|  |  |  |  |
| --- | --- | --- | --- |
| **记录编号：** |  | **文件类别：** |  |

**serendipity社交论坛**

**系统设计说明书**

目录

[一、 系统架构设计 3](#_Toc27798)

[（一） 整体技术架构 3](#_Toc4320)

[（二） JavaFx架构 3](#_Toc21344)

[（三） Springboot架构 6](#_Toc15473)

[（四） 技术关键点 7](#_Toc29568)

[二、 面向对象设计 7](#_Toc11204)

[（一） 类图 8](#_Toc3600)

[（二） 用例图 9](#_Toc7057)

[(1) 用户用例图 9](#_Toc5333)

[(2) 管理员用例图 9](#_Toc18146)

[三、 数据库设计 10](#_Toc3483)

[（一） 数据库选型 10](#_Toc16259)

[（二） 数据库概念设计 10](#_Toc389)

[（三） 数据库逻辑设计 14](#_Toc13950)

[（1） 用户表 14](#_Toc24580)

[（3） 交互互动表 15](#_Toc7632)

[（4） 评论表 15](#_Toc14919)

[（5） 心情记录表 16](#_Toc5249)

[（6） 心情日报表 16](#_Toc9057)

[（7） 标签表 17](#_Toc22725)

[（8） 敏感信息表 17](#_Toc8832)

[（9） 管理员操作日志表 17](#_Toc6092)

[（10） AI 聊天信息表 17](#_Toc20448)

[四、 接口设计 18](#_Toc29035)

[（一） 用户模块 18](#_Toc14950)

[1. 用户注册 18](#_Toc30156)

[2. 用户登录 19](#_Toc25352)

[3. 用户信息更新 19](#_Toc21946)

[4. 发布社区交互内容 20](#_Toc17509)

[5. 获取社区交互内容列表 21](#_Toc591)

[6. 获取社区交互内容详情 22](#_Toc20545)

[7、 发布评论 22](#_Toc7946)

[（二） 管理员模块 23](#_Toc22978)

[五、 详细设计 23](#_Toc406)

[（一） 用户注册 23](#_Toc23191)

[（二） 用户登录 24](#_Toc5322)

[（三） 用户更新个人资料 24](#_Toc12926)

[（四） 用户发布社区交互内容 25](#_Toc24747)

[（五） 用户发布评论 26](#_Toc15291)

[（六） 用户点赞/收藏社区交互内容 26](#_Toc18292)

[（七） 记录心情 27](#_Toc9341)

[（八） 生成心情日报 27](#_Toc18421)

[（九） 发送AI聊天 28](#_Toc15434)

### 系统架构设计

在总体架构设计阶段，架构设计为系统的开发及整体结构打下了坚实的基础。系统总体设计是在线课程管理系统开发的关键阶段，它涉及到系统的整体架构、功能模块、技术选型、数据库设计等多个方面。从而明确系统的核心目标。确保满足用户的特定需求。

1. 整体技术架构

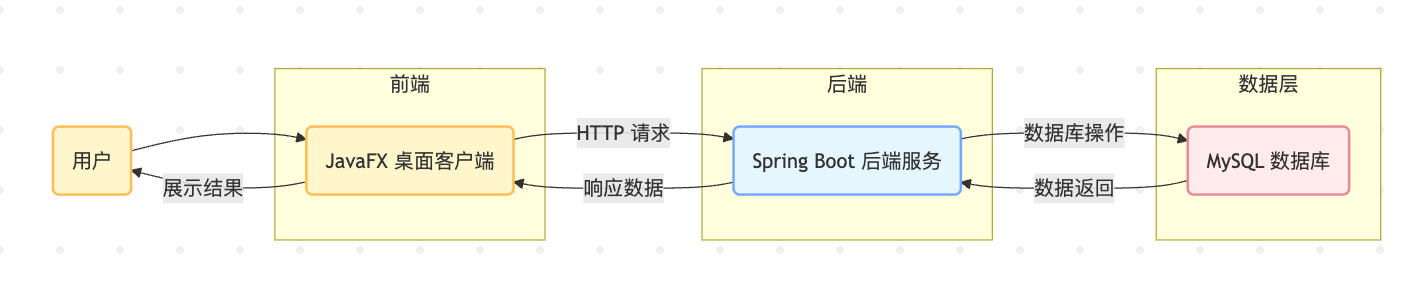
用户与 JavaFX 桌面客户端：用户通过 JavaFX 构建的桌面客户端与系统进行交互。JavaFX 提供了丰富的 UI 组件和布局管理器，能够创建美观、易用的桌面界面。

JavaFX 与 Spring Boot 后端服务：JavaFX 客户端通过 HTTP 请求与 Spring Boot 后端服务进行通信。Spring Boot 作为后端核心框架，负责处理业务逻辑、验证请求、调用数据库等操作。

Spring Boot 与 MySQL 数据库：Spring Boot 后端服务通过 JDBC 或 JPA 等技术与 MySQL 数据库进行交互，实现数据的持久化存储和查询。数据库中存储了用户信息、社区交互内容、评论等数据。

数据返回与结果展示：MySQL 数据库将查询结果返回给 Spring Boot 后端服务，后端服务处理后将响应数据返回给 JavaFX 客户端，客户端将结果展示给用户。

整体技术架构图如下



1. JavaFx架构

#### 1. 用户认证模块（登录/注册）

* **界面实现**：
  1. 使用 JavaFX 的 Stage 和 Scene 管理窗口，通过 FXML 进行界面布局（如 BorderPane、VBox、TextField、PasswordField）。
  2. 样式：通过 CSS 实现响应式设计，支持主题切换（如深色/浅色模式），使用 javafx.scene.control.Skin 自定义组件样式（如按钮涟漪效果）。
* **交互逻辑**：
  1. 登录/注册表单验证：使用 TextFormatter 或自定义 Validator 验证邮箱格式、密码复杂度。
  2. 网络请求：通过 **Java 11+ 原生 HttpClient** 或 **OkHttp** 发送 HTTP 请求（POST /api/auth/login、POST /api/auth/register），请求体为 JSON（使用 ObjectMapper 序列化/反序列化）。
  3. 线程管理：网络请求在独立线程中执行，通过 Platform.runLater() 更新 UI（避免阻塞 JavaFX 主线程）。
* **技术栈**：
  1. 布局：FXML + CSS
  2. 网络：HttpClient/OkHttp
  3. 数据绑定：javafx.beans.property（如 StringProperty 绑定输入框内容）

#### 2. 社区交互模块（动态浏览、发布、互动）

* **界面实现**：
  1. 动态列表：使用 javafx.scene.control.TableView 或自定义 ListView，结合 CellFactory 实现富文本渲染（如图片、表情解析）。
  2. 发布表单：集成富文本编辑器（如 **RichTextFX** 或自定义文本格式组件），支持图片上传（通过文件选择器 FileChooser）。
* **交互逻辑**：
  1. 动态加载：分页查询时，通过 GET /api/interactions?page=1&size=10 请求后端，接收 JSON 数据后使用 FXCollections.observableArrayList() 填充列表。
  2. 互动操作（点赞/收藏/评论）：点击按钮时发送 POST 请求（如 /api/interactions/{id}/like），更新本地 UI 后再同步后端状态（乐观锁处理并发）。
* **技术栈**：
  1. 富文本：RichTextFX
  2. 图片处理：javafx.scene.image.Image 加载本地/网络图片

#### 3. 心情记录模块（情绪日志、AI 分析）

* **界面实现**：
  1. 情绪输入：使用 ComboBox 选择表情符号，TextArea 输入心情文本，日期选择器 DatePicker 记录时间。
  2. 数据可视化：集成 **JFreeChart** 或 JavaFX 原生图表组件（LineChart、PieChart）展示情绪趋势。
* **交互逻辑**：
  1. 提交记录：封装 MoodRecord 对象为 JSON，发送 POST 请求至 /api/moods。
  2. AI 分析：后端返回情绪分析结果（如积极/消极标签），通过 Platform.runLater() 更新 UI 显示。
* **技术栈**：
  1. 图表：JFreeChart + JavaFX 集成

#### 4. 搜索与筛选模块

* **界面实现**：
  1. 搜索栏：TextField 结合 javafx.scene.control.AutoCompleteTextField 实现搜索建议（本地缓存热门标签）。
  2. 筛选条件：CheckBox、ChoiceBox 选择分类/标签，支持多条件组合。
* **交互逻辑**：
  1. 实时搜索：输入框监听 textProperty().addListener()，延迟 300ms 后发送 GET 请求（带 keyword、classificationId 等参数）。
  2. 结果排序：通过 ComboBox 选择排序规则（时间/热度/相关度），拼接请求参数（如 sort=likeCount,desc）。

1. Springboot架构

#### 基础架构

* **分层设计**：
  1. **控制器层（Controller）**：使用 @RestController + @RequestMapping 定义 RESTful 接口，请求参数通过 @RequestBody（JSON）或 @RequestParam（查询参数）接收。
  2. **服务层（Service）**：封装业务逻辑，使用 @Service 注解，通过 @Transactional 管理数据库事务（如发布动态时同时更新用户积分）。
  3. **数据访问层（Repository）**：基于 **Spring Data JPA**，定义 @Repository 接口继承 JpaRepository 或 JpaSpecificationExecutor，实现复杂查询（如动态条件搜索）。
* **技术栈**：
  1. 依赖管理：Maven/Gradle + Spring Boot Starter（web、data-jpa、security）
  2. 数据库连接：HikariCP 连接池（默认配置）

1. 技术关键点

|  |  |  |  |
| --- | --- | --- | --- |
| **模块** | **前端（JavaFX）技术** | **后端（Spring Boot）技术** | **数据库技术** |
| **网络通信** | HttpClient/OkHttp、JSON 序列化（Jackson） | RestController、@RequestBody、异步请求处理 | JDBC 连接池（HikariCP） |
| **用户认证** | FXML 表单验证、JWT 令牌存储（本地/内存） | Spring Security、JWT 生成与校验、BCrypt 加密 | 唯一索引（username/email） |
| **数据绑定** | JavaFX 属性绑定（ObservableList） | Spring Data JPA、自定义 Repository 方法 | 外键约束、事务管理 |
| **性能优化** | 分页加载、图片懒加载 | 缓存（@Cacheable）、异步任务（@Async） | 索引优化、读写分离 |
| **安全防护** | 输入过滤、HTTPS 通信 | XSS 防护（Jsoup）、CSRF 令牌、SQL 注入防护 | 密码加密存储、敏感词标记 |

### 面向对象设计

面向对象分析与设计（OOAD）是一种软件工程方法论，它采用面向对象的观点来分析和设计软件系统。OOAD以类和对象作为基本构造块，通过封装、继承和多态等原则，将系统分解为相互协作的对象。这种方法强调系统的模块化和重用性，使得软件设计更加灵活和易于维护。统一建模语言（UML）是OOAD中常用的图形化建模工具，它提供了一套标准的图表，如用例图、类图、序列图等，来可视化系统的需求和设计。UML帮助开发人员理解和沟通系统的结构和行为，促进团队协作，并为软件开发过程提供文档支持。OOAD结合UML，使得系统分析与设计更加系统化、结构化，有助于提高软件质量和开发效率。

1. 类图

系统类图提供了系统核心组件及其交互的结构化概览。它旨在捕捉每个类的基本属性和方法，反映系统的功能和数据管理能力。

用户类是系统的核心，与多个类关联，体现了用户在系统中的多种行为和数据记录。用户可以发布社区交互内容（与社区交互类关联）、发表评论（与评论类关联）、记录心情（与心情记录类关联）、生成心情日报（与心情日报类关联）、发送AI聊天消息（与AI聊天信息类关联），并且可能会产生敏感信息（与敏感信息类关联）。管理员也可以通过管理员操作日志类对用户进行管理。

社区交互类与用户类通过作者ID关联，明确社区交互内容的归属；同时与交互互动类和评论类关联，记录社区交互内容的互动数据和用户评论，形成了社区内容的社交生态。

交互互动类通过交互ID与社区交互类关联，记录社区交互内容的点赞数和收藏数，反映了内容的受欢迎程度。

评论类通过评论者ID与用户类关联，通过交互ID与社区交互类关联，构建了用户对社区交互内容的反馈机制，并且通过回复的评论ID形成了评论的嵌套结构。

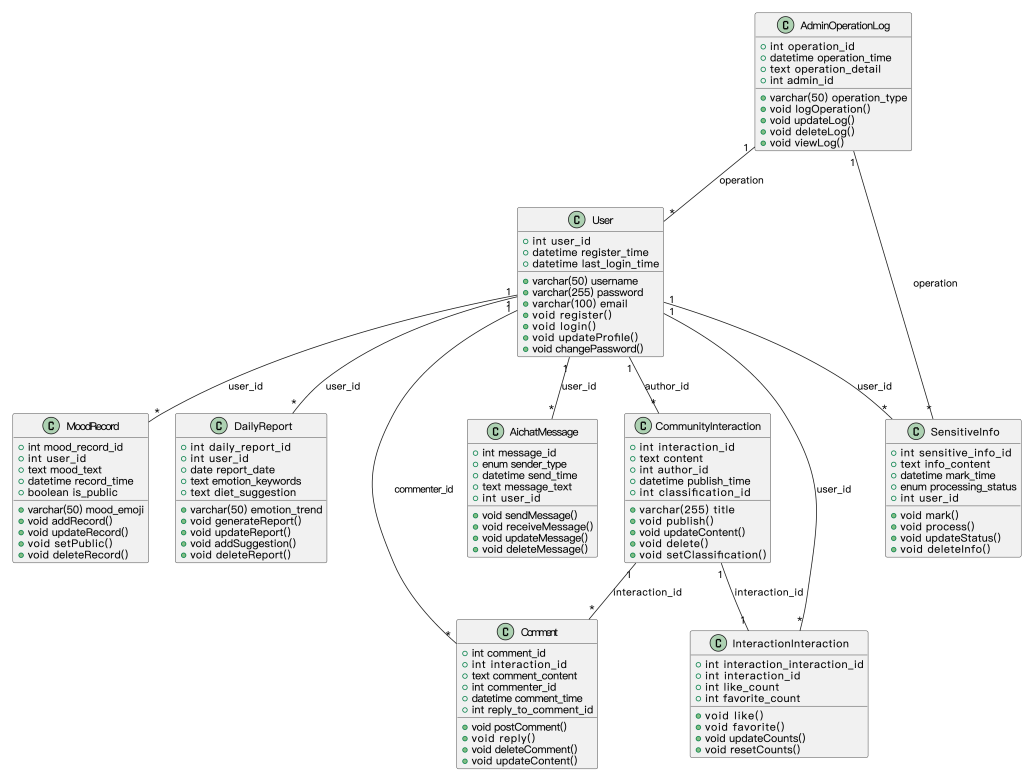
心情记录类和心情日报类都与用户类关联，分别记录用户的心情状态和生成心情日报，为用户提供情绪管理的功能。

