

软件过程管理

期末论文



题 目 **学期论文\_校园表情包图鉴平台系统**

学 院  **软件学院**

专 业  **软件工程**

学生姓名  **黄嘉桢**

年 级 **2022级**

指导教师  **毌攀良**

二Ο二五年06月26日

# 目录

[目录 2](#_Toc4966)

[1 CMMI层次成熟度模型 4](#_Toc5362)

[1.1 起源与发展 4](#_Toc17977)

[1.2 核心概念与原理 4](#_Toc27919)

[1.3 五个成熟度级别 4](#_Toc29500)

[2 校园表情包图鉴平台系统过往内容 7](#_Toc12439)

[2.1 产品透视 7](#_Toc2319)

[2.2 用户类和特征 8](#_Toc3451)

[2.3 需求概述 9](#_Toc22633)

[2.4 系统数据存储需求 10](#_Toc10746)

[2.5 功能性需求 11](#_Toc11626)

[2.6 功能用例 14](#_Toc19748)

[2.7 非功能需求 23](#_Toc30856)

[2.8 领域模型 26](#_Toc31897)

[2.9 系统活动图 27](#_Toc11227)

[2.10 系统时序图 28](#_Toc16675)

[2.11 系统架构 32](#_Toc22484)

[2.12 数据字典 33](#_Toc4367)

[2.13 数据库模式设计 34](#_Toc10919)

[2.14 数据流设计 35](#_Toc5754)

[2.15 需求追踪 37](#_Toc20451)

[3 校园表情包图鉴平台系统的软件过程成熟度深度评估 40](#_Toc30960)

[3.1 项目背景与评估方法 40](#_Toc23068)

[3.2 需求管理能力评估 40](#_Toc19724)

[3.3 项目计划与监控评估 40](#_Toc28305)

[3.4 软件工程实践评估 41](#_Toc8693)

[3.5 质量保证与配置管理评估 41](#_Toc27429)

[3.6 过程标准化程度评估 41](#_Toc9197)

[3.7 度量与分析能力评估 42](#_Toc16698)

[3.8 团队能力与组织支持评估 42](#_Toc17497)

[3.9 综合评估结论 42](#_Toc21533)

[3 校园表情包图鉴平台系统的软件过程改进计划 43](#_Toc6876)

[3.1 第一阶段：完善基础管理过程（1-3个月） 43](#_Toc24620)

[3.2 第二阶段：标准化和优化过程（4-6个月） 43](#_Toc4226)

[3.3 第三阶段：持续改进和创新（7-12个月） 43](#_Toc22497)

[3.4 改进计划的实施保障 44](#_Toc11534)

# 1 CMMI层次成熟度模型

## 起源与发展

能力成熟度模型集成（CMMI）的发展历程可以追溯到20世纪80年代。当时，美国国防部面临着软件项目频繁失败、成本超支、进度延期等严重问题。为了解决这些问题，国防部委托卡内基梅隆大学软件工程研究所（SEI）开发一套评估软件承包商能力的方法。1991年，SEI发布了软件能力成熟度模型（SW-CMM），这是CMMI的前身。

随着时间的推移，除了SW-CMM之外，还出现了系统工程能力成熟度模型（SE-CMM）、集成产品开发能力成熟度模型（IPD-CMM）等多个模型。这些模型虽然在各自领域发挥了重要作用，但也带来了实施的复杂性和不一致性。为了解决这个问题，SEI在2000年开始开发CMMI，将多个成熟度模型整合为一个统一的框架。

CMMI不仅继承了CMM的核心理念，还融入了现代软件工程和系统工程的最佳实践。它提供了一个综合的过程改进框架，帮助组织提高产品和服务的质量，提升项目管理能力，降低开发风险。CMMI已经成为全球软件和系统工程领域最广泛认可的过程改进模型之一。

## 核心概念与原理

CMMI的核心理念是通过持续的过程改进来提高组织的能力和成熟度。它基于这样一个假设：高质量的过程会产生高质量的产品。CMMI将过程视为组织成功的关键因素，通过定义、管理、测量和优化过程，组织可以更好地控制项目结果，提高产品质量，降低成本和风险。

CMMI采用了阶段式表示法和连续式表示法两种方式。阶段式表示法定义了五个成熟度级别，为组织提供了一个清晰的改进路线图。连续式表示法则允许组织选择特定的过程域进行改进，提供了更大的灵活性。本文主要讨论阶段式表示法中的五个成熟度级别。

过程域是CMMI的基本组成单元，每个过程域都包含了一组相关的实践，当这些实践被实施时，可以满足该过程域的目标。CMMI定义了22个过程域，分布在不同的成熟度级别中。每个过程域都有特定目标和通用目标，特定目标描述了该过程域必须实现的独特特征，通用目标则描述了制度化过程的特征。

## 五个成熟度级别

CMMI不仅继承了CMM的核心理念，还融入了现代软件工程和系统工程的最佳实践。它提供了一个综合的过程改进框架，帮助组织提高产品和服务的质量，提升项目管理能力，降低开发风险。CMMI已经成为全球软件和系统工程领域最广泛认可的过程改进模型之一。

* + 1. **初始级（Level 1）**

初始级是所有组织的起点，也是最混乱的状态。在这个级别，组织通常没有稳定的软件开发和维护环境。项目的成功主要依赖于个人的英雄主义行为，而不是组织的能力。当这些关键人员离开时，他们的经验和知识也随之流失，组织无法重复过去的成功。

在初始级，项目经常面临以下问题：需求不明确且频繁变更，没有正式的项目计划或计划形同虚设，开发过程混乱无序，质量控制依赖于最后阶段的测试，项目经常延期和超支。组织在危机驱动的模式下运作，总是在救火，而没有时间进行预防性的活动。

尽管初始级看起来很糟糕，但重要的是要认识到，即使在这个级别，组织仍然可能交付可用的产品。问题在于这种成功是不可预测和不可重复的。组织需要认识到过程改进的必要性，并开始采取行动向更高的成熟度级别迈进。

* + 1. **已管理级（Level 2）**

已管理级标志着组织开始对项目进行基本的管理。在这个级别，组织建立了基本的项目管理规程，项目按照文档化的计划进行实施和管理。虽然不同项目可能使用不同的过程，但都具备了基本的项目管理能力。

这个级别包含七个过程域：需求管理（REQM）确保项目团队和客户对需求有共同的理解；项目计划（PP）建立和维护项目活动的计划；项目监控（PMC）提供对项目进展的了解，以便在实际情况显著偏离计划时采取纠正措施；供应商协议管理（SAM）管理从供应商获取产品和服务；测量和分析（MA）开发和维持测量能力以支持管理信息需求；过程和产品质量保证（PPQA）为项目组和管理层提供对过程和相关工作产品的客观洞察；配置管理（CM）建立和维护工作产品的完整性。

在已管理级，组织能够基于以往类似项目的经验进行计划和管理。管理层可以在关键点上了解项目状态，如主要里程碑和任务完成情况。当项目出现问题时，能够及时发现并采取纠正措施。然而，这个级别的过程仍然是项目级的，不同项目之间可能存在较大差异。

