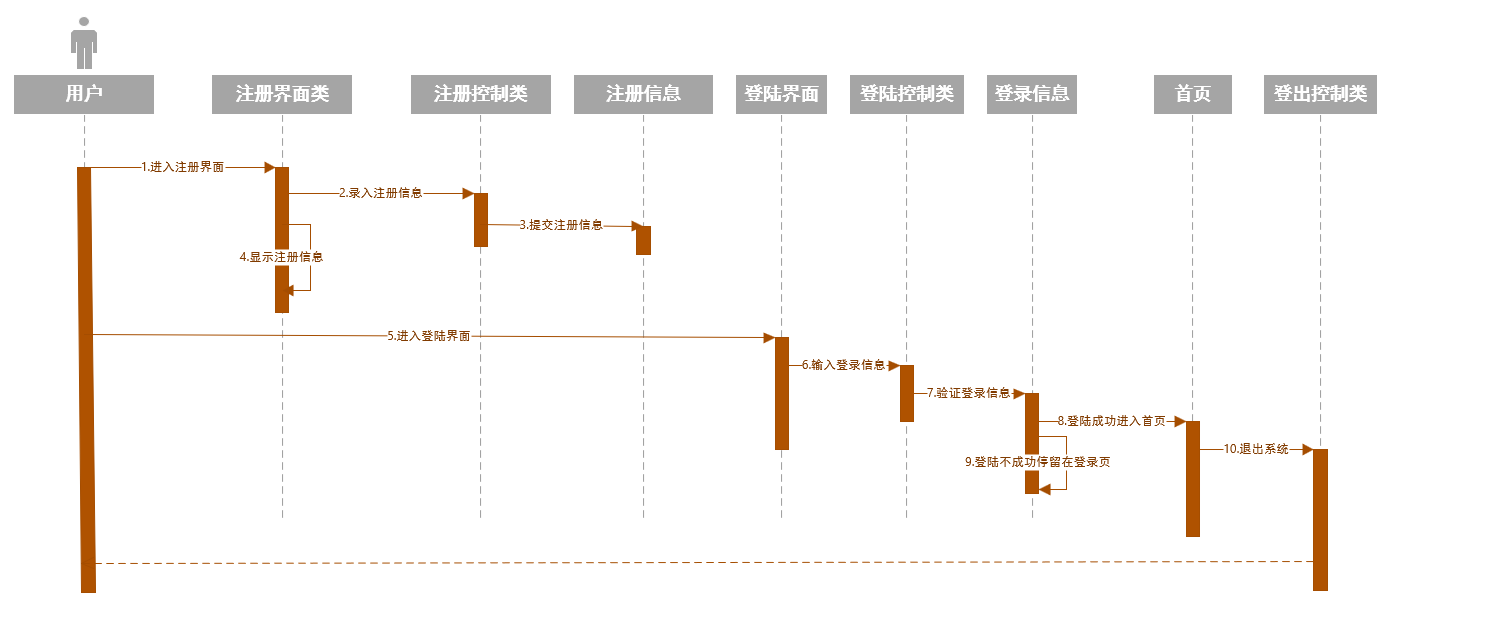
## 6.3 开发视图



**图 15 开发视图**

## 6.4 运行架构视图

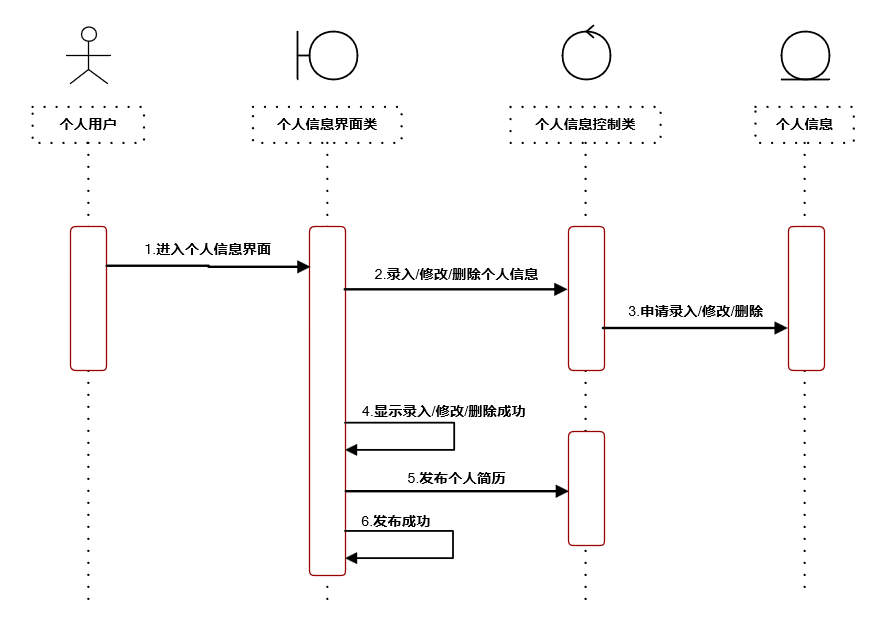
**6.4.1登陆运行视图**



**图 16 登录运行视图**

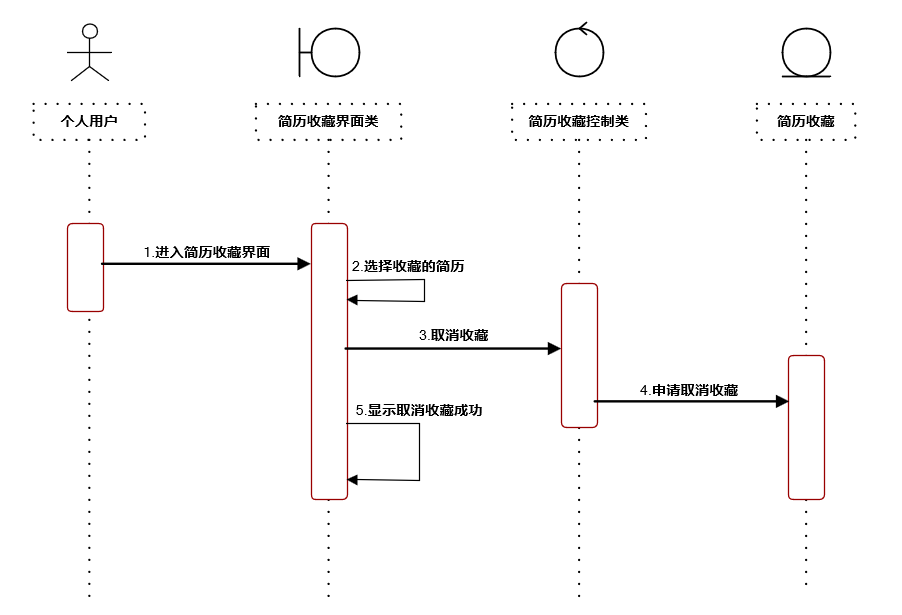
**6.4.2个人信息运行视图**

**（1）个人信息修改运行视图**



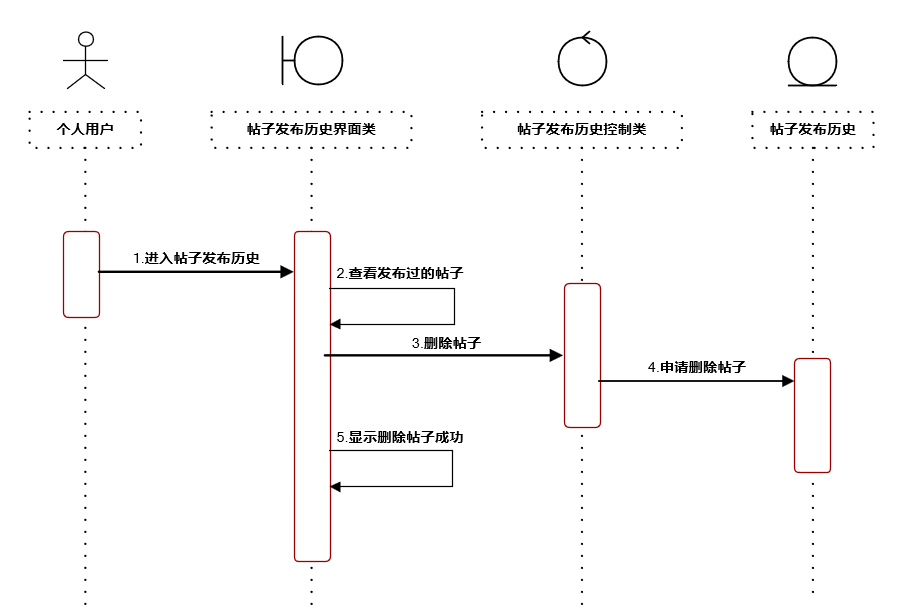
**图 17 个人信息修改运行视图**

**（2）管理建立收藏历史运行视图**



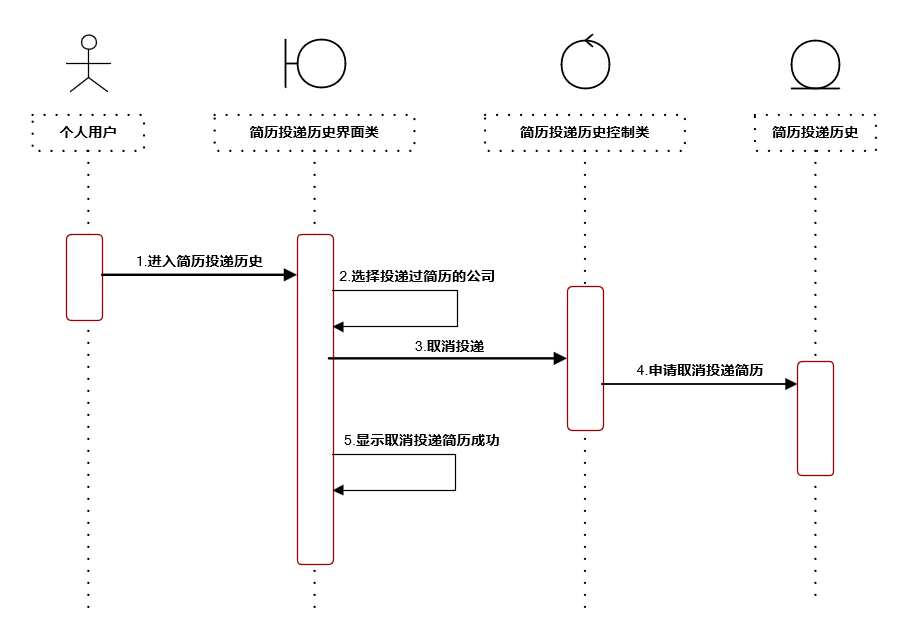
**图 18 管理建立收藏历史运行视图**

**（3）管理帖子发布历史运行视图**



**图 19 管理帖子发布历史运行视图**

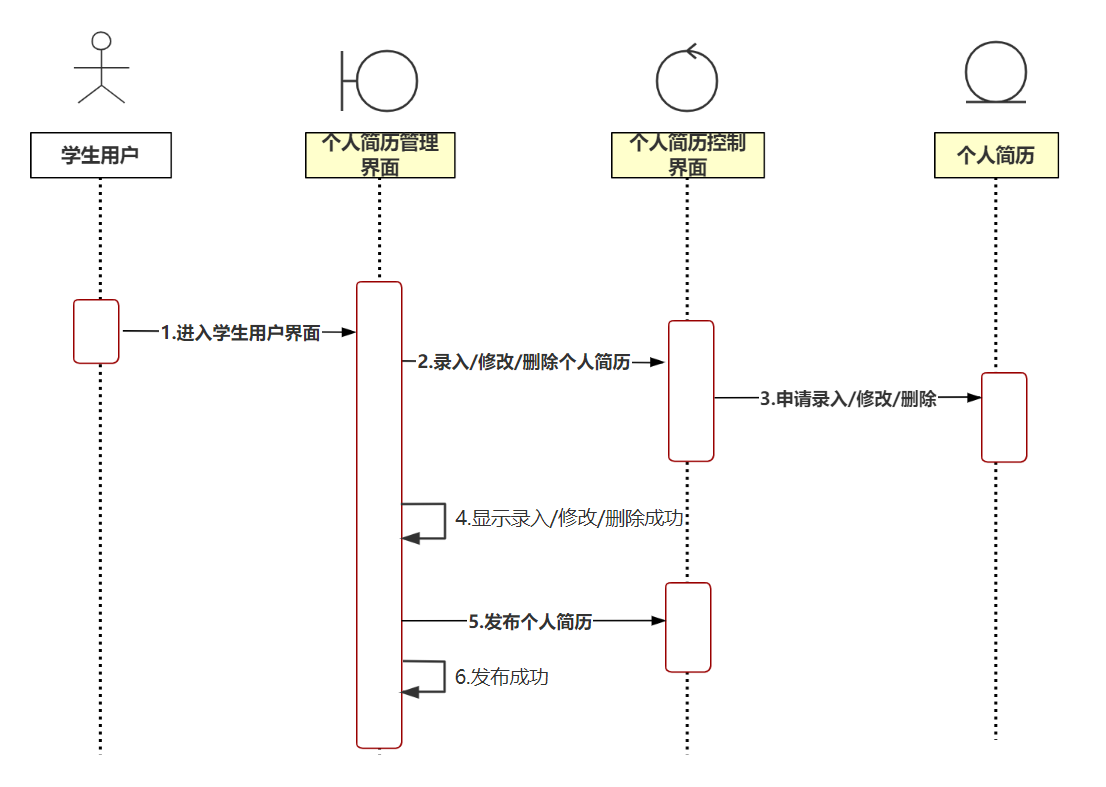
**（4）我的投递运行视图**



**图 20 我的投递运行视图**

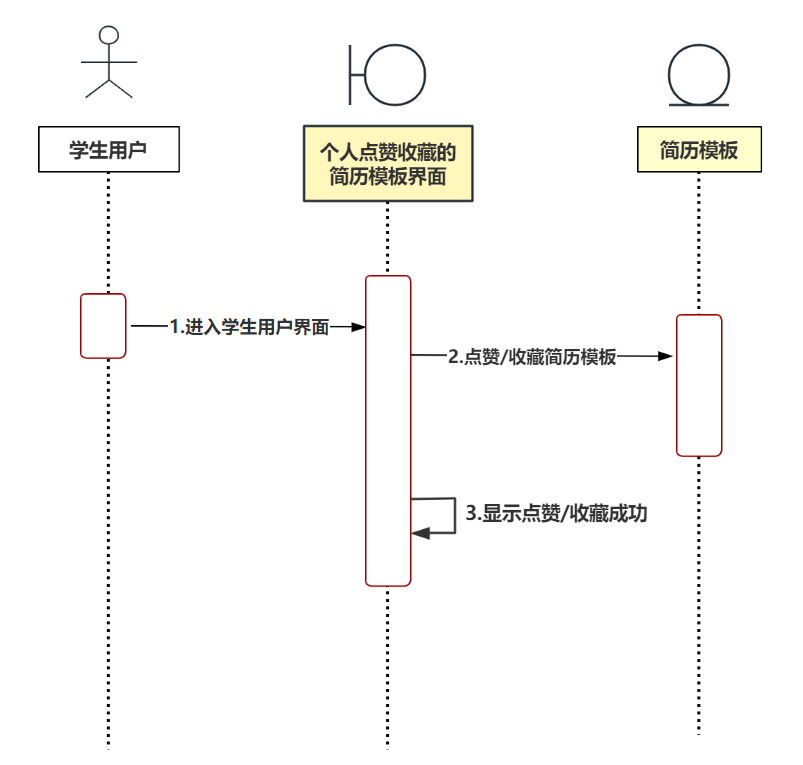