敏感信息类与用户类关联，记录用户产生的敏感信息；同时，管理员可以通过管理员操作日志类对敏感信息进行管理，确保平台的内容安全。

管理员操作日志类与用户类和敏感信息类关联，记录管理员对用户和敏感信息的操作，提供了操作的可追溯性和管理功能。

AI聊天信息类与用户类关联，记录用户与AI的聊天信息，为用户提供AI互动功能。

这些关联关系构成了系统的数据结构和功能逻辑，使得系统能够实现丰富的功能和良好的用户体验。下面是系统类图如下。

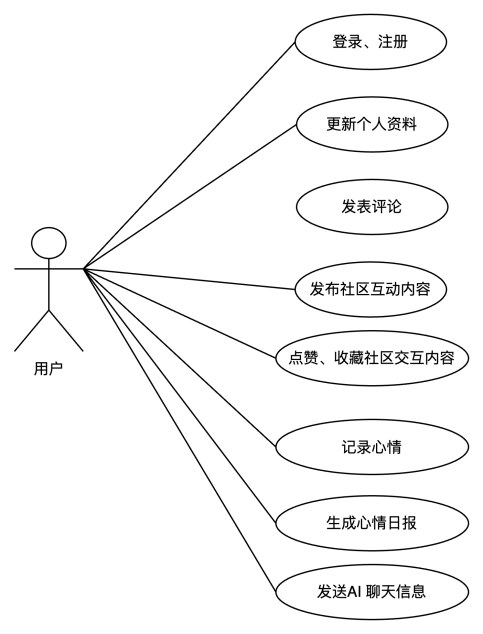


1. 用例图

用例图是一种UML（统一建模语言）图，用于描述系统功能及其与外部参与者（如用户、管理员等）的交互关系。它通过用例（系统功能）和参与者（使用系统的角色）来展示系统的功能范围和交互场景。用例图的作用在于帮助开发团队明确系统需求，梳理功能模块，确保系统设计符合用户期望，同时便于与非技术人员沟通系统功能，促进项目理解和协作。下面从两个角色描述系统的用例。

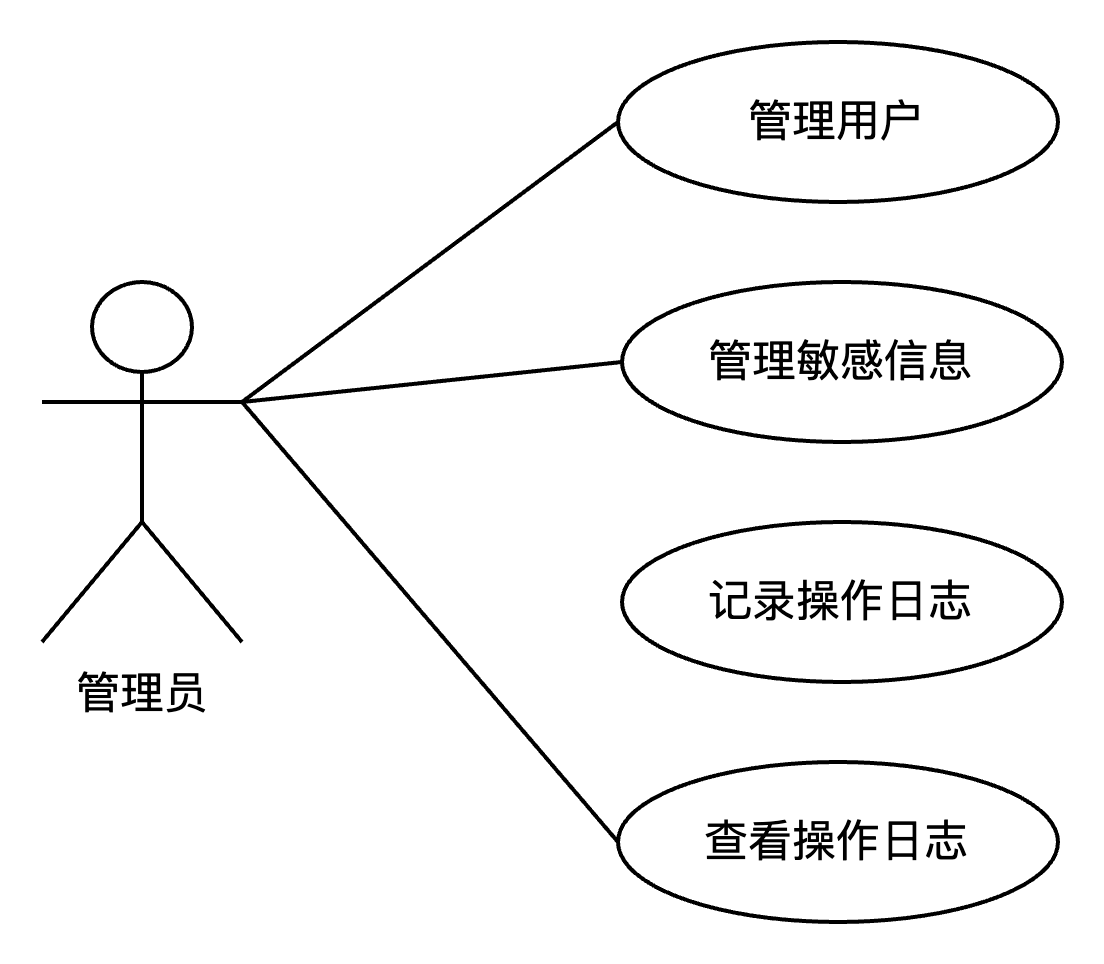
1. 用户用例图

用户用例图展示了用户在系统中可以执行的各种功能。用户可以进行注册和登录操作，以创建和访问自己的账户。用户还可以更新自己的个人资料和修改密码，以管理自己的账户信息。此外，用户可以发布社区交互内容（如帖子），发表评论，对社区交互内容进行点赞或收藏，记录自己的心情，生成心情日报，以及发送AI聊天消息。这些用例涵盖了用户在社区平台中的主要活动，包括内容发布、社交互动、情绪记录和AI互动等功能。



1. 管理员用例图

管理员用例图展示了管理员在系统中可以执行的管理功能。管理员可以管理用户，例如封禁或解封用户账户、审核用户内容等。管理员还可以管理敏感信息，包括标记、处理和更新敏感信息的状态。此外，管理员可以记录操作日志，详细记录每次管理操作的时间、类型和详情；同时，管理员也可以查看操作日志，以便追溯和审计管理活动。这些用例体现了管理员在维护平台安全、内容质量和操作记录方面的重要职责。



### 数据库设计

数据库设计是信息系统中至关重要的一环，其目的和意义主要体现在以下几个方面：数据库设计的目的在于构建一个高效、可靠且易于维护的数据存储和管理系统。通过精心设计的数据库结构，可以更好地组织和存储企业或组织的相关数据，确保数据的完整性、一致性和准确性。 同时，它还便于数据的维护和管理，使数据的更新、插入和删除更加方便快捷，降低了系统维护的成本和难度。总而言之，数据库设计是实现信息系统高效运行、数据安全可靠的基础，对于在线课堂的发展和竞争力的提升具有重要的意义。

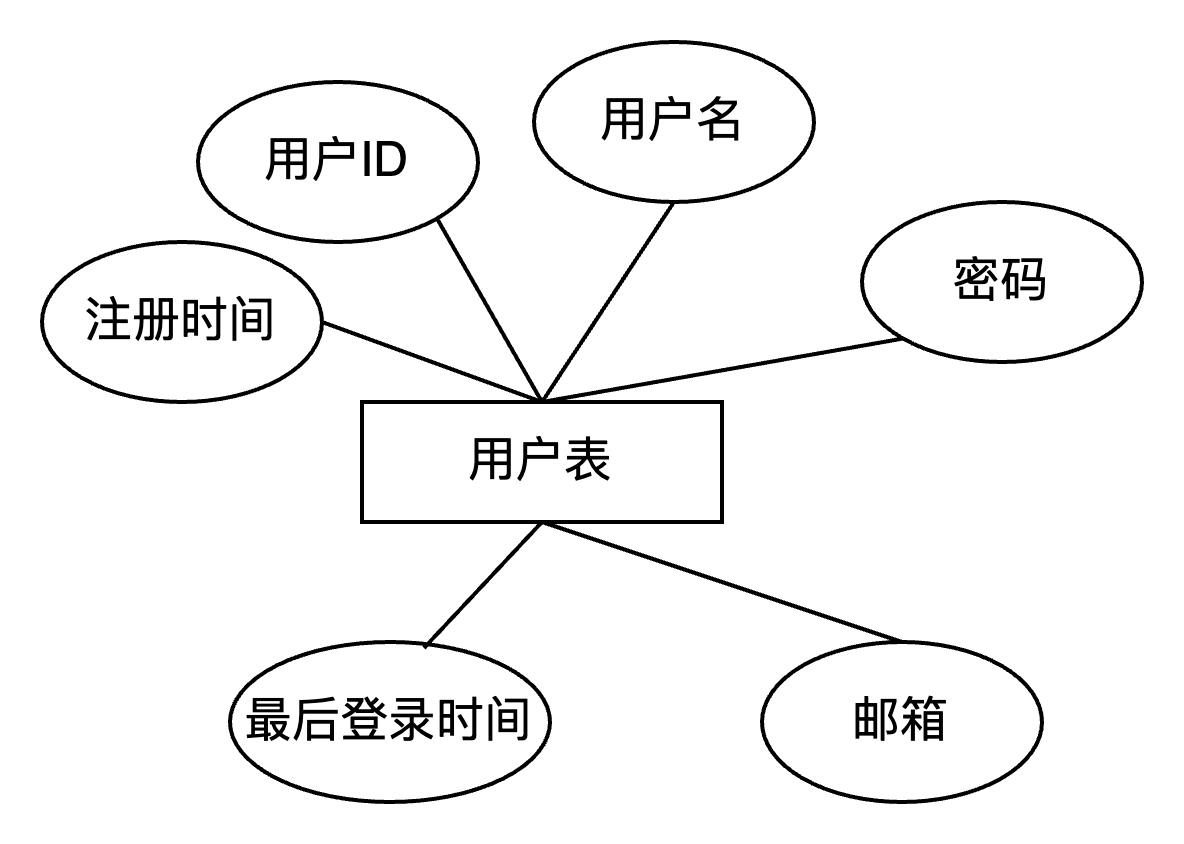
1. 数据库选型

在本系统的设计中，我们经过综合考虑和分析，最终选择了 [MySQL](coco://sendMessage?ext={"s$wiki_link":"https://m.baike.com/wikiid/6782231522138548878"}&msg=MySQL" \t "/Users/liuting/Documents\\x/_blank) 作为数据库。选择 MySQL 主要是基于以下几个重要的目的和原因：首先，MySQL 具有强大的性能和高效的存储能力，能够满足本系统对数据处理和存储的要求，确保系统在处理大量数据时能够保持快速响应和稳定运行。其次，它是开源且免费的，这为我们的项目节省了成本，同时也使得我们能够更加自由地进行定制和优化，以满足系统的特殊需求。