* + 1. **已定义级（Level 3）**

已定义级是组织过程成熟度的一个重要里程碑。在这个级别，组织已经开发了标准的软件过程，所有项目都使用批准的、裁剪后的标准过程版本。过程的标准化带来了一致性，使得组织能够更有效地分享最佳实践和经验教训。

这个级别新增了十一个过程域：需求开发（RD）产生和分析客户、产品和产品组件的需求；技术解决方案（TS）设计、开发和实现满足需求的解决方案；产品集成（PI）将产品组件组装成产品并交付给客户；验证（VER）确保工作产品满足其指定的需求；确认（VAL）证明产品或产品组件在其预期环境中满足其预期用途；组织过程焦点（OPF）在理解现有过程强项和弱项的基础上计划和实施组织过程改进；组织过程定义（OPD）建立和维护可用的组织过程资产和工作环境标准；组织培训（OT）开发人员的技能和知识，使他们能够有效和高效地履行其职责；集成项目管理（IPM）建立和管理项目以及利益相关者的参与；风险管理（RSKM）识别潜在问题，以便策划风险处理活动；决策分析和解决（DAR）使用正式的评估过程来评估已识别的多个备选方案。

已定义级与已管理级的关键区别在于过程的范围。在已管理级，过程是项目特定的；而在已定义级，过程是组织范围的。组织建立了过程资产库，包括标准过程、生命周期模型、过程裁剪指南、过程相关的组织度量数据库等。项目可以根据自身特点对标准过程进行裁剪，但裁剪必须遵循组织的指南。

* + 1. **定量管理级（Level 4）**

定量管理级代表了组织在过程管理上的又一次飞跃。在这个级别，组织和项目建立了质量和过程性能的定量目标，并将其作为管理过程的准则。选定的子过程通过统计和其他定量技术进行控制。

这个级别包含两个过程域：组织过程性能（OPP）建立和维护对组织标准过程性能的定量理解，提供过程性能数据、基线和模型以定量地管理组织的项目；定量项目管理（QPM）定量地管理项目以达成项目确立的质量和过程性能目标。

定量管理的核心在于使用统计和其他定量技术来理解过程的变异。组织收集详细的过程性能度量数据，建立过程性能基线和模型。这些数据和模型帮助项目预测是否能够达成质量和过程性能目标。当发现性能偏离预期时，可以进行根本原因分析并采取纠正措施。

在这个级别，质量和过程性能被理解为统计术语，需要在整个过程的生命周期中进行管理。组织能够区分过程性能的共同原因变异和特殊原因变异，并采取适当的行动。预测基于对过程变异的统计理解，使得管理决策更加科学和可靠。

* + 1. **持续优化级（Level 5）**

持续优化级是CMMI的最高成熟度级别。在这个级别，组织通过对性能的定量理解，持续改进其过程。组织使用定量的方法来理解过程中固有的变异并找出过程改进的原因。

这个级别包含两个过程域：组织绩效管理（OPM）主动管理组织的绩效以满足其业务目标，包括分析聚合数据、识别潜在的改进领域、选择和部署改进措施；原因分析和解决（CAR）识别缺陷和其他问题的原因并采取行动防止它们再次发生。

在持续优化级，改进包括渐进式的改进和创新性的改进。组织建立了机制来识别、评估和部署改进措施。创新的想法得到鼓励和支持，成功的改进被推广到整个组织。组织的文化支持持续改进，每个人都被授权进行改进。

这个级别的组织能够快速响应市场变化和新的机会。他们使用从整个组织收集的数据来识别性能差距，并系统地部署改进措施来缩小这些差距。过程改进成为组织日常业务活动的一部分，而不是额外的负担。

# 2 校园表情包图鉴平台系统过往内容

## 2.1 产品透视

校园表情包平台旨在提供一个专注于管理和传播校园文化表情包的在线空间。该平台通过系统化的存储和管理功能，帮助学生轻松找到和分享与校园生活相关的表情包，增强社交体验。用户可以上传自己的创作、浏览他人作品，并通过分类和标签功能快速检索感兴趣的内容。此外，平台还会定期举办活动，激励学生参与创作，促进校园文化的传播与交流。我们希望通过这个平台，让每个表情包都能生动展现校园生活的乐趣。为此，我们对系统进行如下问题陈述：

表1：表情包图鉴平台问题陈述

|  |  |
| --- | --- |
| **校园表情包图鉴平台系统问题陈述** | |
| **元素** | **描述** |
| **问题背景** | 随着校园文化的发展，表情包的共享性和传播性不足以支持其在校园社交中日渐攀升的地位 |
| **目标** | 开发一个便于用户上传、浏览和管理校园表情包的在线平台，提高表情包的共享和传播效率 |

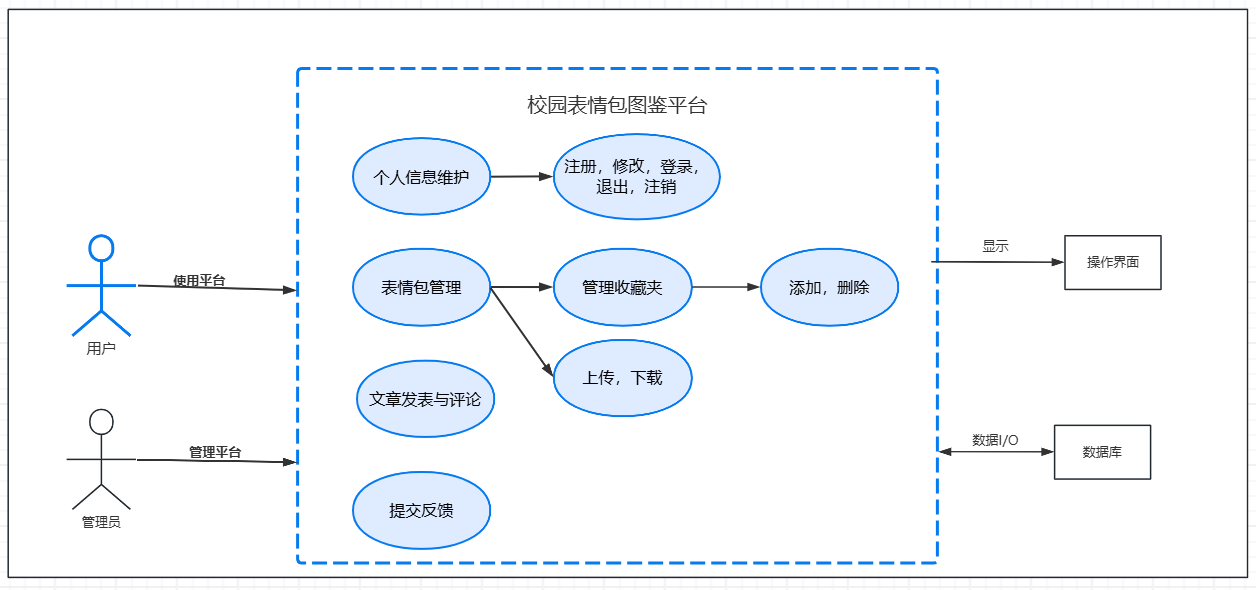
进一步地，我们对软件系统进行透视以明确系统大体样貌。具体地，我们定义了表情包管理系统的系统边界，如下图所示：