**6.4.3 简历模板运行视图**

**（1）简历模板应用运行视图**



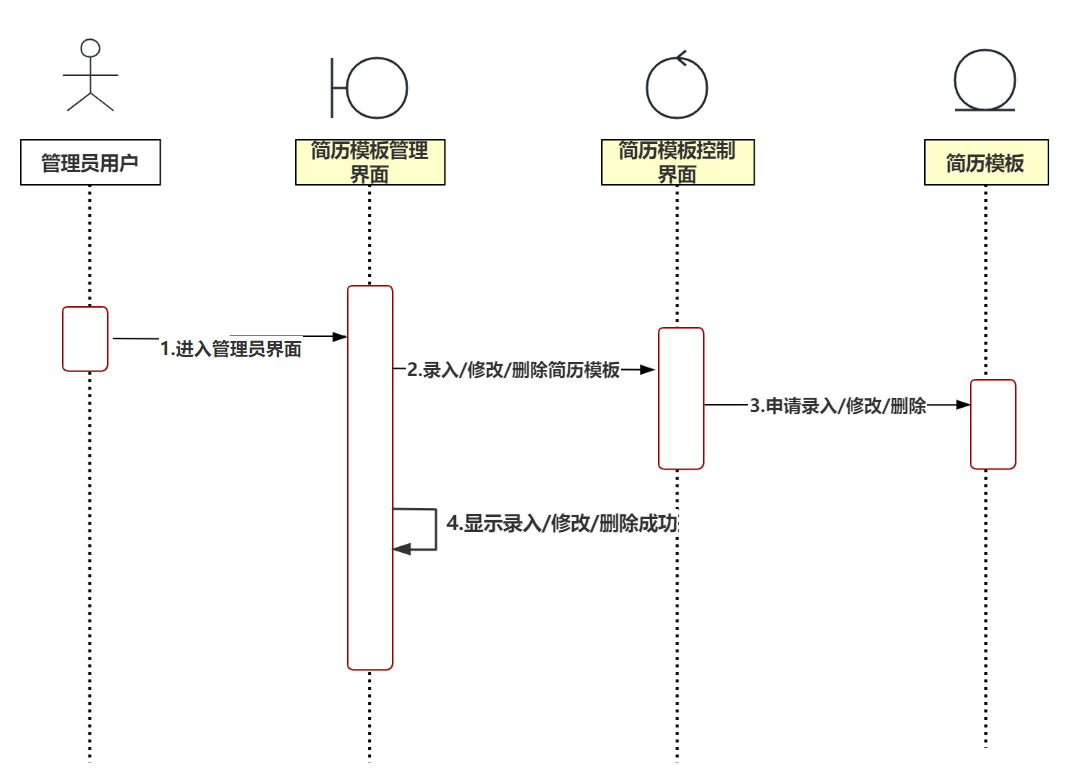
**图 21 简历模版应用运行视图**

**（2）简历模板点赞、收藏运行视图**



**图 22 简历模板点赞、收藏运行视图**

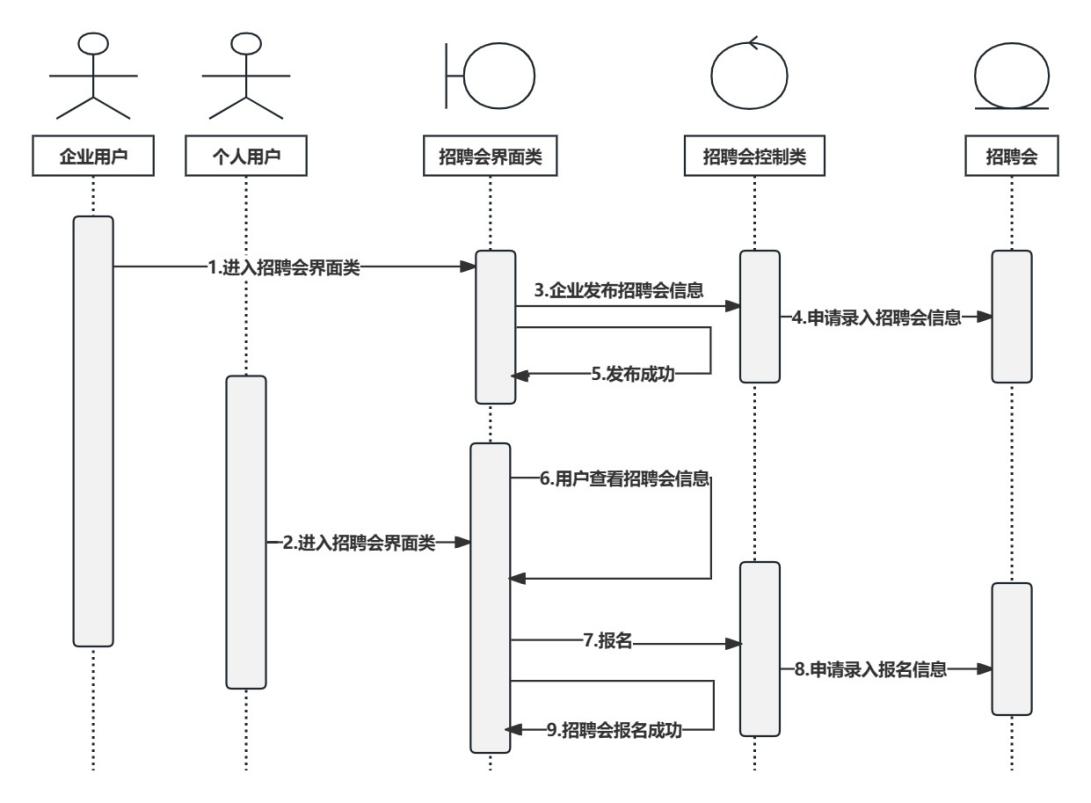
**（3）简历模板上传运行视图**



**图 23 简历模板上传运行视图**

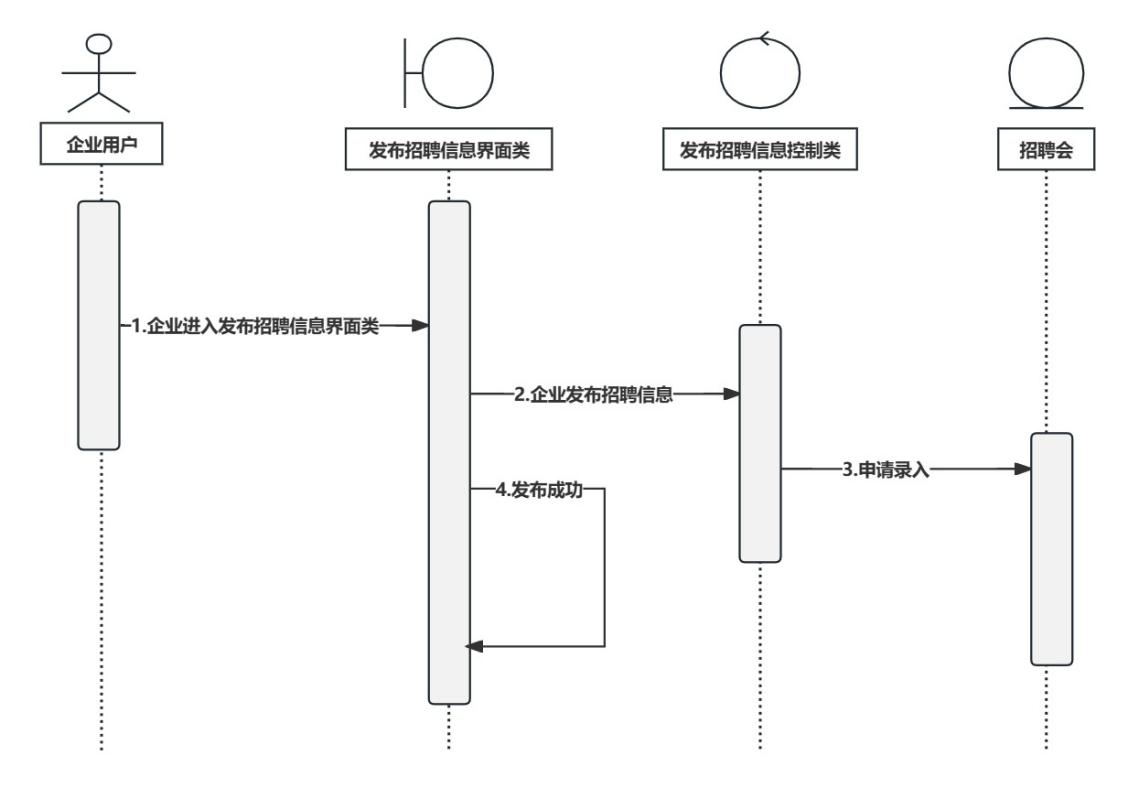
**6.4.4招聘运行视图**

**（1）招聘会运行视图**



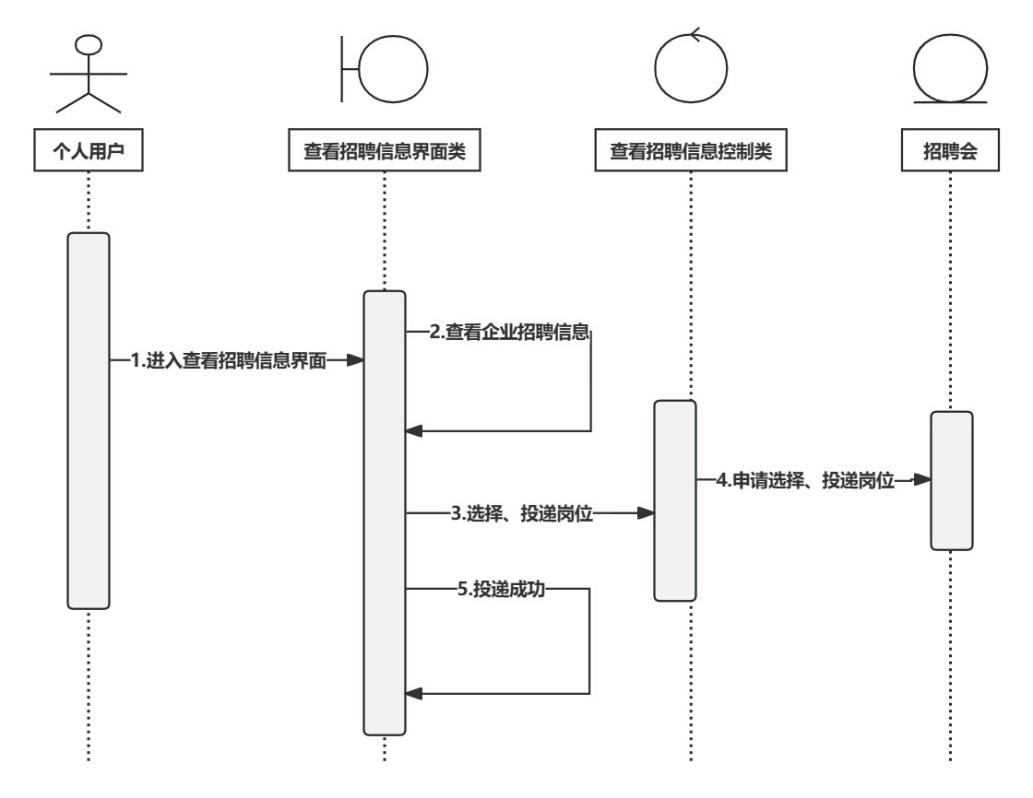
**图 24 招聘会运行视图**

**（2）企业发布招聘信息运行视图**



**图 25 企业发布招聘信息运行视图**

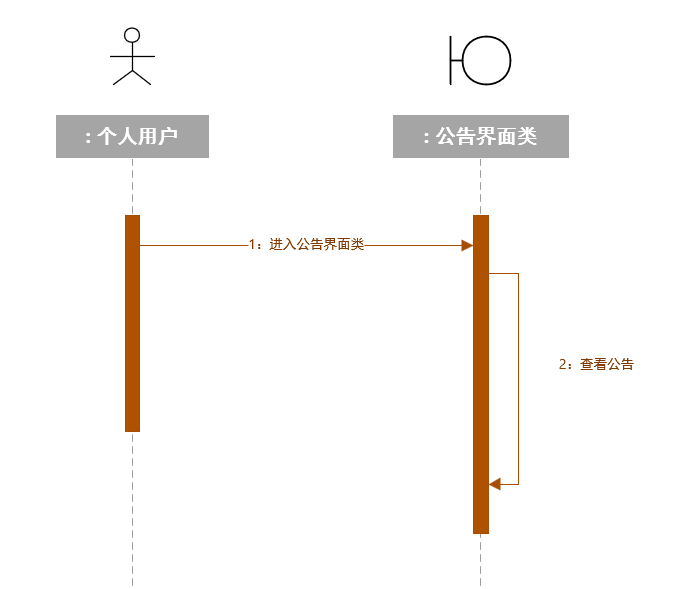
**（3）求职者查看招聘信息运行视图**



**图 26 求职者查看招聘信息运行视图**

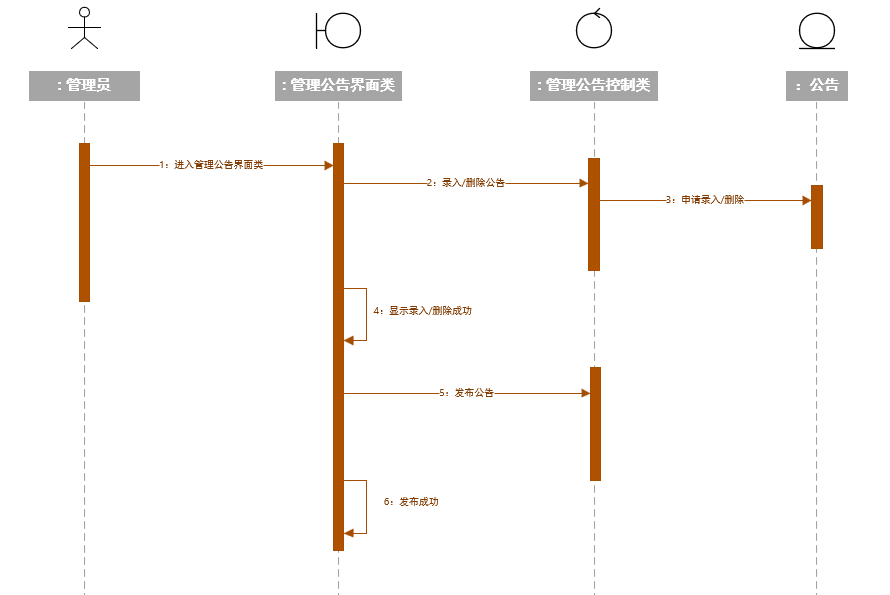