1. 数据库概念设计

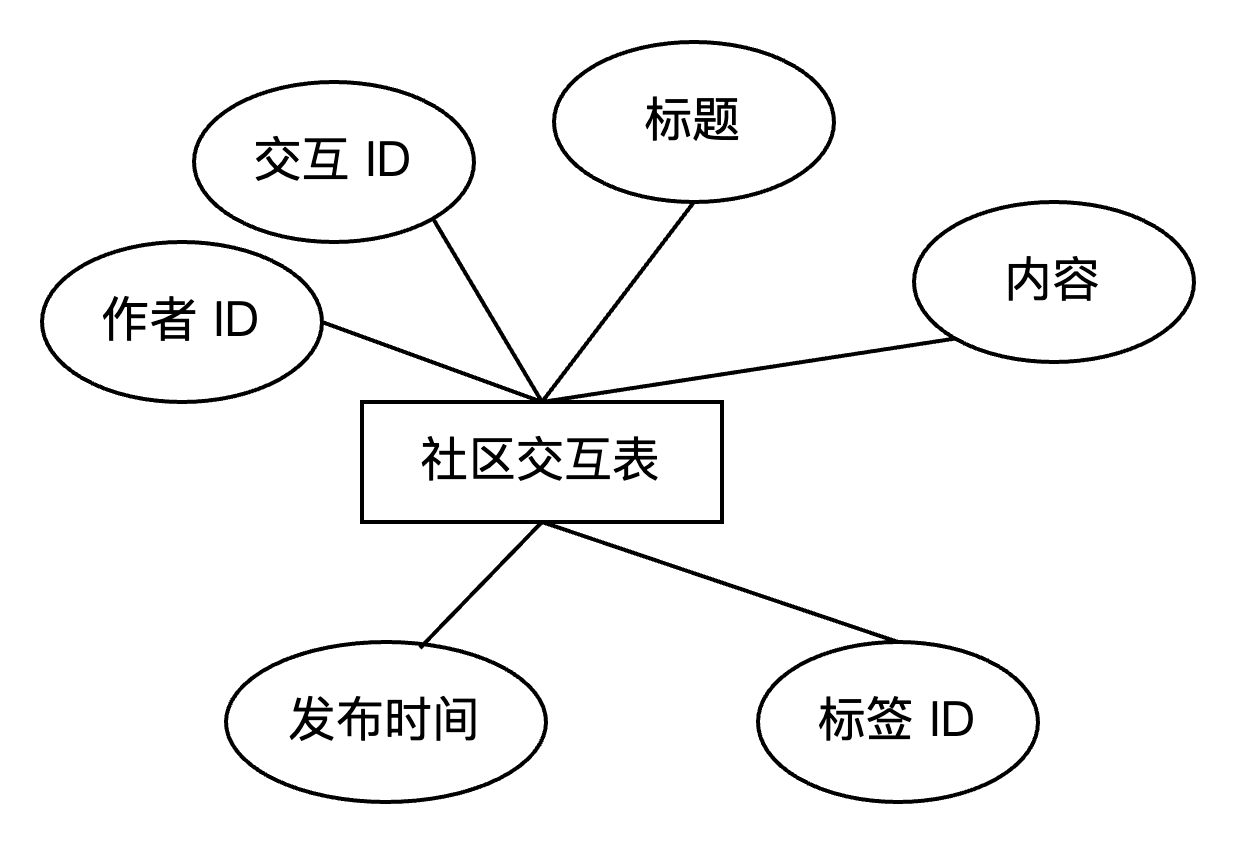
一个数据库的设计是否优秀，在很大程度上决定了系统的成功与否。概念设计是数据库设计的核心环节。通过对用户需求进行整合、抽象与总结，形成一个独立于具体数据库管理系统的概念模型。有了ER图，就能根据ER图快速建立数据库。

1. 用户实体图

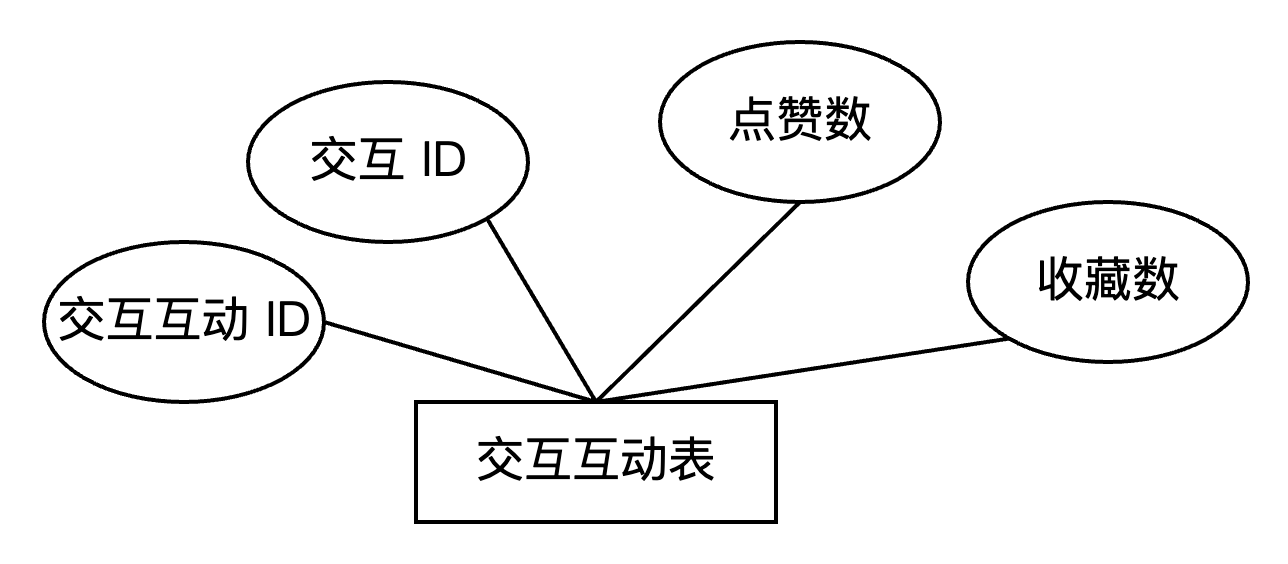


1. 社区交互实体图

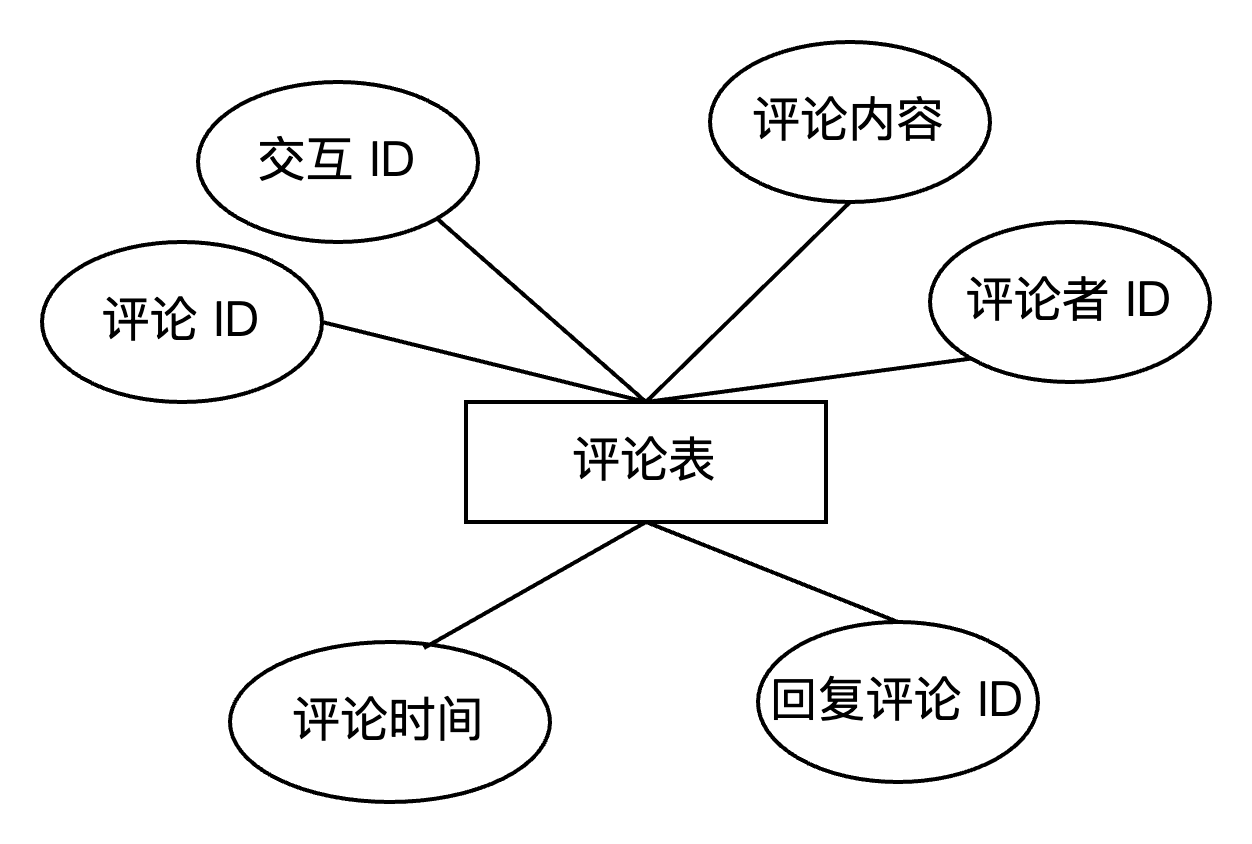
%3CmxGraphModel%3E%3Croot%3E%3CmxCell%20id%3D%220%22%2F%3E%3CmxCell%20id%3D%221%22%20parent%3D%220%22%2F%3E%3CmxCell%20id%3D%222%22%20style%3D%22rounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3BexitX%3D0.5%3BexitY%3D0%3BexitDx%3D0%3BexitDy%3D0%3BentryX%3D0.5%3BentryY%3D1%3BentryDx%3D0%3BentryDy%3D0%3BendArrow%3Dnone%3BstartFill%3D0%3BfontSize%3D12%3B%22%20edge%3D%221%22%20source%3D%227%22%20target%3D%229%22%20parent%3D%221%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%223%22%20style%3D%22edgeStyle%3Dnone%3Bshape%3Dconnector%3Brounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3BexitX%3D0.5%3BexitY%3D0%3BexitDx%3D0%3BexitDy%3D0%3BentryX%3D0%3BentryY%3D1%3BentryDx%3D0%3BentryDy%3D0%3BstrokeColor%3Ddefault%3Balign%3Dcenter%3BverticalAlign%3Dmiddle%3BfontFamily%3DHelvetica%3BfontSize%3D12%3BfontColor%3Ddefault%3BlabelBackgroundColor%3Ddefault%3BstartFill%3D0%3BendArrow%3Dnone%3B%22%20edge%3D%221%22%20source%3D%227%22%20target%3D%2210%22%20parent%3D%221%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%224%22%20style%3D%22edgeStyle%3Dnone%3Bshape%3Dconnector%3Brounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3BexitX%3D0.5%3BexitY%3D0%3BexitDx%3D0%3BexitDy%3D0%3BentryX%3D1%3BentryY%3D1%3BentryDx%3D0%3BentryDy%3D0%3BstrokeColor%3Ddefault%3Balign%3Dcenter%3BverticalAlign%3Dmiddle%3BfontFamily%3DHelvetica%3BfontSize%3D12%3BfontColor%3Ddefault%3BlabelBackgroundColor%3Ddefault%3BstartFill%3D0%3BendArrow%3Dnone%3B%22%20edge%3D%221%22%20source%3D%227%22%20target%3D%228%22%20parent%3D%221%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%225%22%20style%3D%22edgeStyle%3Dnone%3Bshape%3Dconnector%3Brounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3BexitX%3D0.5%3BexitY%3D0%3BexitDx%3D0%3BexitDy%3D0%3BentryX%3D1%3BentryY%3D0.5%3BentryDx%3D0%3BentryDy%3D0%3BstrokeColor%3Ddefault%3Balign%3Dcenter%3BverticalAlign%3Dmiddle%3BfontFamily%3DHelvetica%3BfontSize%3D12%3BfontColor%3Ddefault%3BlabelBackgroundColor%3Ddefault%3BstartFill%3D0%3BendArrow%3Dnone%3B%22%20edge%3D%221%22%20source%3D%227%22%20target%3D%2212%22%20parent%3D%221%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%226%22%20style%3D%22edgeStyle%3Dnone%3Bshape%3Dconnector%3Brounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3BexitX%3D0.5%3BexitY%3D1%3BexitDx%3D0%3BexitDy%3D0%3BentryX%3D0.5%3BentryY%3D0%3BentryDx%3D0%3BentryDy%3D0%3BstrokeColor%3Ddefault%3Balign%3Dcenter%3BverticalAlign%3Dmiddle%3BfontFamily%3DHelvetica%3BfontSize%3D12%3BfontColor%3Ddefault%3BlabelBackgroundColor%3Ddefault%3BstartFill%3D0%3BendArrow%3Dnone%3B%22%20edge%3D%221%22%20source%3D%227%22%20target%3D%2211%22%20parent%3D%221%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%227%22%20value%3D%22%E7%A4%BE%E5%8C%BA%E4%BA%A4%E4%BA%92%E8%A1%A8%22%20style%3D%22rounded%3D0%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3BfontSize%3D12%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%22900%22%20y%3D%22575%22%20width%3D%2290%22%20height%3D%2230%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%228%22%20value%3D%22%26lt%3Bp%20style%3D%26quot%3Bmargin%3A%200pt%200pt%200.0001pt%3B%20text-align%3A%20justify%3B%20font-family%3A%20Calibri%3B%26quot%3B%20class%3D%26quot%3BMsoNormal%26quot%3B%26gt%3B%26lt%3Bspan%26gt%3B%26lt%3Bfont%20face%3D%26quot%3B%E5%AE%8B%E4%BD%93%26quot%3B%26gt%3B%E4%BA%A4%E4%BA%92%26lt%3B%2Ffont%26gt%3B%20ID%26lt%3B%2Fspan%26gt%3B%26lt%3B%2Fp%26gt%3B%22%20style%3D%22ellipse%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3BfontSize%3D12%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%22858%22%20y%3D%22495%22%20width%3D%2270%22%20height%3D%2240%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%229%22%20value%3D%22%26lt%3Bp%20style%3D%26quot%3Bmargin%3A%200pt%200pt%200.0001pt%3B%20text-align%3A%20justify%3B%20font-family%3A%20Calibri%3B%26quot%3B%20class%3D%26quot%3BMsoNormal%26quot%3B%26gt%3B%26lt%3Bspan%26gt%3B%26lt%3Bfont%20face%3D%26quot%3B%E5%AE%8B%E4%BD%93%26quot%3B%26gt%3B%E6%A0%87%E9%A2%98%26lt%3B%2Ffont%26gt%3B%26lt%3B%2Fspan%26gt%3B%26lt%3B%2Fp%26gt%3B%22%20style%3D%22ellipse%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3BfontSize%3D12%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%22948%22%20y%3D%22485%22%20width%3D%2270%22%20height%3D%2240%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%2210%22%20value%3D%22%26lt%3Bp%20style%3D%26quot%3Bmargin%3A%200pt%200pt%200.0001pt%3B%20text-align%3A%20justify%3B%20font-family%3A%20Calibri%3B%26quot%3B%20class%3D%26quot%3BMsoNormal%26quot%3B%26gt%3B%26lt%3Bspan%26gt%3B%26lt%3Bfont%20face%3D%26quot%3B%E5%AE%8B%E4%BD%93%26quot%3B%26gt%3B%E5%86%85%E5%AE%B9%26lt%3B%2Ffont%26gt%3B%26lt%3B%2Fspan%26gt%3B%26lt%3B%2Fp%26gt%3B%22%20style%3D%22ellipse%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3BfontSize%3D12%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%221038%22%20y%3D%22525%22%20width%3D%2270%22%20height%3D%2240%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%2211%22%20value%3D%22%26lt%3Bp%20style%3D%26quot%3Bmargin%3A%200pt%200pt%200.0001pt%3B%20text-align%3A%20justify%3B%26quot%3B%20class%3D%26quot%3BMsoNormal%26quot%3B%26gt%3B%26lt%3Bfont%20face%3D%26quot%3B%E5%AE%8B%E4%BD%93%26quot%3B%26gt%3B%E6%A0%87%E7%AD%BE%26lt%3B%2Ffont%26gt%3B%26lt%3Bfont%20face%3D%26quot%3BCalibri%26quot%3B%26gt%3B%20ID%26lt%3B%2Ffont%26gt%3B%26lt%3B%2Fp%26gt%3B%22%20style%3D%22ellipse%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3BfontSize%3D12%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%22999%22%20y%3D%22635%22%20width%3D%2270%22%20height%3D%2240%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%2212%22%20value%3D%22%26lt%3Bp%20style%3D%26quot%3Bmargin%3A%200pt%200pt%200.0001pt%3B%20text-align%3A%20justify%3B%20font-family%3A%20Calibri%3B%26quot%3B%20class%3D%26quot%3BMsoNormal%26quot%3B%26gt%3B%26lt%3Bspan%26gt%3B%26lt%3Bfont%20face%3D%26quot%3B%E5%AE%8B%E4%BD%93%26quot%3B%26gt%3B%E4%BD%9C%E8%80%85%26lt%3B%2Ffont%26gt%3B%20ID%26lt%3B%2Fspan%26gt%3B%26lt%3B%2Fp%26gt%3B%22%20style%3D%22ellipse%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3BfontSize%3D12%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%22820%22%20y%3D%22535%22%20width%3D%2270%22%20height%3D%2240%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%2213%22%20value%3D%22%26lt%3Bp%20style%3D%26quot%3Bmargin%3A%200pt%200pt%200.0001pt%3B%20text-align%3A%20justify%3B%20font-family%3A%20Calibri%3B%26quot%3B%20class%3D%26quot%3BMsoNormal%26quot%3B%26gt%3B%26lt%3Bspan%26gt%3B%26lt%3Bfont%20face%3D%26quot%3B%E5%AE%8B%E4%BD%93%26quot%3B%26gt%3B%E5%8F%91%E5%B8%83%E6%97%B6%E9%97%B4%26lt%3B%2Ffont%26gt%3B%26lt%3B%2Fspan%26gt%3B%26lt%3B%2Fp%26gt%3B%22%20style%3D%22ellipse%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3BfontSize%3D12%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%22866%22%20y%3D%22635%22%20width%3D%2280%22%20height%3D%2240%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%2214%22%20style%3D%22edgeStyle%3Dnone%3Bshape%3Dconnector%3Brounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3BexitX%3D0.5%3BexitY%3D1%3BexitDx%3D0%3BexitDy%3D0%3BentryX%3D0.606%3BentryY%3D0.038%3BentryDx%3D0%3BentryDy%3D0%3BentryPerimeter%3D0%3BstrokeColor%3Ddefault%3Balign%3Dcenter%3BverticalAlign%3Dmiddle%3BfontFamily%3DHelvetica%3BfontSize%3D12%3BfontColor%3Ddefault%3BlabelBackgroundColor%3Ddefault%3BstartFill%3D0%3BendArrow%3Dnone%3B%22%20edge%3D%221%22%20source%3D%227%22%20target%3D%2213%22%20parent%3D%221%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3C%2Froot%3E%3C%2FmxGraphModel%3E