图1：表情包图鉴平台系统边界图

图中，用户对校园表情包图鉴平台进行使用，可进行个人信息维护（包括注册，修改，登录，退出，注销），表情包管理（包括上传下载表情包，管理收藏夹等），文章发表和评论，提交反馈等操作，操作结果显示在操作界面上，同时，平台也要与后端数据库进行对应的交互。同理，管理员可以对本平台进行管理，对用户的信息以及操作进行管理，操作结果也将显示在对应的页面，平台也应与数据库进行对应的交互。

除此之外，该软件与其他内部或外部的系统/子系统的关系如下表：

表2：表情包图鉴平台产品透视关系分析表

|  |  |  |  |
| --- | --- | --- | --- |
| **系统/子系统类别** | **名称** | **功能** | **关系** |
| 内部系统/子系统 | 用户管理系统 | 负责用户注册、登录、权限管理等。 | 表情包平台依赖用户管理系统确保用户信息的安全性和访问权限的控制。 |
| 内容管理系统 | 支持表情包的上传、审核、分类和标签管理。 | 表情包平台需要与内容管理系统集成，以便于高效地管理和展示表情包内容。 |
| 活动管理系统 | 组织和管理创作活动，激励用户参与。 | 表情包平台需与活动管理系统协作，以推动校园文化活动并收集用户反馈。 |
| 外部系统 | 社交网络平台 | 允许用户通过社交媒体分享表情包。 | 与主流社交平台（如微信、QQ）进行API对接，促进用户在外部平台分享校园文化。 |
| 搜索引擎 | 提升用户在互联网上搜索表情包的效率。 | 通过SEO优化，使表情包平台在搜索引擎中更容易被发现。 |
| 数据分析平台 | 分析用户行为、表情包使用频率等数据。 | 定期将平台数据传输至数据分析系统，以便进行深入分析和优化平台功能。 |
| 云存储服务 | 存储大量的表情包文件和用户数据。 | 平台通过API与云存储服务进行文件的上传和下载，确保数据的安全和可扩展性。 |

## 2.2 用户类和特征

表3：表情包图鉴平台用户类-特征表

|  |  |  |
| --- | --- | --- |
| **用户角色** | **特征描述** | **主要需求** |
| 终端用户 | 学生、教师、校友等使用平台的主要用户 | 便捷上传和分享表情包 |
| 易于搜索和浏览表情包 |
| 管理员 | 负责平台维护和管理的后台用户 | 用户管理 |
| 内容审核 |
| 平台监控 |
| 技术人员 | 负责系统开发与维护的技术支持角色 | 系统性能优化 |
| 解决技术问题 |
| 数据备份 |

## 2.3 需求概述

表情包图鉴平台的用户是普通学生。系统的任务是提供全面的信息管理功能，包括管理用户信息、管理表情包库信息、管理收藏库信息、管理文章信息、收集普通用户反馈等。同时，系统需要满足各类用户的不同需求，提供用户友好的界面和操作方式。用户能够通过本平台来浏览、下载、上传、收藏、搜索表情包，管理自己的收藏夹，并与其他用户以文章和评论的形式进行交流。

我们建立了用户列表、公开表情包库、个人收藏夹、文章列表、评论列表等列表。系统需要支持用户可以完成以下几方面功能需求：

* 在公开表情包库中进行所有表情包的查找和上传、下载。
* 在个人收藏夹中进行收藏表情包的增删改查。
* 在所有文章列表中进行查找。
* 在个人文章列表中进行个人文章的增删改查。
* 针对所有文章进行个人评论的增删。

进一步地，我们概括了以上需求的类别与优先级划分如下：

表4：表情包图鉴平台需求概述表

|  |  |  |  |
| --- | --- | --- | --- |
| **需求大类** | **需求小类** | **需求概述** | **优先级** |
| 用户信息管理 | 用户注册与管理 | 用户可以注册账户，系统将管理用户的基本资料和登录信息。 | 高 |
| 用户反馈收集 | 系统将提供一个反馈机制，允许用户提交意见和建议。 | 低 |
| 表情包库管理 | 表情包上传、下载、浏览与搜索 | 用户可以在公开表情包库中上传表情包，系统需要验证上传内容的合规性。同时，用户可以浏览和下载公开表情包库中的表情包，系统提供关键字搜索功能。 | 高 |
| 收藏库管理 | 表情包收藏 | 用户可以将喜欢的表情包添加到个人收藏夹。 | 中 |
| 收藏夹管理 | 用户可以对个人收藏夹中的表情包进行增删改查操作。 | 中 |
| 文章信息管理 | 文章浏览与搜索 | 用户可以在所有文章列表中查找感兴趣的文章。 | 中 |
| 个人文章管理 | 用户可以管理自己的文章，包括发布、编辑和删除。 | 中 |
| 文章评论 | 用户可以对文章发表评论，并管理自己的评论（增删）。 | 中 |

## 2.4 系统数据存储需求

**3.3.1 数据存储需求**

根据需求概述与数据逻辑关系，我们概括了该项目需要存储的信息以及对应类型的需求如下：

**1. 用户信息（User）**

存储需求：存储用户的基本信息，包括用户名、用户ID、密码、邮箱和手机号。

安全需求：用户密码需要加密存储，以保护用户隐私。

1. **表情包（Meme）**

存储需求：存储表情包的元数据，如表情包ID、名称、风格和图片链接。

性能需求：表情包图片可能较大，需要高效的存储和检索机制。

1. **收藏夹（Collection）**

存储需求：存储用户收藏的表情包集合信息，包括集合ID和名称。

关系需求：与用户和表情包建立多对多关系，通过中间表实现。

1. **文章（Article）**

存储需求：存储文章的详细信息，包括文章ID、标题、分类和内容。

性能需求：文章内容可能包含大量文本，需要优化存储和检索性能。

1. **评论（Comment）**

存储需求：存储用户对文章的评论，包括评论ID、内容和时间戳。

关系需求：与文章建立一对多关系。

1. **反馈（Feedback）**

存储需求：存储用户反馈信息，包括反馈ID、名称、内容和时间。

性能需求：反馈内容可能需要快速检索和分析。

**3.3.2 数据组织与持久化机制**

**1. 数据组织：**

关系型数据库：考虑到ER图中实体之间的关系，关系型数据库是合适的选择，如MySQL、PostgreSQL等。

数据模型：采用关系模型，通过表和外键来维护实体之间的关系。

1. **持久化机制：**

ACID属性：确保事务的原子性、一致性、隔离性和持久性。

数据备份与恢复：定期备份数据库，以防止数据丢失，并在需要时能够恢复。

**3.3.3 数据库设计概述**

除去基础的存储需求与组织设计之外，项目的数据还应该满足如下条件：

1. 对于经常查询的字段，如user\_name, meme\_name, article\_name等，建立索引以提高查询效率。
2. 对敏感信息如user\_pwd进行加密处理。
3. 实施访问控制，确保只有授权用户才能访问或修改数据。