**6.4.5系统公告运行视图**

**（1）个人用户查看系统公告运行视图**



**图 27 个人用户查看系统公告运行视图**

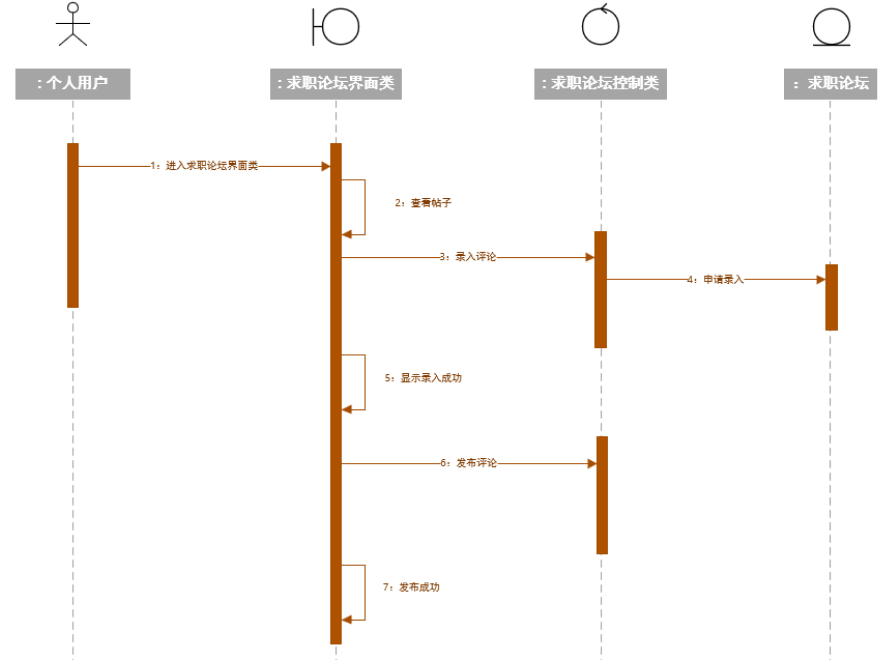
**（2）管理员发布、删除系统公告运行视图**



**图 28 管理员发布、删除系统公告运行视图**

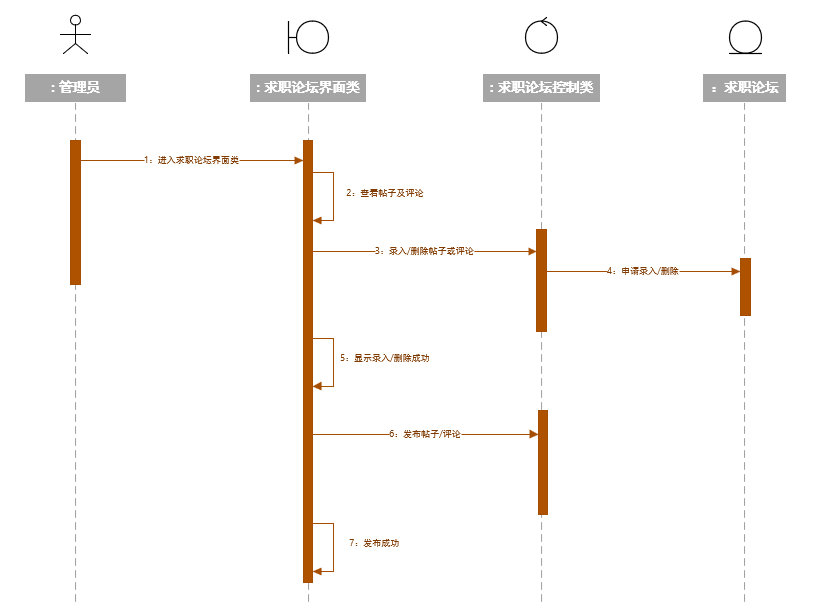
**6.4.6求职论坛运行视图**

**（1）评论帖子运行视图**



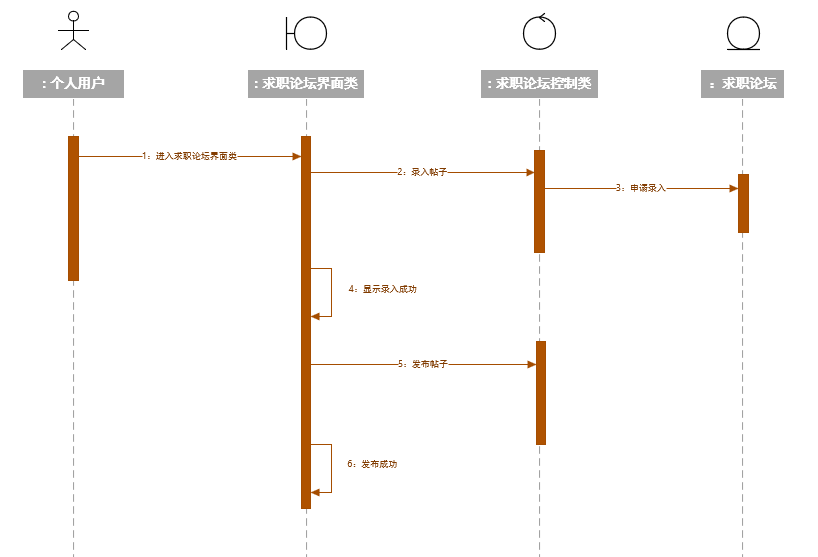
**图 29 评论帖子运行视图**

**（2）管理帖子运行视图**



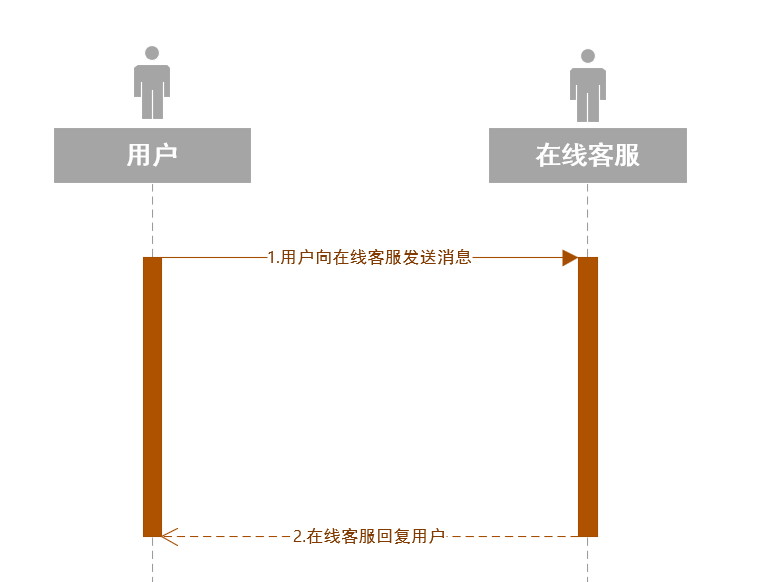
**图 30 管理帖子运行视图**

（3）发布帖子运行视图



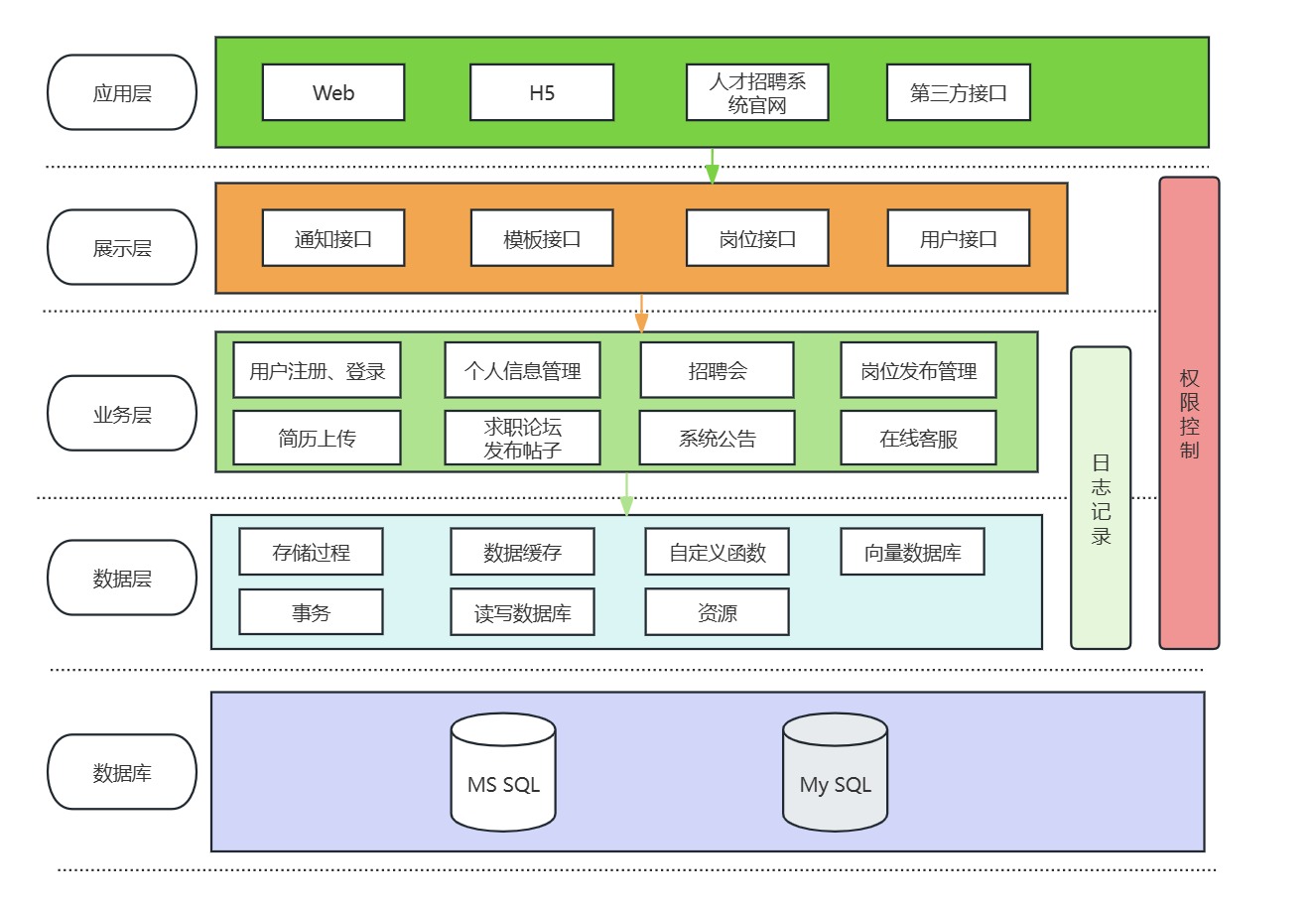
**图 31 发布帖子运行视图**

**6.4.7在线客服运行视图**



**图 32 在线客服运行视图**

## 6.5 实现视图



**图 33 实现视图**

## 6.6 物理架构视图

**6.6.1物理拓扑**

为了支持人才招聘系统的高效运行，我们进行了以下硬件选型。