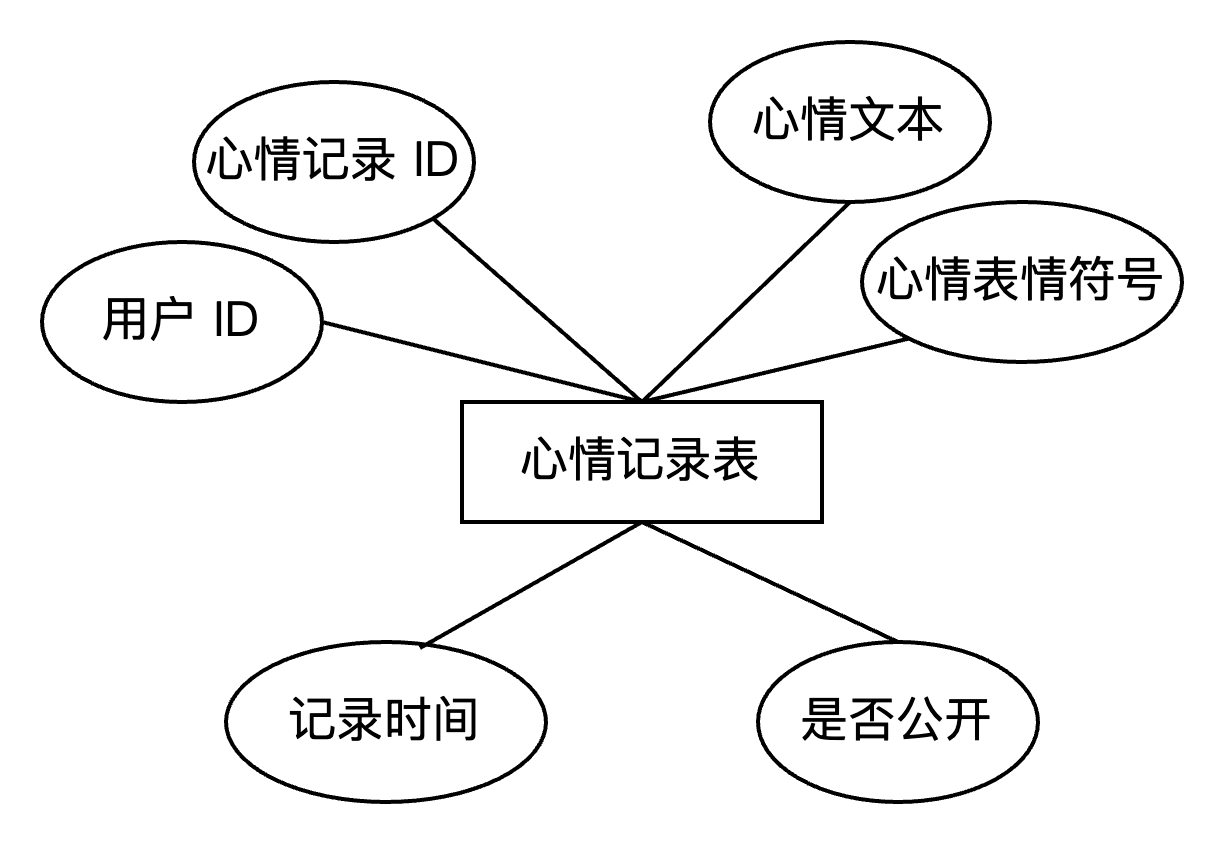
1. 互动实体图



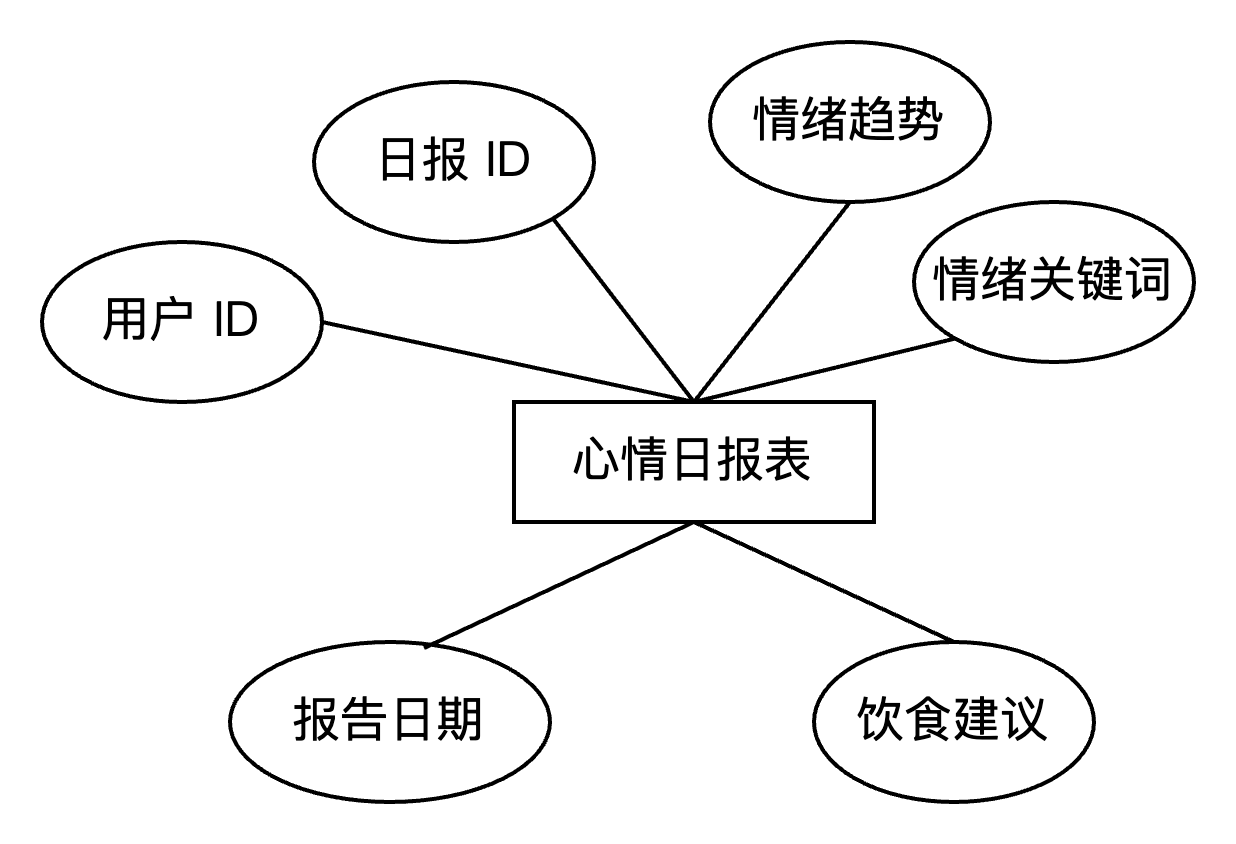
1. 评论实体图



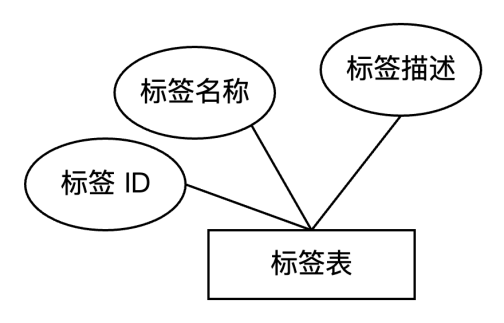
1. 心情记录实体图



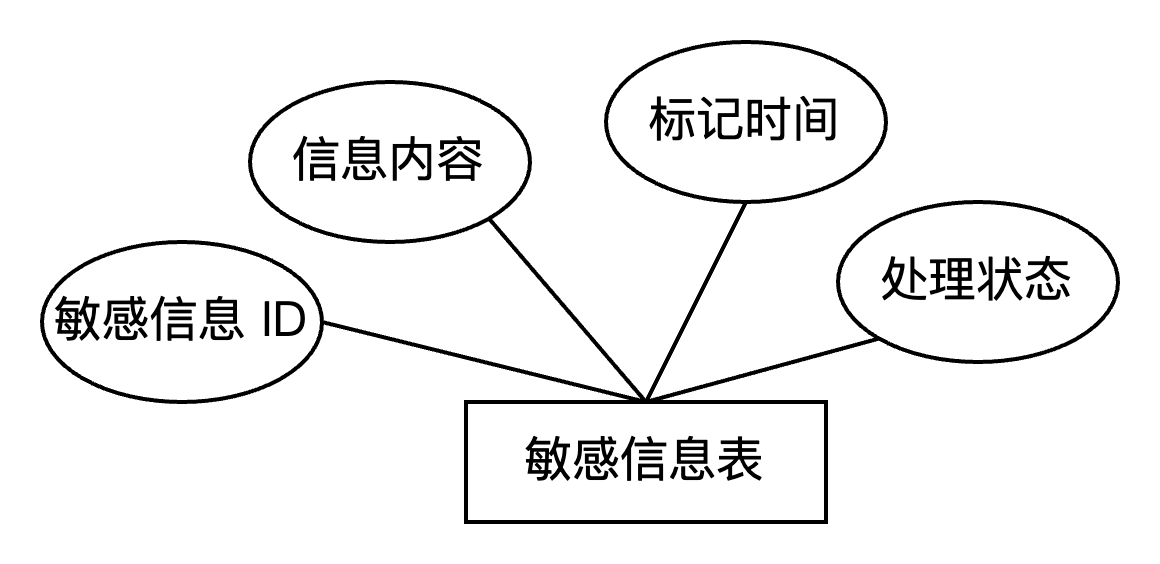
1. 心情日报实体图



1. 标签实体图



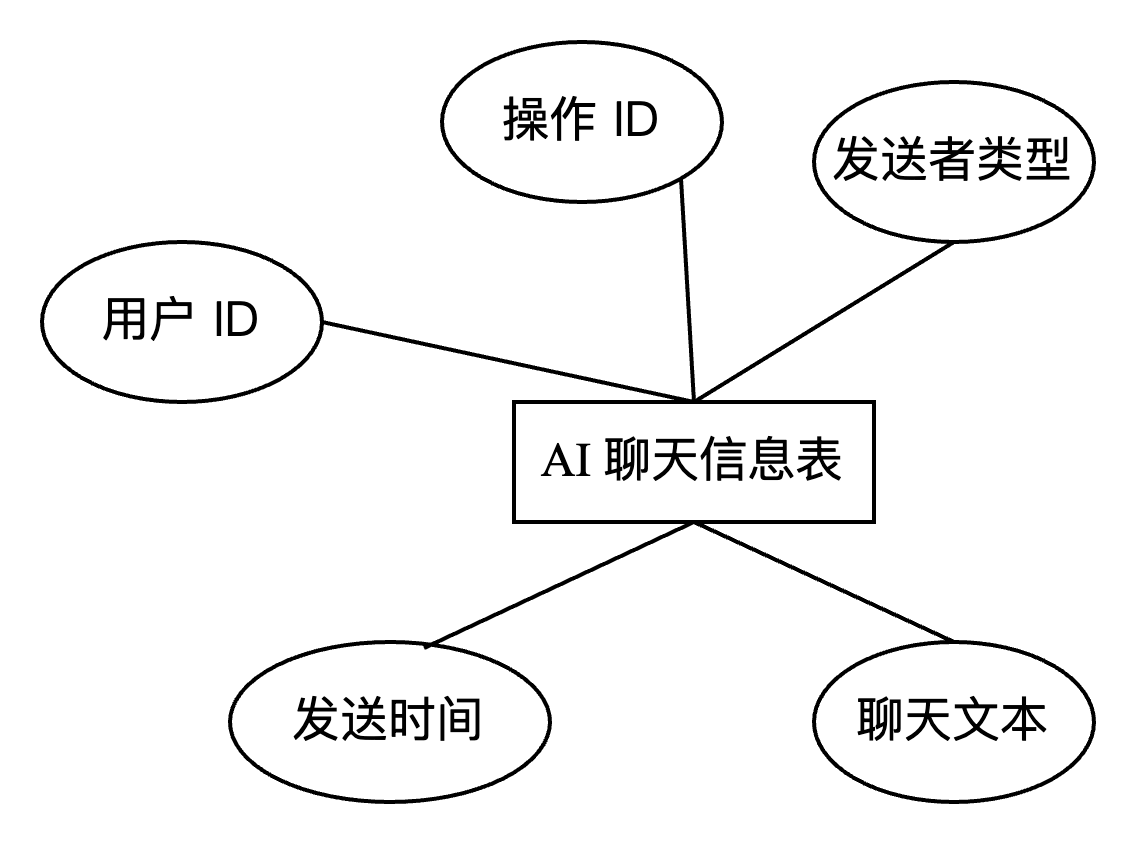
1. 敏感信息实体图



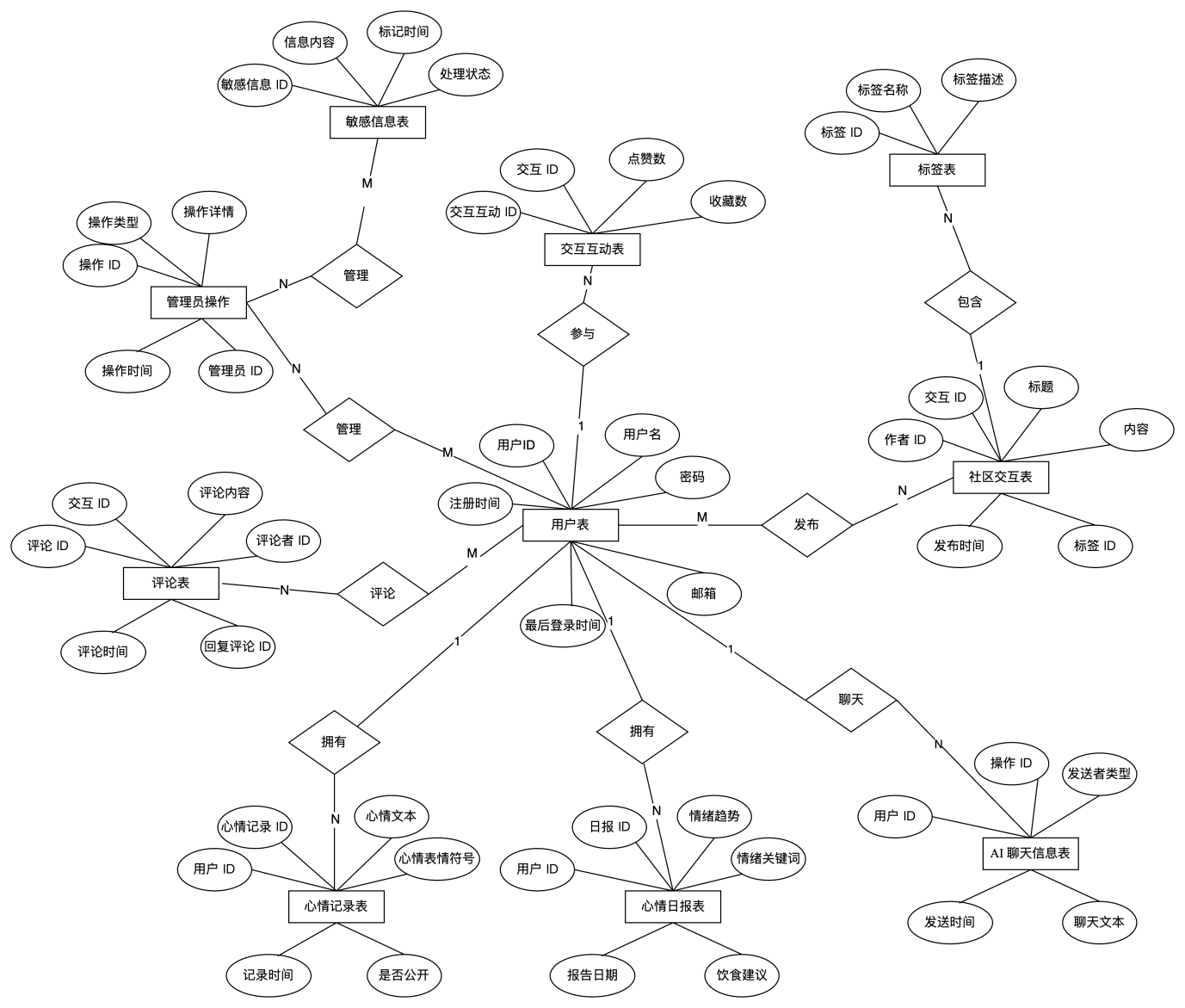
1. 管理员操作日志实体图



1. AI 聊天信息实体图



根据上述实体属性图，构建出整个系统的 E-R 模型如图所示。



1. 数据库逻辑设计

数据库逻辑设计是数据库设计过程中的重要环节，主要关注于将数据抽象为计算机可以理解和处理的数据模型。

本系统主要数据部分项目采用了关系型数据库MYSQL来进行存储。下面选取几张核心数据表进行展示和说明。

* + 1. 用户表

存储用户的基本信息.如表1所示。

表1 用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| user\_id | 用户 ID | INT 自增主键 | 唯一标识用户的编号 |
| username | 用户名 | VARCHAR(50) 非空唯一 | 用户在平台使用的名称，不能重复 |
| password | 密码 | VARCHAR(255) 非空 | 用户登录的密码 |
| email | 邮箱 | VARCHAR(100) 非空唯一 | 用户注册时使用的邮箱，不能重复 |
| register\_time | 注册时间 | DATETIME 默认当前时间 | 用户注册到平台的时间，默认为注册时的当前时间 |
| last\_login\_time | 最后登录时间 | DATETIME | 用户最后一次登录平台的时间 |

#### 社区交互表

存储社区交互的相关内容，如帖子。参照表2。

表2 社区表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| interaction\_id | 交互 ID | INT 自增主键 | 唯一标识社区交互内容的编号 |
| title | 标题 | VARCHAR(255) 非空 | 社区交互内容的标题 |
| content | 内容 | TEXT | 社区交互内容的详细文本信息 |
| author\_id | 作者 ID | INT 外键 | 发布该社区交互内容的用户 ID |
| publish\_time | 发布时间 | DATETIME 默认当前时间 | 社区交互内容发布的时间，默认为发布时的当前时间 |
| classification\_id | 分类 ID | INT 外键 | 该社区交互内容所属的分类 ID |

* + 1. 交互互动表

存储社区交互内容的互动数据，如点赞数、收藏数。参照表3。

表3 交互互动表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| interaction\_interaction\_id | 交互互动 ID | INT 自增主键 | 唯一标识社区交互互动记录的编号 |
| interaction\_id | 交互 ID | INT 外键 | 关联的社区交互内容的 ID |
| like\_count | 点赞数 | INT 默认 0 | 该社区交互内容的点赞数量，默认为 0 |
| favorite\_count | 收藏数 | INT 默认 0 | 该社区交互内容的收藏数量，默认为 0 |

* + 1. 评论表

用于存储对社区交互内容的评论信息。参照表4。

表4评论表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| comment\_id | 评论 ID | INT 自增主键 | 唯一标识评论的编号 |
| interaction\_id | 交互 ID | INT 外键 | 该评论所针对的社区交互内容的 ID |
| comment\_content | 评论内容 | TEXT | 评论的具体文本内容 |
| commenter\_id | 评论者 ID | INT 外键 | 发表该评论的用户 ID |
| comment\_time | 评论时间 | DATETIME 默认当前时间 | 评论发表的时间，默认为发表时的当前时间 |
| reply\_to\_comment\_id | 回复评论 ID | INT 外键 | 如果该评论是回复其他评论的，则关联被回复评论的 comment\_id，可空 |

* + 1. 心情记录表

用于存储用户的心情记录信息。参照表5。

表5心情记录表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| mood\_record\_id | 心情记录 ID | INT 自增主键 | 唯一标识心情记录的编号 |
| user\_id | 用户 ID | INT 外键 | 创建该心情记录的用户 ID，关联 users 表的 user\_id |
| mood\_text | 心情文本 | TEXT | 用户描述心情的文本内容 |
| mood\_emoji | 心情表情符号 | VARCHAR(50) | 代表用户心情的表情符号 |
| record\_time | 记录时间 | DATETIME 默认当前时间 | 用户记录心情的时间，默认为记录时的当前时间 |
| is\_public | 是否公开 | BOOLEAN 默认 FALSE | 该心情记录是否对其他用户公开，默认为不公开 |

* + 1. 心情日报表

用于存储用户的心情日报信息。参照表6。

表.6 心情日报表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| daily\_report\_id | 日报 ID | INT 自增主键 | 唯一标识心情日报的编号 |
| user\_id | 用户 ID | INT 外键 | 该心情日报所属的用户 ID，关联 users 表的 user\_id |
| report\_date | 报告日期 | DATE | 心情日报对应的日期 |
| emotion\_trend | 情绪趋势 | VARCHAR(50) | 该日用户情绪的总体趋势描述 |
| emotion\_keywords | 情绪关键词 | TEXT | 描述该日用户情绪的关键词 |
| diet\_suggestion | 饮食建议 | TEXT | 根据用户情绪给出的饮食方面的建议 |

* + 1. 标签表

存储社区交互内容的标签信息。参照表7

表7 标签表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| tag\_id | 标签 ID | INT 自增主键 | 唯一标识标签的编号 |
| tag\_name | 标签名称 | VARCHAR(50) 非空 | 标签的名称，如 “Python”、“旅游” 等 |
| tag\_description | 标签描述 | TEXT | 对该标签的详细说明 |