基于上述需求，为校园表情包图鉴平台建立ER（实体-关系）图如下所示：

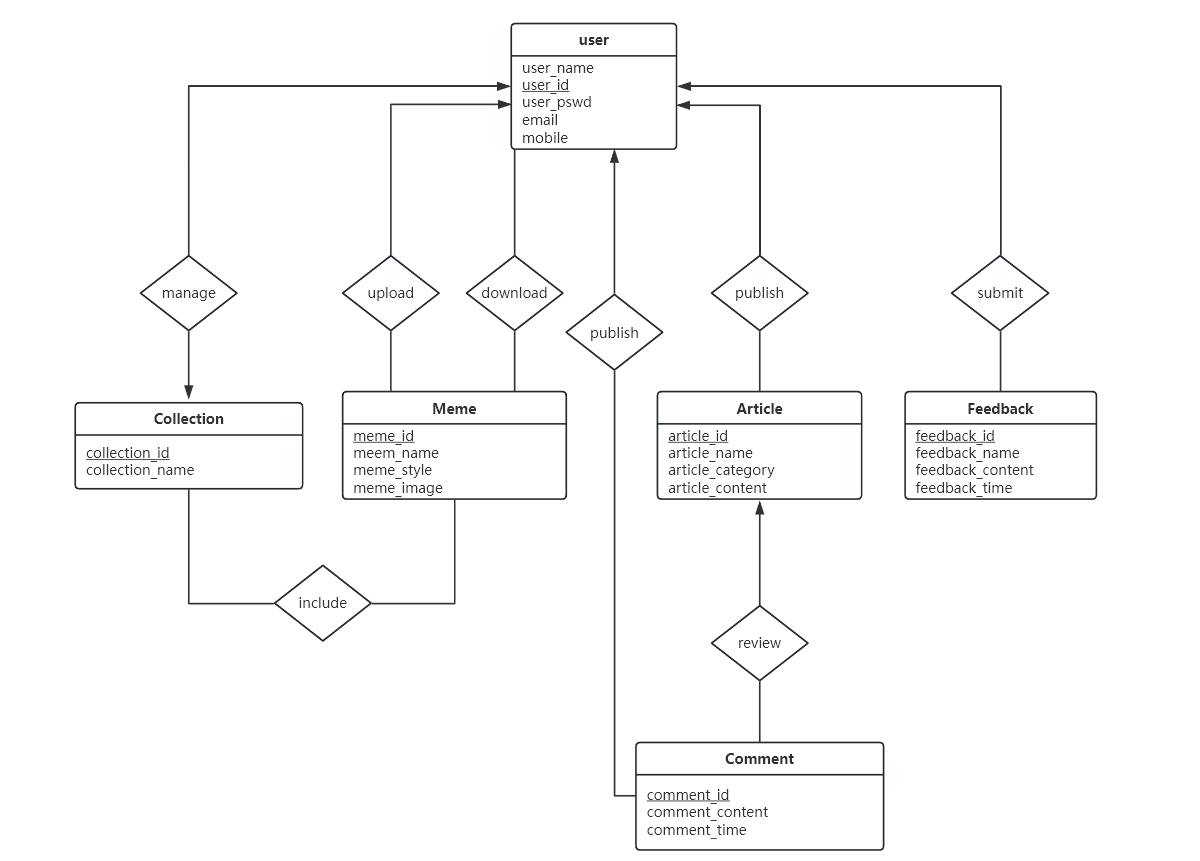


图2：表情包图鉴平台实体关系图（E-R图）

## 2.5 功能性需求

系统的功能性需求如下表：

表5：表情包图鉴平台系统功能性需求表

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **功能需求类别** | **需求概述** | **子功能** | **子功能描述** | **输入** | **输出** | **业务规则** | **数据流** | **特殊处理** |
| **个人信息维护** | 管理用户的基本信息和账户操作 | 注册与注销 | 新用户可创建账户，已有用户可注销账户 | 用户名、密码、邮箱、手机号 | 注册/注销确认消息 | 用户名唯一，密码加密存储，邮箱和手机号格式验证 | 用户输入信息 —— 验证 —— 数据库存储/删除 | 用户名已存在、密码强度不足、邮箱/手机号格式错误 |
| 登录 | 用户通过用户名和密码登录系统 | 用户名、密码 | 登录成功/失败消息 | 验证用户名和密码，记录登录时间 | 用户输入 —— 验证 —— 显示界面 | 用户名/密码错误、账户锁定 |
| 修改个人信息 | 用户可修改个人信息（如昵称、密码、联系方式） | 用户ID、新昵称、新密码、新联系方式 | 修改成功/失败消息 | 验证新密码强度，联系方式格式，更新用户信息 | 用户输入 —— 验证 —— 数据库更新 | 密码强度不足,更改密码、联系方式格式错误，修正联系方式 |
| **表情包管理** | 提供用户对表情包的上传、下载、删除和收藏功能 | 上传表情包 | 用户可上传校园相关表情包，并填写风格、主题、标题等信息 | 用户ID、表情包文件、风格、主题、标题 | 上传成功/失败消息 | 验证文件格式和大小，风格和主题分类 | 用户选择文件 —— 验证 —— 数据库存储 | 文件格式不支持、文件大小超限 |
| 下载表情包 | 用户可将公开库中的表情包下载到本地设备 | 用户ID、表情包ID | 下载链接或文件 | 检查用户权限和表情包状态 | 用户请求 —— 验证 —— 提供下载链接或文件 | 无权限访问、表情包不存在 |
| 删除表情包 | 用户可以删除自己已经下载和上传的表情包 | 用户ID、表情包ID | 删除成功/失败消息 | 检查用户权限和表情包所有权 | 用户请求 —— 验证 —— 数据库删除 | 无权限访问、表情包不存在 |
| 表情包收藏夹 | 用户可以将喜欢的表情包添加到个人收藏夹，并可管理收藏（添加、删除等操作） | 用户ID、表情包ID、操作类型（添加/删除） | 操作成功/失败消息 | 检查用户权限和表情包状态 | 用户请求 —— 验证 —— 数据库更新 | 无权限访问、表情包不存在 |
| **文章管理** | 用户可以发布与表情包相关的文章并互动交流 | 发表文章 | 用户可以撰写表情包相关文章并发布到平台上，与其他用户分享 | 用户ID、文章内容、分类 | 发布成功/失败消息 | 验证文章内容和分类，防止敏感词 | 用户输入 —— 验证 —— 数据库存储 | 内容违规、分类错误 |
| 删除文章 | 用户可以删除自己已经发表的文章 | 用户ID、文章ID | 删除成功/失败消息 | 检查用户权限和文章所有权 | 用户请求 —— 验证 —— 数据库删除 | 无权限访问、文章不存在 |
| 评论文章 | 用户可以在文章下进行评论，与文章作者和其他用户交流 | 用户ID、文章ID、评论内容 | 评论成功/失败消息 | 验证评论内容，防止敏感词 | 用户输入 —— 验证 —— 数据库存储 | 内容违规、文章不存在 |
| **系统反馈** | 用户对平台使用过程中遇到的问题进行反馈 | 提交反馈 | 用户可以通过提交反馈的方式，向平台反馈使用过程中的问题或建议 | 用户ID、反馈内容 | 提交成功/失败消息 | 验证反馈内容，记录反馈时间 | 用户输入 —— 验证 —— 数据库存储 | 内容违规、提交失败 |
| **管理员功能** | 管理员对系统进行后台管理，确保平台的正常运作和用户体验 | 管理用户 | 管理员可以创建或删除用户账号，审核修改用户的相关信息 | 管理员ID、用户操作（创建/删除/修改） | 操作成功/失败消息 | 检查管理员权限，验证用户信息 | 管理员请求 —— 验证 —— 数据库更新/删除 | 权限不足、用户信息错误 |
| 管理表情包 | 管理员可以对公开表情包库进行管理，审核、删除表情包 | 管理员ID、表情包ID、操作类型（审核/删除） | 操作成功/失败消息 | 检查管理员权限，验证表情包内容 | 管理员请求 —— 验证 —— 数据库更新/删除 | 权限不足、表情包不存在 |
| 文章管理 | 管理员对文章进行审核，可删除不合规的文章 | 管理员ID、文章ID、操作类型（审核/删除） | 操作成功/失败消息 | 检查管理员权限，验证文章内容 | 管理员请求 —— 验证 —— 数据库更新/删除 | 权限不足、文章不存在 |
| 查看、处理反馈 | 管理员可以查看用户提交的反馈并进行处理 | 管理员ID、反馈ID | 处理结果 | 检查管理员权限，记录处理时间 | 管理员请求 —— 验证 —— 数据库更新反馈状态 | 权限不足、反馈不存在 |