（1）应用服务器：

型号：tomcat7

CPU：可以选择搭载IntelCorei513500H的LenovoThinkBook14，基准频率为2.5GHz。

内存：16GBRAM

存储：512GBSSD

（2）数据库服务器：

型号：DellPowerEdgeT130

CPU：至少具有4个核心的处理器，如IntelCorei5或AMDRyzen5。

内存：至少16GBDDR4RAM，数据库操作对内存需求较高。

存储：至少1TB的SATAHDD或更高容量的SSD，根据数据增长预期进行选择。

（3）负载均衡器

型号：F5BIGIP2200

性能：支持高达10Gbps的数据吞吐量。

功能：提供SSL卸载、连接复用和智能流量管理。

（4）安全设备

防火墙：CiscoASA5585，提供下一代防火墙功能，包括入侵防御和VPN。

入侵检测系统：Snort，部署在网络边缘，监控可疑活动并提供实时警报。

（5）备份设备

型号：QuantumDXi4700，提供数据备份和恢复解决方案。

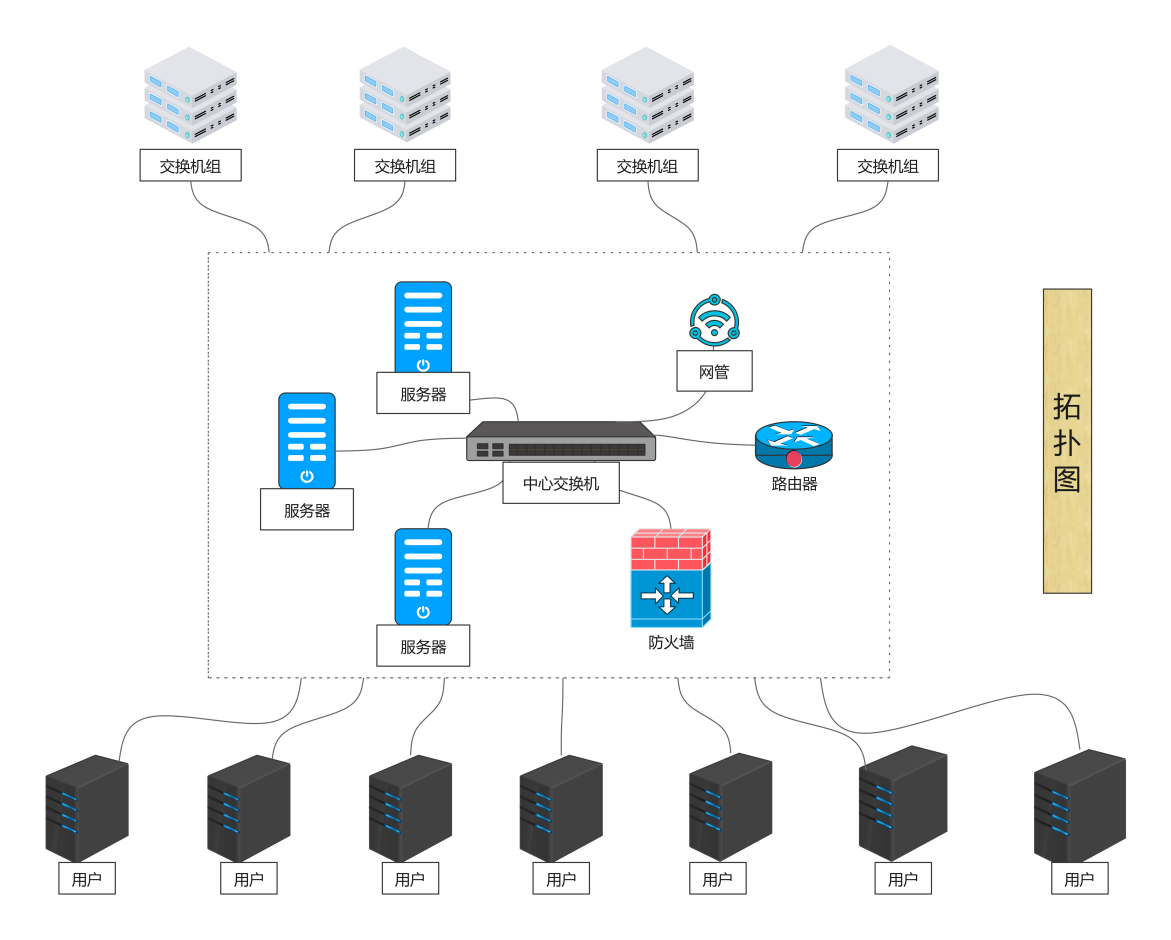
容量：最大支持8PB裸容量，支持虚拟带库功能。

（6）监控系统