* + 1. 敏感信息表

存储平台检测到的敏感信息。参照表8。

表8敏感信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| sensitive\_info\_id | 敏感信息 ID | INT 自增主键 | 唯一标识敏感信息记录的编号 |
| info\_content | 信息内容 | TEXT | 敏感信息的具体内容 |
| mark\_time | 标记时间 | DATETIME 默认当前时间 | 该敏感信息被标记的时间，默认为标记时的当前时间 |
| processing\_status | 处理状态 | ENUM('unprocessed', 'processed') 默认 'unprocessed' | 敏感信息的处理状态，默认为未处理 |

* + 1. 管理员操作日志表

记录管理员的操作信息。参照表9。

表9 管理员操作日志表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| operation\_id | 操作 ID | INT 自增主键 | 唯一标识操作记录的编号 |
| operation\_type | 操作类型 | VARCHAR(50) | 管理员操作的类型，如 “删除帖子”、“封禁用户” 等 |
| operation\_time | 操作时间 | DATETIME 默认当前时间 | 管理员进行操作的时间，默认为操作时的当前时间 |
| operation\_detail | 操作详情 | TEXT | 对操作的详细描述 |
| admin\_id | 管理员 ID | INT 外键 | 执行该操作的管理员 ID |

* + 1. AI 聊天信息表

记录AI 聊天信息。参照表10。

表10 AI 聊天信息表

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 中文释义 | 类型 | 说明 |
| message\_id | 操作 ID | INT 自增主键 | 唯一标识ai会话的编号 |
| sender\_type | 发送者类型 | enum | 发送信息操作者的类型，如 “客户端”、“ai” 等 |
| send\_time | 发送时间 | DATETIME 默认当前时间 | 发送操作时的当前时间 |
| message\_text | 聊天文本 | TEXT | 对文本的详细描述 |
| user\_id | 用户 ID | INT 外键 | 该聊天所属的用户 ID，关联 users 表的 user\_id |