## 2.6 功能用例

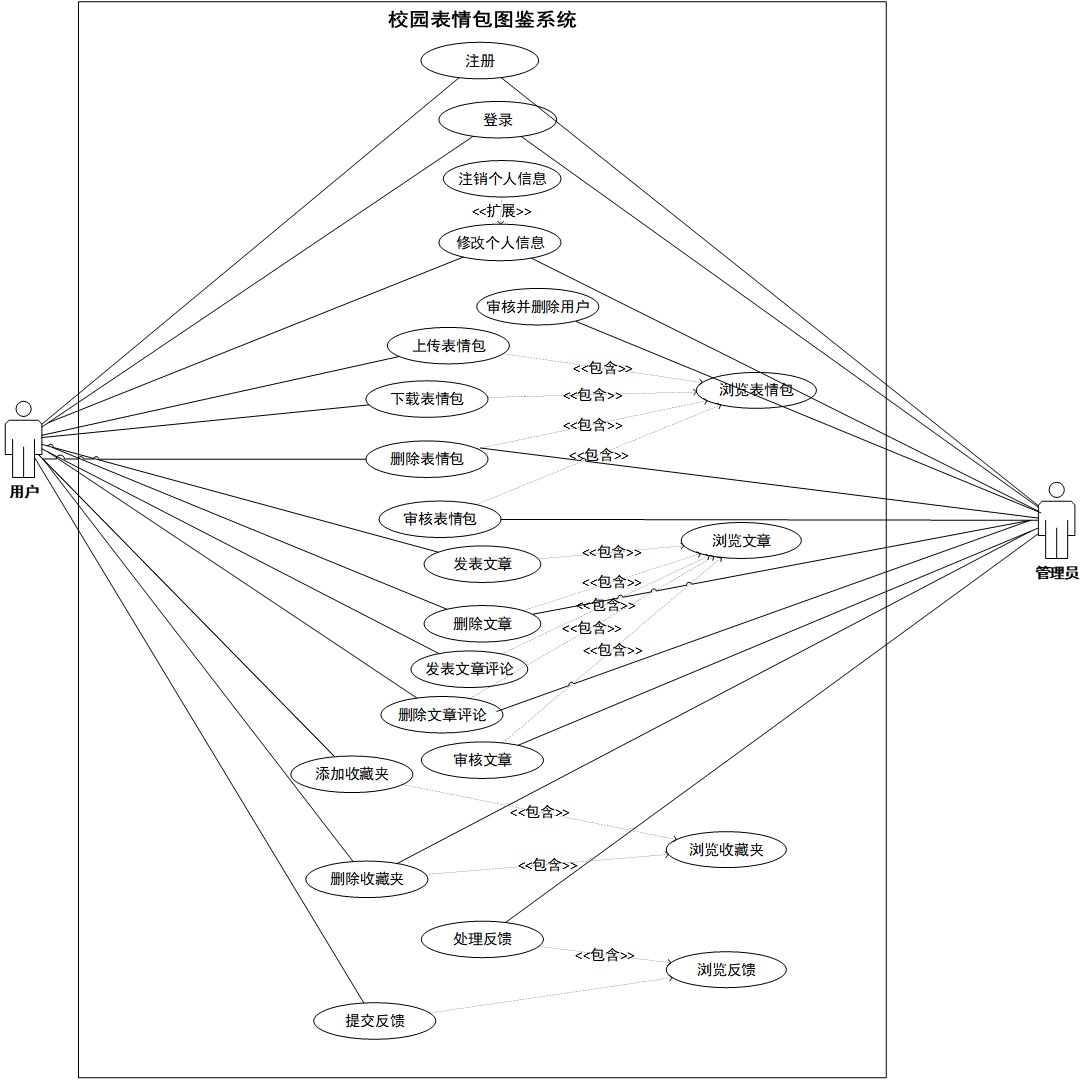
本系统的用例图如下：

图3：表情包图鉴平台功能用例图

对应的用例场景如下：

1. 注册

需求标识：UC01

需求名称：用户注册

场景描述：用户通过输入必要的信息在系统中创建账号。

执行者：用户

优先级：高

使用频度：低

前置条件：用户未在系统中注册。

正常过程：

1.用户点击“注册”。

2.系统提示输入注册信息（如用户名、密码、邮箱等）。

3.用户输入信息并提交。

4.系统验证信息并创建新用户。

后置条件：用户成功创建账户。

特殊需求：系统应确保用户名唯一性。

1. 登录

需求标识: UC02

需求名称: 用户登录

场景描述：用户通过输入必要的信息在系统中创建账号。

执行者: 用户

优先级: 高

使用频度: 高

前置条件: 用户已经成功注册并有账号。

正常过程:

* + 1. 用户点击“登录”。
    2. 系统要求输入用户名和密码。
    3. 用户输入信息并提交。
    4. 系统验证用户信息，允许登录。

后置条件: 用户进入系统主页。

异常过程: 如果用户名或密码错误，系统提示错误信息。

1. 注销个人信息

需求标识: UC03

需求名称: 注销账户

场景描述：用户可以注销个人账户信息。

执行者: 用户

优先级: 中

使用频度: 低

前置条件: 用户已成功登录。

正常过程:

* + 1. 用户点击“注销个人信息”。
    2. 系统确认注销操作。
    3. 用户确认注销，系统删除用户个人信息。

后置条件: 用户个人信息从系统中删除。

特殊需求: 系统应提供数据备份选项。

1. 修改个人信息

需求标识: UC04

需求名称: 修改个人信息

场景描述：用户可以修改个人账户信息。

执行者: 用户

优先级: 中

使用频度: 中

前置条件: 用户已成功登录。

正常过程:

* + 1. 用户点击“修改个人信息”。
    2. 系统展示个人信息编辑页面。
    3. 用户修改信息并提交。
    4. 系统更新用户个人信息。

后置条件: 用户个人信息更新成功。

异常过程: 如果信息更新失败，系统应提示错误并允许用户重试。

1. 上传表情包

需求标识: UC05

需求名称: 上传表情包

场景描述：用户可以上传新的表情包到系统中。

执行者: 用户

优先级: 中

使用频度: 中

前置条件: 用户已成功登录。

正常过程:

* + 1. 用户点击“上传表情包”。
    2. 系统提示选择表情包文件进行上传。
    3. 用户选择文件并提交。
    4. 系统接收并保存表情包，等待管理员审核。

后置条件: 表情包上传成功，进入审核流程。

特殊需求: 系统应限制上传文件的大小和格式。

1. 下载表情包

需求标识: UC06

需求名称: 下载表情包

场景描述：用户可以从系统下载已审核通过的表情包。

执行者: 用户

优先级: 高

使用频度: 高

前置条件: 用户已成功登录，并有表情包可供下载。

正常过程:

* + 1. 用户点击“下载表情包”。
    2. 系统展示可下载的表情包列表。
    3. 用户选择表情包并点击下载。

后置条件: 表情包下载成功。

异常过程: 如果下载失败，系统应提示错误信息。

1. 删除表情包

需求标识: UC07

需求名称: 删除表情包

场景描述：用户可以删除自己上传的表情包。

执行者: 用户

优先级: 低

使用频度: 低

前置条件: 用户已成功登录，且有已上传的表情包。

正常过程:

* + 1. 用户点击“删除表情包”。
    2. 系统确认删除操作。
    3. 用户确认，系统删除表情包。

后置条件: 表情包从系统中删除。

异常过程: 如果删除失败，系统应提示错误信息。

1. 审核表情包

需求标识: UC08

需求名称: 审核表情包

场景描述：管理员审核用户上传的表情包，决定是否通过审核。

执行者: 管理员

优先级: 高

使用频度: 中

前置条件: 有用户提交的表情包待审核。

正常过程:

* + 1. 管理员登录系统，进入“审核表情包”页面。
    2. 系统展示待审核表情包列表。
    3. 管理员选择一个表情包进行审核。
    4. 管理员选择“通过”或“拒绝”。

后置条件: 表情包审核结果更新，若通过则表情包公开，若拒绝则通知用户。

特殊需求: 系统应提供审核历史记录。

1. 发表文章

需求标识: UC09

需求名称: 发表文章

场景描述：用户可以在系统中发表文章（可能与表情包相关的讨论）。

执行者: 用户

优先级: 中

使用频度: 中

前置条件: 用户已登录系统。

正常过程:

* + 1. 用户点击“发表文章”。
    2. 系统提示输入文章标题和内容。
    3. 用户填写相关信息并提交。
    4. 系统保存文章并显示在文章列表中。

后置条件: 文章成功发表，所有用户可以浏览。

特殊需求: 系统应支持文章的格式化编辑。

1. 删除文章

需求标识: UC10

需求名称: 删除文章

场景描述：用户或管理员可以删除已发表的文章。

执行者: 用户/管理员

优先级: 中

使用频度: 低

前置条件: 用户已登录并拥有文章。

正常过程:

* + 1. 用户或管理员点击“删除文章”。
    2. 系统确认删除操作。
    3. 用户确认删除，文章被系统移除。

后置条件: 文章成功删除。

异常过程: 如果删除失败，系统应提示错误信息。

1. 发表文章评论

需求标识: UC11

需求名称: 发表文章评论

场景描述：用户可以对文章发表评论。

执行者: 用户

优先级: 低

使用频度: 低

前置条件: 用户已登录系统并阅读文章。

正常过程:

* + 1. 用户点击“发表文章评论”。
    2. 系统提示输入评论内容。
    3. 用户输入评论并提交。
    4. 系统保存评论并展示在文章下方。

后置条件: 评论成功发表。

特殊需求: 系统应支持评论的审核机制。

1. 删除文章评论

需求标识: UC12

需求名称: 删除文章评论

场景描述：用户或管理员可以删除已发表的文章评论。

执行者: 用户/管理员

优先级: 低

使用频度: 低

前置条件: 用户已登录并拥有评论。

正常过程:

* + 1. 用户或管理员点击“删除文章评论”。
    2. 系统确认删除操作。
    3. 用户确认，系统删除评论。

后置条件: 评论从系统中删除。

异常过程: 如果删除失败，系统应提示错误信息。

1. 审核文章

需求标识: UC13

需求名称: 审核文章

场景描述：管理员可以审核用户发表的文章。

执行者: 管理员

优先级: 高

使用频度: 中

前置条件: 用户发表的文章等待审核。

正常过程:

* + 1. 管理员进入“审核文章”页面。
    2. 系统展示待审核文章列表。
    3. 管理员选择文章并审核。
    4. 管理员决定是否通过文章并提交结果。

后置条件: 文章的审核结果更新。

特殊需求: 系统应提供审核历史记录。

1. 浏览文章

需求标识: UC14

需求名称: 浏览文章

场景描述：用户和管理员都可以浏览系统中的文章。

执行者: 用户/管理员

优先级: 高

使用频度: 高

前置条件: 系统中有已发表的文章。

正常过程: 1. 用户或管理员点击“浏览文章”。

2. 系统展示文章列表。

3. 用户或管理员点击文章标题查看文章详情。

后置条件: 用户或管理员成功查看文章内容。

特殊需求: 系统应提供文章搜索功能。

1. 添加收藏夹

需求标识: UC15

需求名称: 添加收藏夹

场景描述：用户可以将表情包或文章添加到收藏夹。

执行者: 用户

优先级: 中

使用频度: 中

前置条件: 用户已登录，并浏览表情包或文章。

正常过程:

* + 1. 用户点击“添加收藏夹”。
    2. 系统提示选择收藏的内容。
    3. 用户确认添加。

后置条件: 内容成功添加到收藏夹中。

特殊需求: 系统应允许用户对收藏内容进行分类。

1. 删除收藏夹

需求标识: UC16

需求名称: 删除收藏夹

场景描述：用户可以从收藏夹中删除表情包或文章。

执行者: 用户

优先级: 低

使用频度: 低

前置条件: 用户已登录，并有收藏的内容。

正常过程:

* + 1. 用户点击“删除收藏夹”。
    2. 系统确认删除操作。
    3. 用户确认删除，系统从收藏夹中移除该内容。

后置条件: 内容成功从收藏夹中删除。

异常过程: 如果删除失败，系统应提示错误信息。

1. 提交反馈

需求标识: UC17

需求名称: 提交反馈

场景描述：用户可以向系统提交使用反馈。

执行者: 用户

优先级: 中

使用频度: 低

前置条件: 用户已登录系统。

正常过程:

* + 1. 用户点击“提交反馈”。
    2. 系统提示输入反馈内容。
    3. 用户填写反馈内容并提交。
    4. 系统保存反馈并通知管理员。

后置条件: 反馈信息提交成功。

特殊需求: 系统应允许用户跟踪反馈的处理状态。

1. 处理反馈

需求标识: UC18

需求名称: 处理反馈

场景描述：管理员可以处理用户提交的反馈，并进行回应。

执行者: 管理员

优先级: 高

使用频度: 中

前置条件: 系统中有用户提交的反馈。

正常过程:

* + 1. 管理员点击“处理反馈”。
    2. 系统显示反馈列表。
    3. 管理员选择反馈并进行处理。
    4. 管理员提交处理结果，系统通知用户。
  + 后置条件: 反馈处理完成，用户收到处理结果。
  + 特殊需求: 系统应提供反馈处理的优先级设置。

1. 浏览反馈

需求标识: UC19

需求名称: 浏览反馈

场景描述：管理员可以浏览用户提交的反馈。

执行者: 管理员

优先级: 中

使用频度: 中

前置条件: 系统中有用户提交的反馈。

正常过程:

* + 1. 管理员点击“浏览反馈”。
    2. 系统展示所有反馈信息。

后置条件: 管理员成功查看反馈内容。

特殊需求: 系统应提供反馈的筛选和排序功能。

## 2.7 非功能需求

系统的非功能性需求列表，及其属性与分析如下：

表6：表情包图鉴平台系统非功能性需求表（总括）

1. **性能需求**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Priority** | **Effort** | **Risk** | **特性解释和分析** |
| 数据库访问性能优化 | Critical | High | High | 对于校园表情包库这种高频查询、上传、下载的系统，数据库性能优化是关键需求。若数据库查询和存储的效率不高，将直接影响用户体验和系统稳定性。通过索引优化、缓存等技术，系统可以加快查询速度，提升用户体验。此特性主要作用于表情包管理和用户信息模块，保障系统在高并发时依然流畅。优化难度较高，但优先级非常高，风险也高。 |
| 表情包文件的上传、下载和存储稳定性 | Important | Medium | Medium | 系统在表情包文件的处理上需具备较高的稳定性，避免文件丢失或损坏。这个特性对校园用户非常重要，因为表情包的展示和分享是平台的核心功能，任何上传或下载问题都可能影响用户体验。开发难度中等，风险为中等。 |
| 前端界面的响应速度优化 | Important | Medium | Medium | 由于表情包展示页面可能包含大量图片，提高页面加载速度和交互的流畅性可以极大提升用户体验。这个特性主要作用于表情包浏览、评论区等内容密集页面的加载，确保用户在快速浏览时没有卡顿感。实现难度适中，需要优化代码和资源加载速度，风险为中等。 |

1. **质量需求**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Priority** | **Effort** | **Risk** | **特性解释和分析** |
| 支持多用户权限管理 | Critical | Medium | Medium | 系统需要提供普通用户和管理员的不同权限，普通用户可以上传和收藏表情包，管理员则可以审核和管理内容，确保平台内容的合法合规。这个特性在权限控制、内容审核和用户管理等模块中发挥作用，确保系统内的资源管理和内容质量。开发难度适中，涉及数据库设计和权限逻辑配置，Priority和Risk均为中等。 |
| 可测试性 | Important | Medium | Low | 系统应易于测试，包括单元测试、集成测试和性能测试。这有助于在开发过程中及早发现问题，减少后期的维护成本。 |

1. **可用性需求**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Priority** | **Effort** | **Risk** | **特性解释和分析** |
| 兼容不同操作系统（Windows和Linux等） | Important | High | Medium | 校园环境可能使用不同的操作系统，因此系统需要支持多种操作系统，以便在多种环境中部署和使用。尤其是后端系统的开发和数据库的部署，确保它们能够在不同操作系统下稳定运行。实现难度较高，特别是在跨平台的后台和数据库部署方面，可能会遇到兼容性问题。 |
| 支持跨平台前端兼容性（浏览器兼容性） | Useful | Medium | Low | 校园用户可能会使用不同的浏览器访问系统，因此确保前端页面在不同浏览器中正常显示至关重要。该特性可以提升用户在各种设备和浏览器上的访问便利性，尤其适用于表情包展示和文章阅读模块。实现难度中等，优先级和风险较低。 |

1. **安全性需求**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Priority** | **Effort** | **Risk** | **特性解释和分析** |
| 并发访问安全性 | Critical | High | High | 确保系统在多用户同时访问下保持数据安全和不被篡改。对于校园表情包平台而言，学生们可能会在同一时间上传或下载表情包，甚至进行评论和互动。并发访问安全性可以防止数据在高并发情况下出现冲突或数据覆盖。这个特性主要体现在表情包的上传、下载和评论模块中，确保用户在同时操作时不会互相干扰。开发复杂度高，风险也高，属于Critical级别。 |
| 系统的用户信息和隐私保护（包括密码加密） | Critical | Medium | High | 由于系统涉及校园学生的个人信息，确保信息加密和隐私保护是必须的，防止信息泄露和滥用。具体来说，这一特性体现在用户注册、登录和个人资料管理模块中，为用户提供安全保障。难度和风险均较高，Priority为Critical。 |
| 访问控制 | Critical | Medium | Medium | 系统应实现严格的访问控制，确保用户只能访问其权限范围内的数据和功能。这需要角色基础的访问控制策略和细粒度的权限设置。 |
| 安全审计 | Important | Medium | Medium | 系统应记录安全相关的操作日志，以便在发生安全事件时进行审计和追踪。这包括日志收集、存储和分析的机制。 |

1. **可靠性需求**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Priority** | **Effort** | **Risk** | **特性解释和分析** |
| 数据库访问性能优化 | Critical | High | High | 对于校园表情包库这种高频查询、上传、下载的系统，数据库性能优化是关键需求。若数据库查询和存储的效率不高，将直接影响用户体验和系统稳定性。通过索引优化、缓存等技术，系统可以加快查询速度，提升用户体验。此特性主要作用于表情包管理和用户信息模块，保障系统在高并发时依然流畅。优化难度较高，但优先级非常高，风险也高。 |
| 表情包文件的上传、下载和存储稳定性 | Important | Medium | Medium | 系统在表情包文件的处理上需具备较高的稳定性，避免文件丢失或损坏。这个特性对校园用户非常重要，因为表情包的展示和分享是平台的核心功能，任何上传或下载问题都可能影响用户体验。开发难度中等，风险为中等。 |

1. **可维护性需求**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Priority** | **Effort** | **Risk** | **特性解释和分析** |
| 操作记录和日志系统（便于管理员追踪和管理） | Important | Medium | Medium | 记录用户的操作日志，便于管理员排查系统问题和管理内容。对于校园表情包平台，管理员可以通过操作记录追踪不当行为，及时处理违规内容，提升系统安全性。实现难度适中，风险中等。 |
| 错误处理和友好提示系统 | Important | Medium | Low | 在操作出现错误时，通过友好提示帮助用户理解问题并提供解决方案，可以提升用户体验。该特性在校园表情包平台中非常实用，如上传失败、登录错误等场景下，系统可以向用户清晰地解释原因。实现难度和风险均较低，但对用户体验影响很大。 |