硬件监控：Nagios，用于监控服务器和网络设备的健康状况。

性能监控：Zabbix，用于监控系统性能指标，如CPU使用率、内存使用率等。

根据硬件选型，构建硬件间的物理拓扑关系如图所示。



**图 34 硬件间的物理拓扑关系**

**6.6.2软件到硬件的映射**

（1）应用服务器映射：

硬件：LenovoThinkBook14

型号：搭载IntelCorei513500H

CPU：基准频率2.5GHz

内存：16GBRAM

存储：512GBSSD

软件：

操作系统：Windows11

Web服务器：ApacheTomcat7

应用平台：JavaSE11或更高版本

开发框架：SpringFramework5.x

（2）数据库服务器映射：

硬件：

型号：DellPowerEdgeT130

CPU：至少4核心的IntelCorei5或AMDRyzen5

内存：至少16GBDDR4RAM

存储：至少1TBSATAHDD或更高容量SSD

软件：

数据库系统：MySQL5.7

**6.6.3优化部署**

（1）硬件优化

存储**性能**优化：使用高性能的存储设备，如SSD（固态硬盘），用于提高数据访问速度，特别是对于频繁读写操作的数据库服务器。对于数据库服务器，使用RAID10配置以提供数据冗余和改善读写性能。

网络优化：升级服务器以支持10Gbps或更高速率的以太网端口，以减少网络延迟并提高数据传输速率。部署专用的网络接口卡(NIC)，以优化网络流量管理和提升网络性能。

服务器扩展性：选择具有足够扩展槽的服务器，以便未来可以增加更多的CPU、内存和存储资源。采用模块化的硬件设计，使得硬件升级和维护更加便捷。

（2）软件优化

代码和资源优化：实施代码剖析，识别并优化瓶颈代码段，减少CPU和内存资源的不必要消耗。

使用内存池和对象重用策略，减少垃圾收集的频率和影响。

数据库优化：定期进行数据库索引优化，确保查询性能。实施分区技术，对大型表进行分区，提高查询和维护效率。

应用服务器优化：配置连接池，如数据库连接池和HTTP连接池，以减少连接建立的开销。启用缓存机制，如使用Redis或Memcached，来缓存频繁访问的数据。

前端优化：实施懒加载和异步数据加载，以提高Web页面的加载速度。

（3）安全性优化

网络层面安全：在路由器和交换机上实施访问控制列表(ACLs)，限制不必要的网络流量。配置网络隔离，如使用虚拟局域网(VLAN)，将敏感区域与公共区域隔离。

系统和应用安全：定期进行安全审计和漏洞扫描，确保系统和应用程序没有安全漏洞。