### 接口设计

1. 用户模块

1. 用户注册

接口地址：/api/users/register

请求方法：POST

请求参数：

username: 用户名，字符串类型，必填

password: 密码，字符串类型，必填

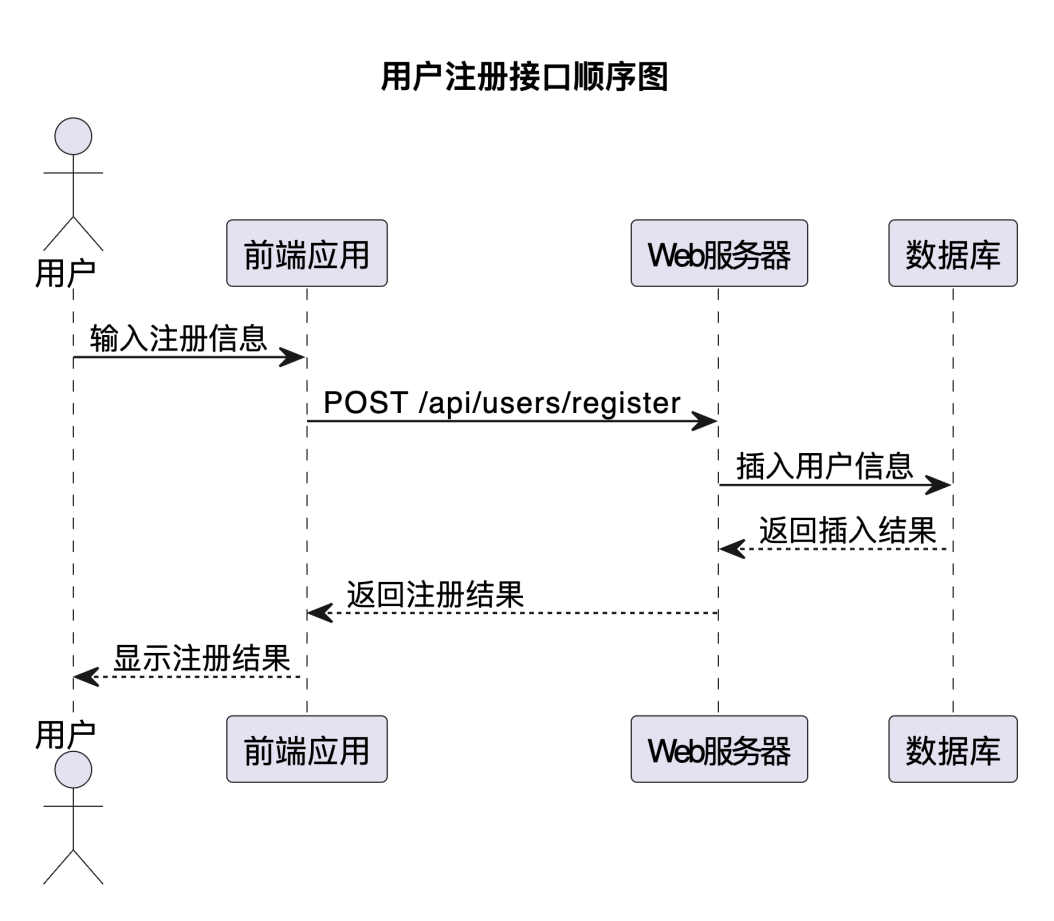
email: 邮箱，字符串类型，必填

registerCode: 注册码，字符串类型，必填

响应数据：

code: 状态码，整数类型

message: 消息，字符串类型

data: 注册成功的用户 ID，整数类型

2. 用户登录

接口地址：/api/users/login

请求方法：POST

请求参数：

username: 用户名，字符串类型，必填

password: 密码，字符串类型，必填

响应数据：

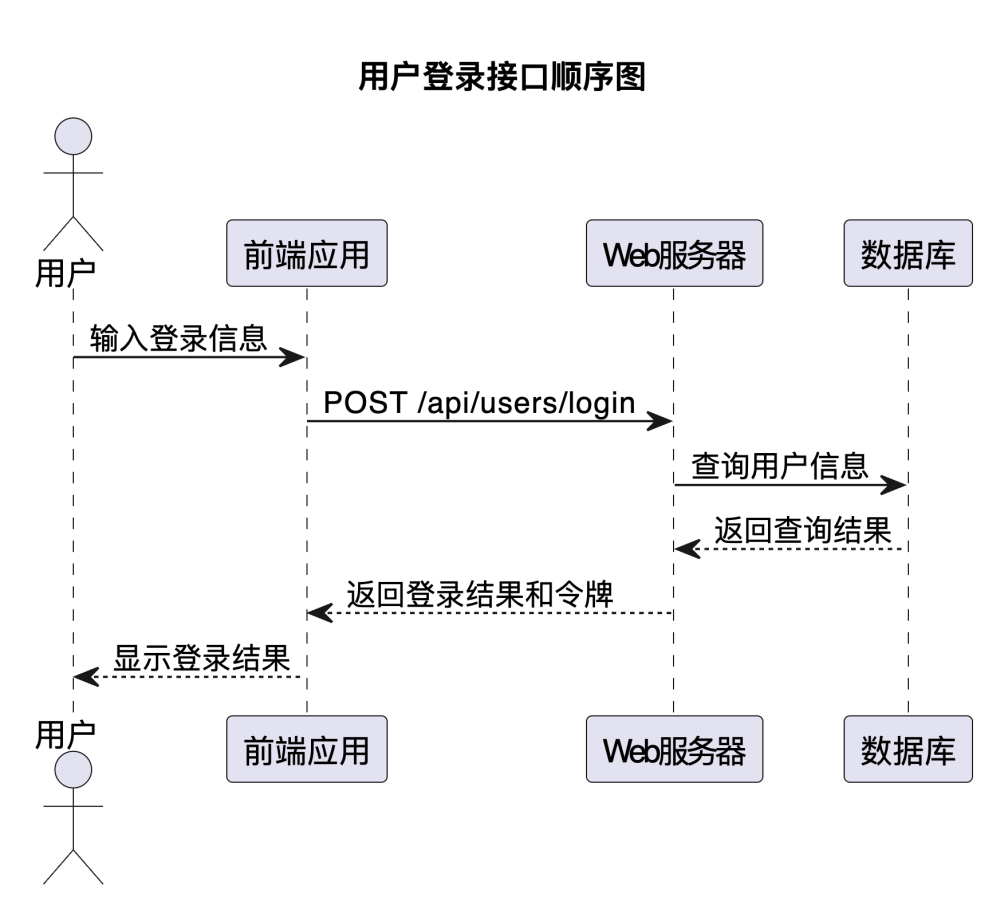
code: 状态码，整数类型

message: 消息，字符串类型

data: {

userId: 用户 ID，整数类型

token: 令牌，字符串类型 }



3. 用户信息更新

接口地址：/api/users/{userId}

请求方法：PUT

请求头：

Authorization: Bearer {token}

请求参数：

username: 用户名，字符串类型，可选

password: 密码，字符串类型，可选

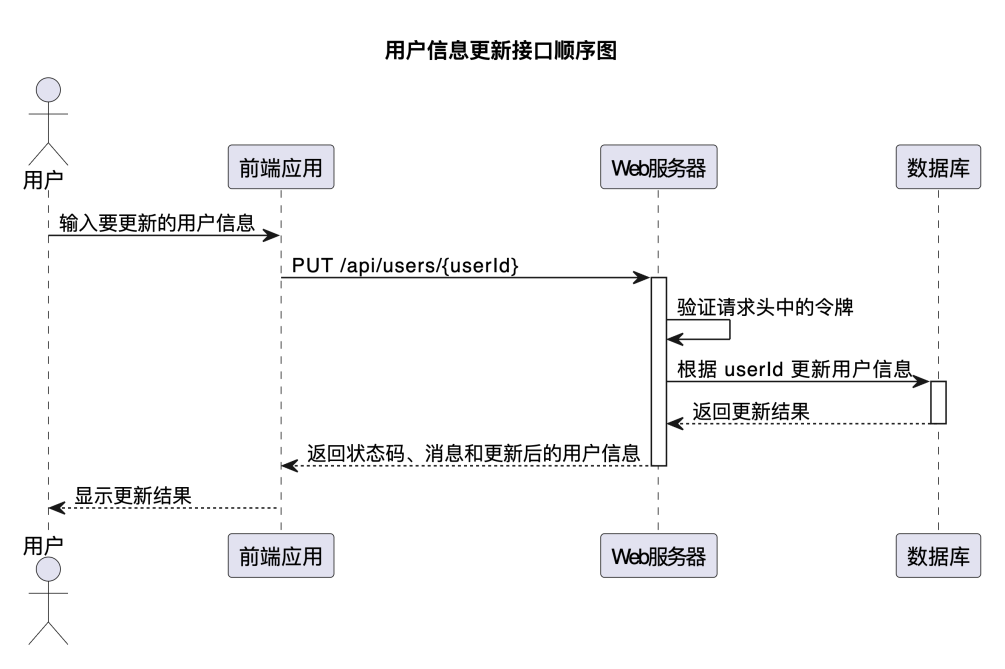
email: 邮箱，字符串类型，可选

响应数据：

code: 状态码，整数类型

message: 消息，字符串类型

data: 更新后的用户信息，对象类型



4. 发布社区交互内容

接口地址：/api/interactions

请求方法：POST

请求头：

Authorization: Bearer {token}

请求参数：

title: 标题，字符串类型，必填

content: 内容，文本类型，必填

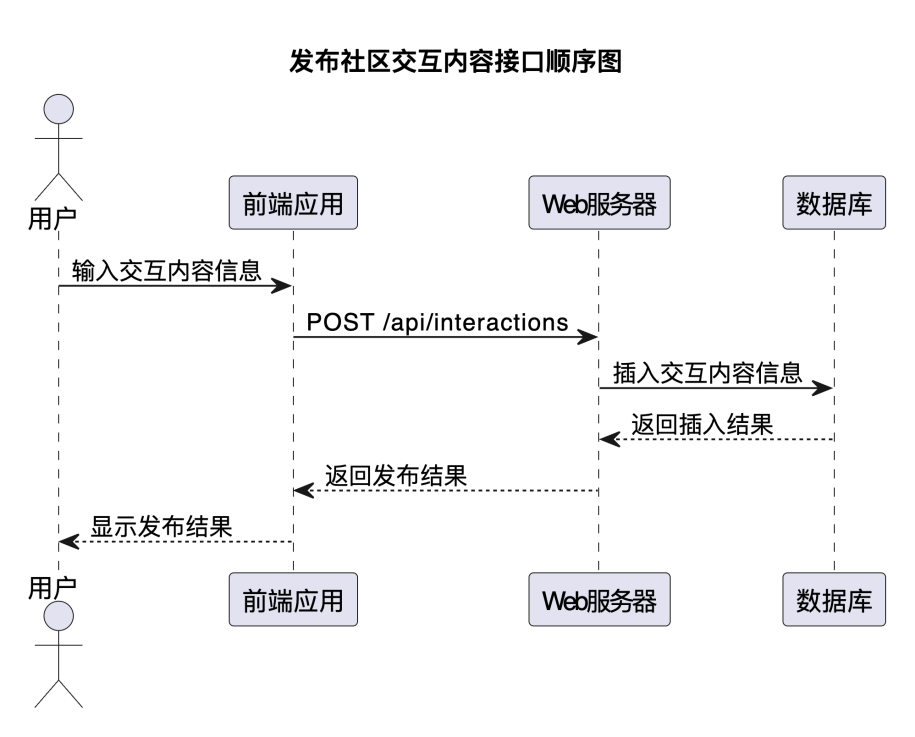
classificationId: 分类 ID，整数类型，必填

响应数据：

code: 状态码，整数类型

message: 消息，字符串类型

data: 发布成功的交互内容 ID，整数类型



5. 获取社区交互内容列表

接口地址：/api/interactions

请求方法：GET

请求参数：

classificationId: 分类 ID，整数类型，可选

keyword: 关键词，字符串类型，可选

page: 页码，整数类型，默认 1

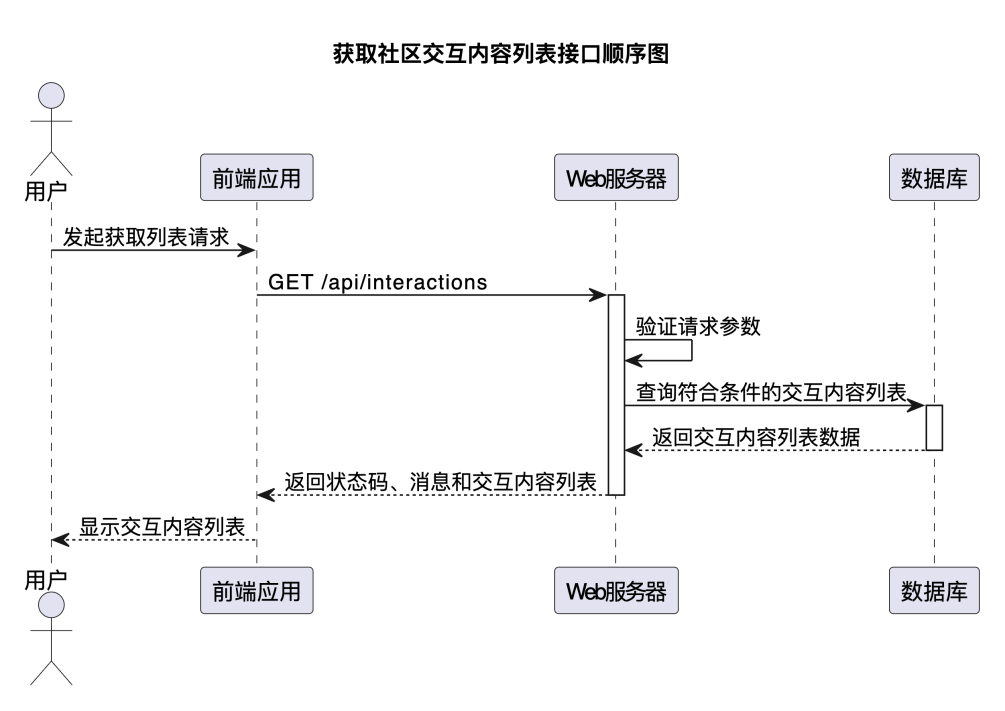
pageSize: 每页数量，整数类型，默认 10

响应数据：

code: 状态码，整数类型

message: 消息，字符串类型