实施最小权限原则，确保用户和应用程序仅具有完成其任务所需的最小权限。

数据安全：使用数据加密技术，如TLS/SSL，保护数据传输安全。对敏感数据使用透明数据加密(TDE)和磁盘加密解决方案，保护静态数据。

# 

# **7.关键质量属性的设计原理**

## 7.1 性能

（1）设计目标：保证系统能够快速响应用户请求，提供流畅的用户体验。

（2）设计策略：使用Redis缓存频繁查询的数据，采用异步处理和消息队列来优化长任务处理，对数据库查询进行优化，使用索引和查询缓存，实施负载均衡，分散用户请求到多个服务器。

## 7.2 安全性

（1）设计目标：保护系统免受未经授权的访问和各种网络威胁。

（2）设计策略：实施多层安全防护，包括防火墙、入侵检测系统和防病毒软件。使用SSL/TLS加密数据传输，保护用户数据安全。存储敏感数据时采用加密技术，如AES加密。实施定期的安全审计和漏洞扫描。

## 7.3 可用性

（1）设计目标：确保系统在高负载和部分硬件故障的情况下仍可正常运行。

（2）设计策略：设计高可用架构，包括数据库镜像和应用服务器集群。实施故障转移和灾难恢复机制。使用健康检查和自动故障恢复技术。确保关键组件如数据库和应用服务器的冗余。

## 7.4 可维护性

（1）设计目标：简化系统的维护和升级过程，降低维护成本。

（2）设计策略：采用模块化设计，便于独立更新和替换系统组件。实施自动化部署和监控，减少人工干预。维护详细的文档和代码注释。定期进行代码审查和重构。

## 7.5 可扩展性

（1）设计目标：使系统能够灵活扩展，以应对用户数量和数据量的增长。

（2）设计策略：设计松耦合的系统组件，便于水平扩展。使用云服务和虚拟化技术，以便快速扩展资源。采用微服务架构，实现服务的独立扩展。规划数据库分片和分区策略。

## 7.6 兼容性

（1）设计目标：确保系统能够兼容不同的浏览器、操作系统和设备。

（2）设计策略：使用响应式设计和跨浏览器测试。支持多种操作系统和移动设备。实施向后兼容的API设计。

## 7.7 可测试性

（1）设计目标：确保系统的所有组件都可以通过自动化测试进行验证。

（2）设计策略：实施单元测试、集成测试和系统测试。使用持续集成/持续部署(CI/CD)流程自动化测试。开发测试用例和自动化测试脚本。

## 7.8 用户体验

（1）设计目标：提供直观、易用且响应迅速的用户界面。

（2）设计策略：设计简洁直观的用户界面。实施用户反馈机制，持续改进用户体验。优化前端性能，如减少页面加载时间和提高交互响应性。