data: [ { interactionId: 交互内容 ID，整数类型 title: 标题，字符串类型 content: 内容，文本类型 authorId: 作者 ID，整数类型 publishTime: 发布时间，日期时间类型 likeCount: 点赞数，整数类型 favoriteCount: 收藏数，整数类型 } ]



6. 获取社区交互内容详情

接口地址：/api/interactions/{interactionId}

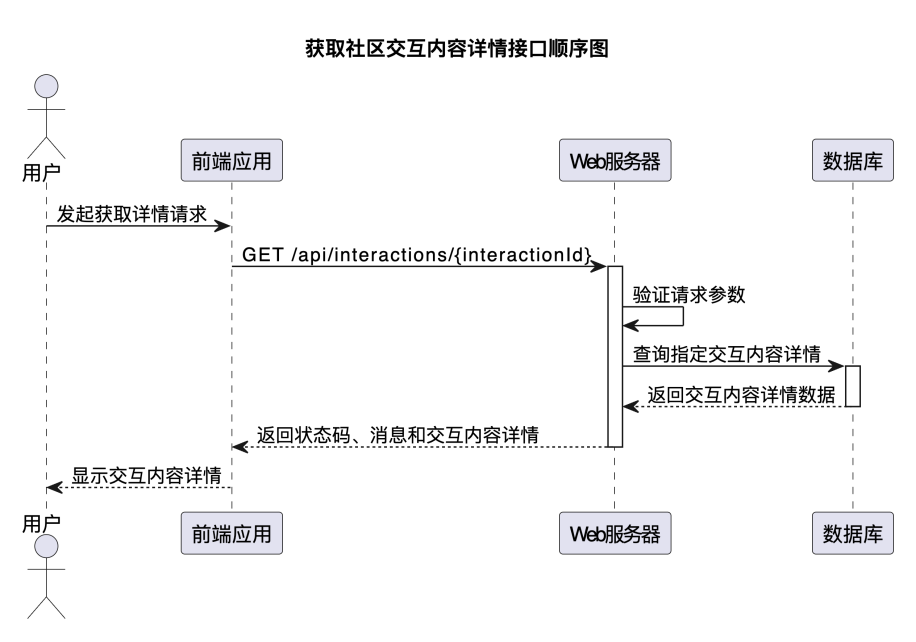
请求方法：GET

响应数据：

code: 状态码，整数类型

message: 消息，字符串类型

data: { interactionId: 交互内容 ID，整数类型 title: 标题，字符串类型 content: 内容，文本类型 authorId: 作者 ID，整数类型 publishTime: 发布时间，日期时间类型 likeCount: 点赞数，整数类型 favoriteCount: 收藏数，整数类型 comments: [ { commentId: 评论 ID，整数类型 commentContent: 评论内容，文本类型 commenterId: 评论者 ID，整数类型 commentTime: 评论时间，日期时间类型 } ] }



7、 发布评论

接口地址：/api/interactions/{interactionId}/comments

请求方法：POST

请求头：

Authorization: Bearer {token}

请求参数：

commentContent: 评论内容，文本类型，必填

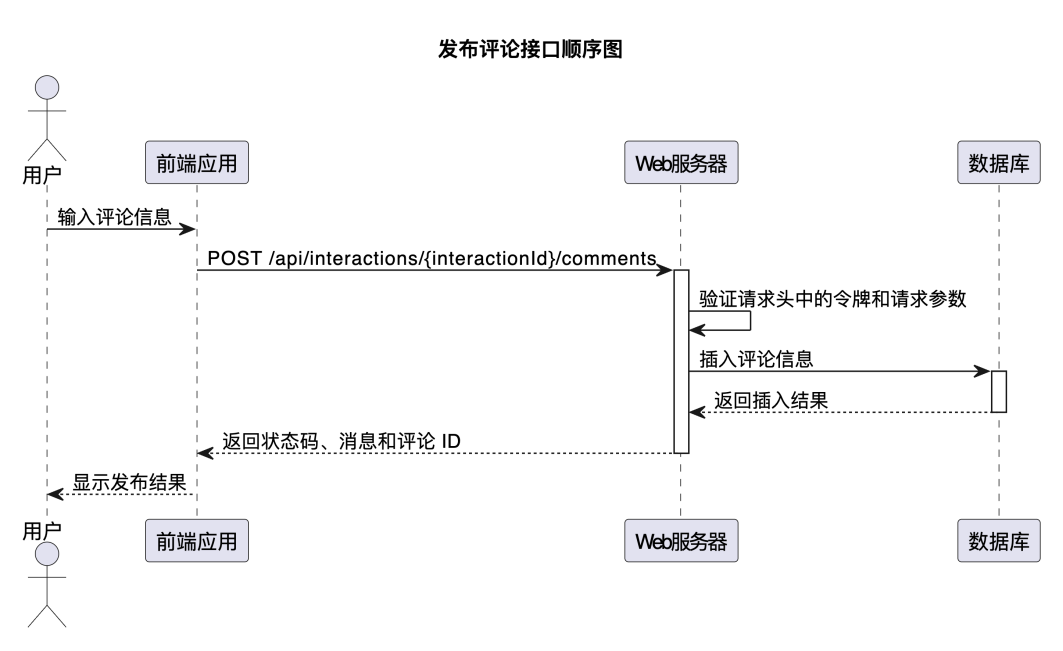
replyToCommentId: 回复的评论 ID，整数类型，可选

响应数据：

code: 状态码，整数类型

message: 消息，字符串类型

data: 发布成功的评论 ID，整数类型



1. 管理员模块

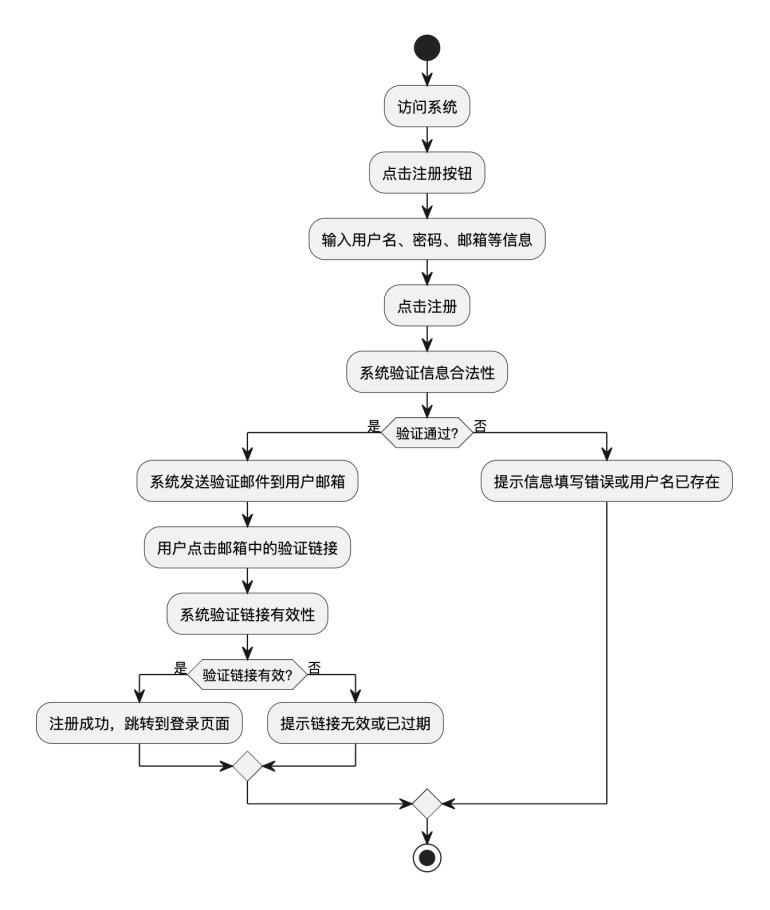
### 详细设计

系统详细设计部分借助 UML 建模工具中的活动图来进行描述。活动图作为一种重要的用于描述系统动态行为的工具，能够将系统内的各类活动以及活动间的转移关系清晰地呈现出来。

对于系统的详细设计，我们能够利用活动图来描绘用户、讲师以及管理员在系统中的行为流程。

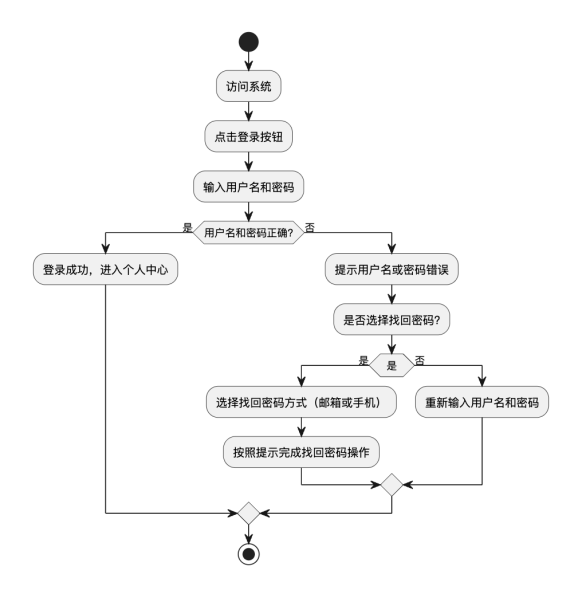
1. 用户注册

用户访问系统后点击注册按钮，输入必要的注册信息，如用户名、密码和邮箱等。系统会验证信息的合法性，若验证通过，系统会向用户邮箱发送验证邮件。用户点击邮件中的验证链接，系统再次验证链接的有效性。若链接有效，注册成功，用户跳转到登录页面；若链接无效或已过期，提示用户。若信息填写错误或用户名已存在，系统也会相应提示。



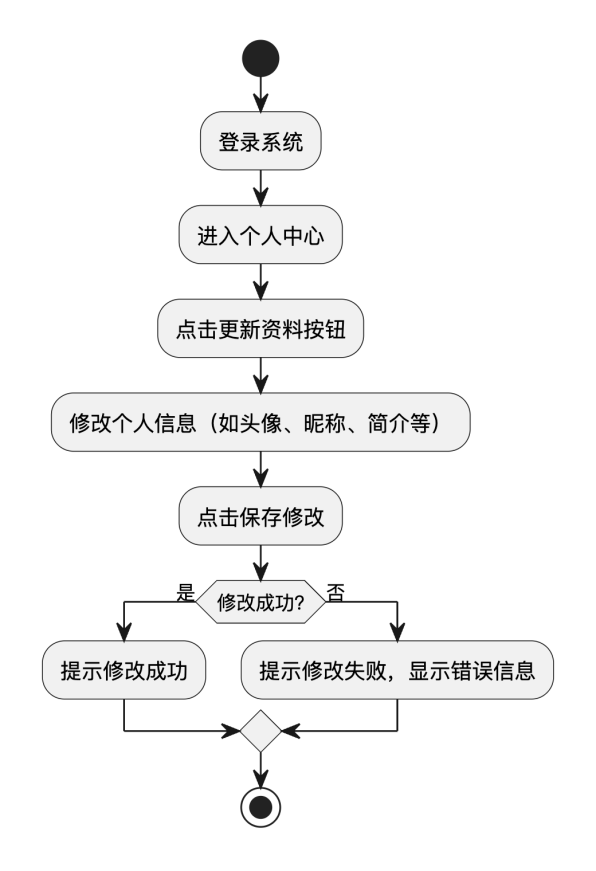
1. 用户登录

用户访问系统后点击登录按钮，输入用户名和密码。系统会验证用户名和密码是否正确。若正确，用户登录成功，进入个人中心；若错误，系统提示用户名或密码错误，并询问用户是否选择找回密码。若用户选择找回密码，可选择通过邮箱或手机等方式找回，按照提示完成操作；若不选择找回密码，则重新输入用户名和密码。



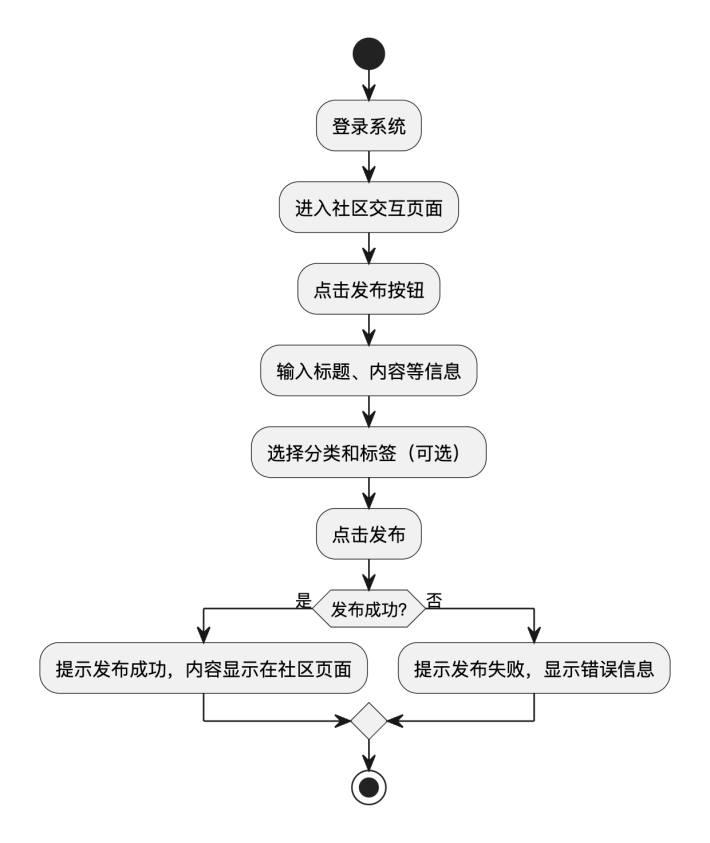
1. 用户更新个人资料

用户登录系统后进入个人中心，点击更新资料按钮，可修改个人信息，如头像、昵称、简介等。修改完成后点击保存修改，系统会验证修改是否成功。若成功，提示用户修改成功；若失败，提示修改失败并显示具体的错误信息，用户可根据提示进行相应调整。



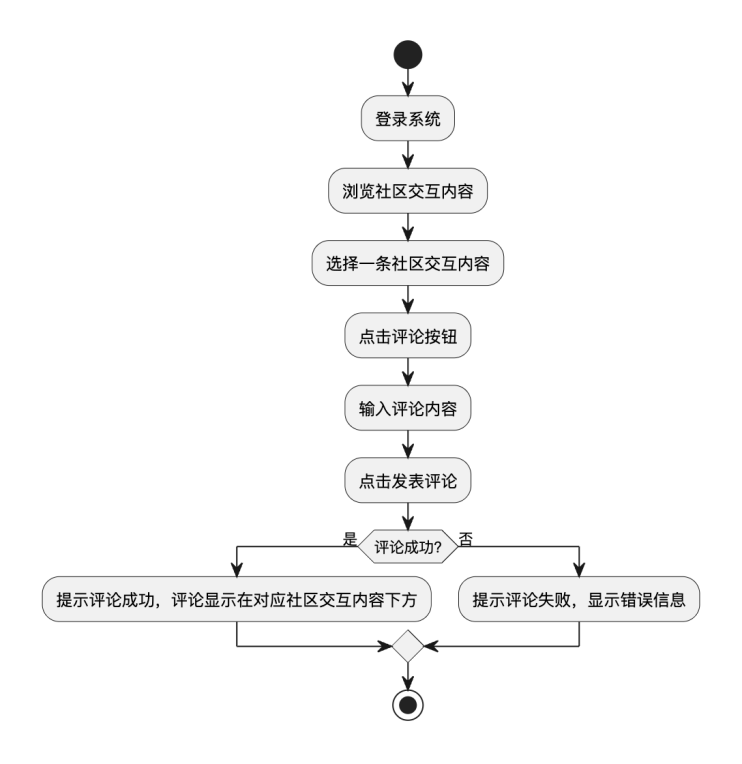
1. 用户发布社区交互内容

用户登录系统后进入社区交互页面，点击发布按钮，输入社区交互内容的标题、内容等必要信息，还可选择分类和标签（非必选）。点击发布后，系统会验证发布是否成功。若成功，提示用户发布成功，内容会显示在社区页面；若失败，提示发布失败并显示具体的错误信息，用户可根据提示进行调整。



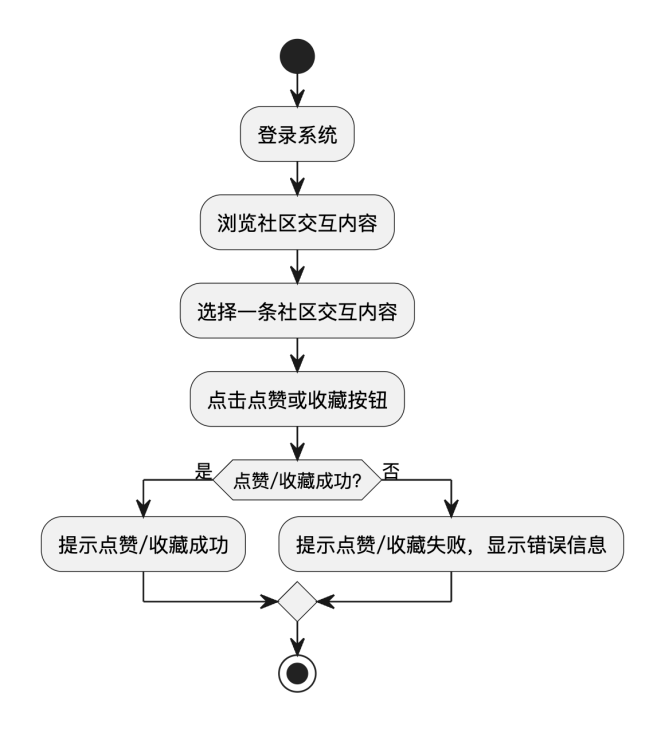
1. 用户发布评论

用户登录系统后浏览社区交互内容，选择一条感兴趣的内容，点击评论按钮，在弹出的输入框中输入评论内容，然后点击发表评论。系统会验证评论是否成功。若成功，提示用户评论成功，评论会显示在对应社区交互内容下方；若失败，提示评论失败并显示具体的错误信息，用户可根据提示进行调整。



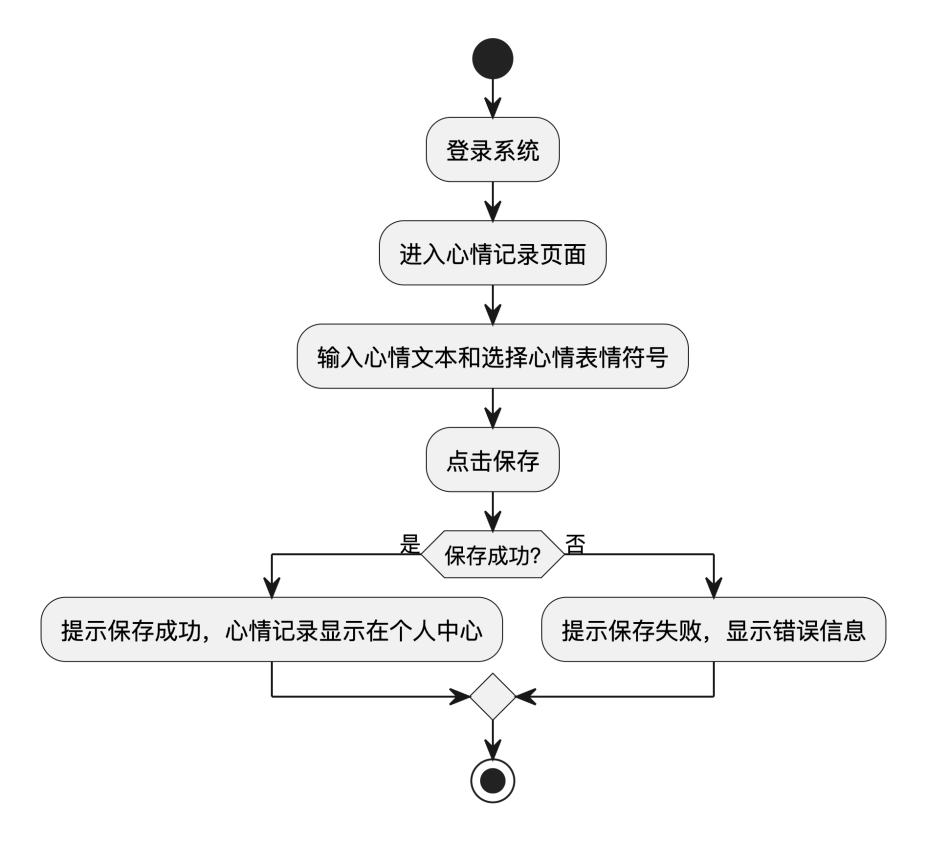
1. 用户点赞/收藏社区交互内容

用户登录系统后浏览社区交互内容，选择一条感兴趣的内容，点击点赞或收藏按钮。系统会验证点赞或收藏是否成功。若成功，提示用户点赞或收藏成功；若失败，提示点赞或收藏失败并显示具体的错误信息，用户可根据提示进行调整。



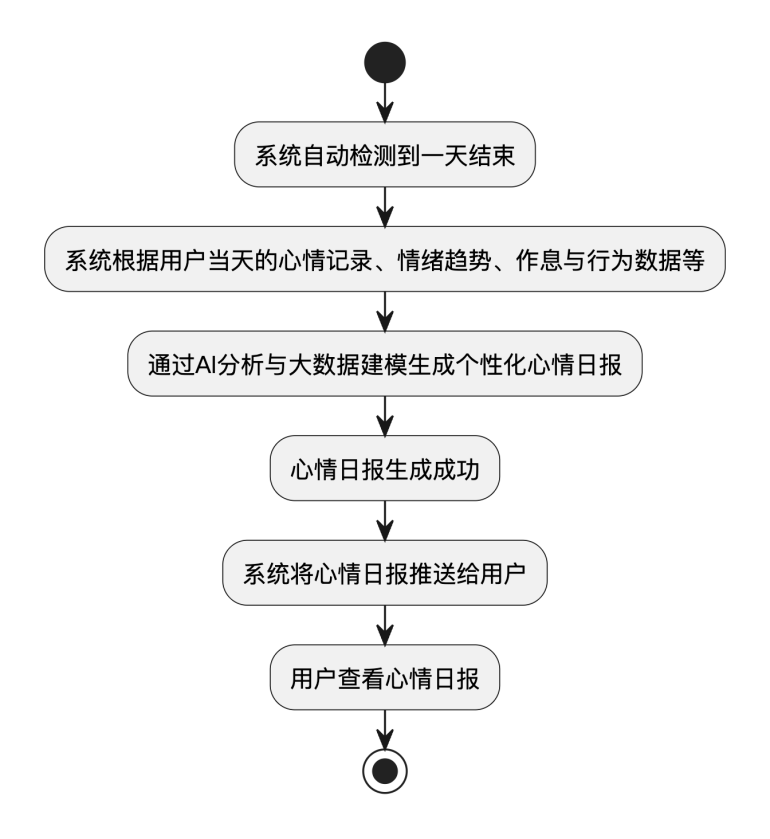
1. 记录心情

用户登录系统后进入心情记录页面，输入心情文本并选择相应的心情表情符号，点击保存。系统会验证保存是否成功。若成功，提示用户保存成功，心情记录会显示在个人中心；若失败，提示保存失败并显示具体的错误信息，用户可根据提示进行调整。



1. 生成心情日报

系统会在一天结束时自动根据用户当天的心情记录、情绪趋势、作息与行为数据等信息，通过AI分析与大数据建模生成个性化的心情日报。心情日报生成成功后，系统会将其推送给用户，用户可查看心情日报，了解自己当天的情绪波动和生活状态等。



1. 发送AI聊天

用户登录系统后进入AI聊天页面，输入聊天文本并点击发送。系统接收到消息后进行处理，根据AI模型生成回复消息，并将回复消息发送给用户。用户可在AI聊天页面查看系统的回复消息，实现与AI的互动交流。