1. **可扩展性需求**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Priority** | **Effort** | **Risk** | **特性解释和分析** |
| 可扩展性 | Important | High | Medium | 系统应具备良好的扩展性，以便日后支持更多功能（如新类型的表情包或视频分享）。这一特性可以在系统的架构设计中体现，例如模块化设计便于后续功能添加，并提高了应对用户增长的能力。Effort高，风险中等。 |
| 系统的扩展性与插件支持 | Useful | High | Medium | 系统应具备支持插件的能力，以便日后引入社交分享、数据分析等第三方插件。对于校园表情包平台，这个特性可以让用户便捷地分享表情包至其他社交平台，扩展功能性。开发难度较高，风险为中等。 |

1. **其它需求**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Priority** | **Effort** | **Risk** | **特性解释和分析** |
| 数据备份与恢复机制 | Useful | Low | Low | 为了防止数据意外丢失（如因系统故障或误操作），需要定期进行数据备份，并提供恢复机制。这项特性在校园表情包系统中可帮助管理员在数据丢失或损坏时快速恢复重要内容，特别是用户上传的表情包和收藏数据。该功能的开发难度低，但在安全管理上有实际价值。 |
| 国际化 | Important | Medium | Medium | 系统应支持多语言，以便不同语言的用户都能使用。这需要考虑本地化策略、语言包管理和文化适应性。 |
| 支持负载均衡 | Important | High | High | 校园用户人数多，若系统在同一时间点被大量访问，将可能导致服务器超载。因此，支持负载均衡可以保障系统在高并发场景下的稳定运行。负载均衡特性主要作用于后台服务器配置，尤其在热门表情包或文章发布期间尤为重要。难度和风险均较高。 |
| 提供API接口以支持第三方平台集成 | Useful | Medium | Medium | 为未来的扩展性和开放性，系统提供API接口，便于第三方平台集成。对于校园表情包系统，API接口可以允许与校内其他平台进行数据共享，如将表情包功能集成到校园论坛中，拓展表情包的使用范围。开发难度适中，风险中等。 |

## **2.8** 领域模型

根据需求进行分析，本平台的领域模型( Domain Model )如下所示：



图4：表情包图鉴平台领域模型图

该模型适用于校园表情包图鉴平台，涵盖了表情包的创建、管理和用户交互，包含以下实体：

* 用户：平台的用户，可以是学生、教师或工作人员。
* 表情包：定义了表情包共有的属性和行为。
* 收藏：用户的表情收藏夹。
* 评论：用户对表情包的评论。
* 反馈：用户发表的反馈。
* 文章：用户撰写发布的文章。
* 管理员：平台的管理员，管理平台信息和内容。

实体的行为和职责如下：

* 用户：可以上传表情包、发表评论和文章、收藏表情包、进行反馈等。
* 收藏：可以被创建、更新、删除、分享等。
* 表情：可以被上传、查看、评论等。
* 评论：可以被发表、审核、删除等。
* 文章：可以被发表、审核、删除等。

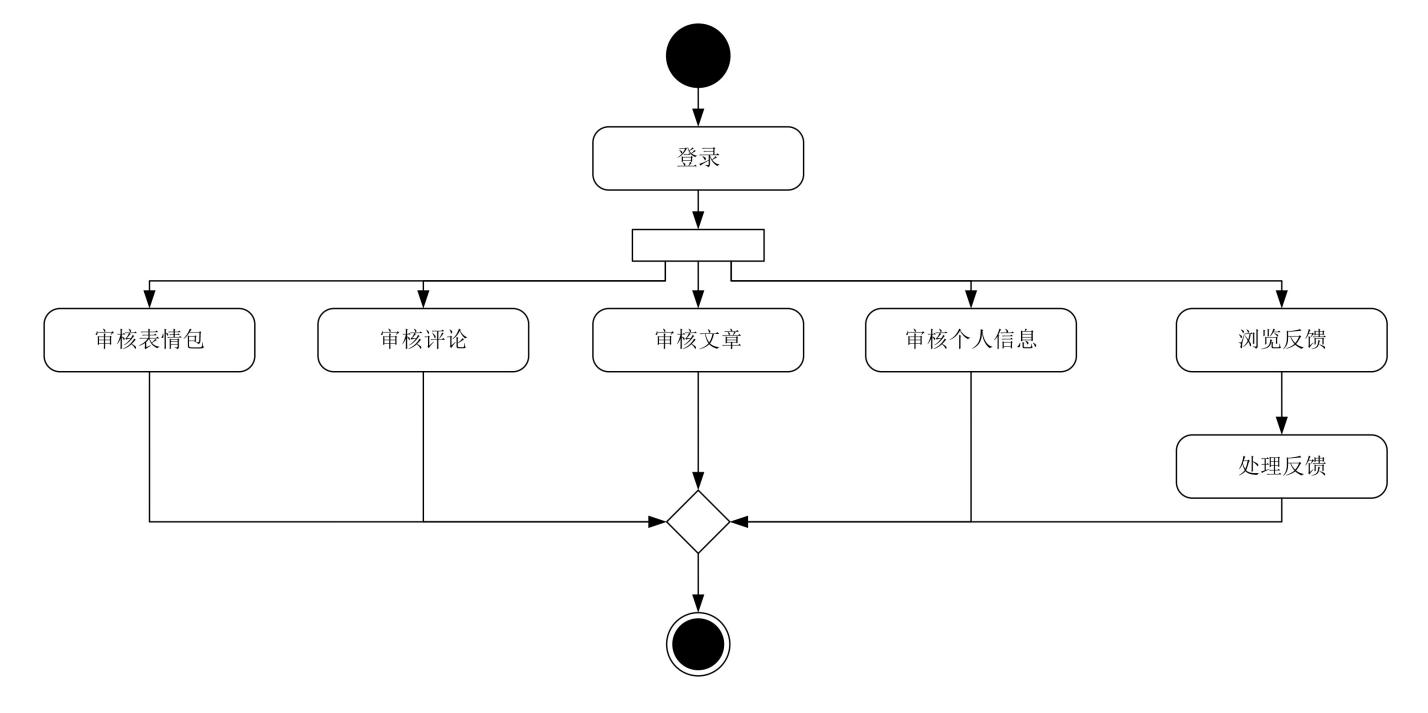
实体之间的关系如下：

* 用户与表情包：一个用户可以上传多个表情包，一个表情包只能由一个用户创建。
* 用户与评论：一个用户可以对多个表情包或表情发表评论，一个评论只能由一个用户发表。
* 用户与收藏：一个用户可以有多个收藏，一个收藏夹只属于一个用户。
* 用户与文章：一个用户可以发布多个文章，一个文章只属于一个用户。
* 用户与反馈：一个用户可以发布多个反馈，一个反馈只属于一个用户。
* 收藏与表情：一个收藏包含多个表情，一个表情只能属于一个收藏。

## 2.9 系统活动图

平台包括用户端和管理端两部分，各部分的活动图如下：

**管理端：**

图5：表情包图鉴平台管理端活动图

**用户端：**

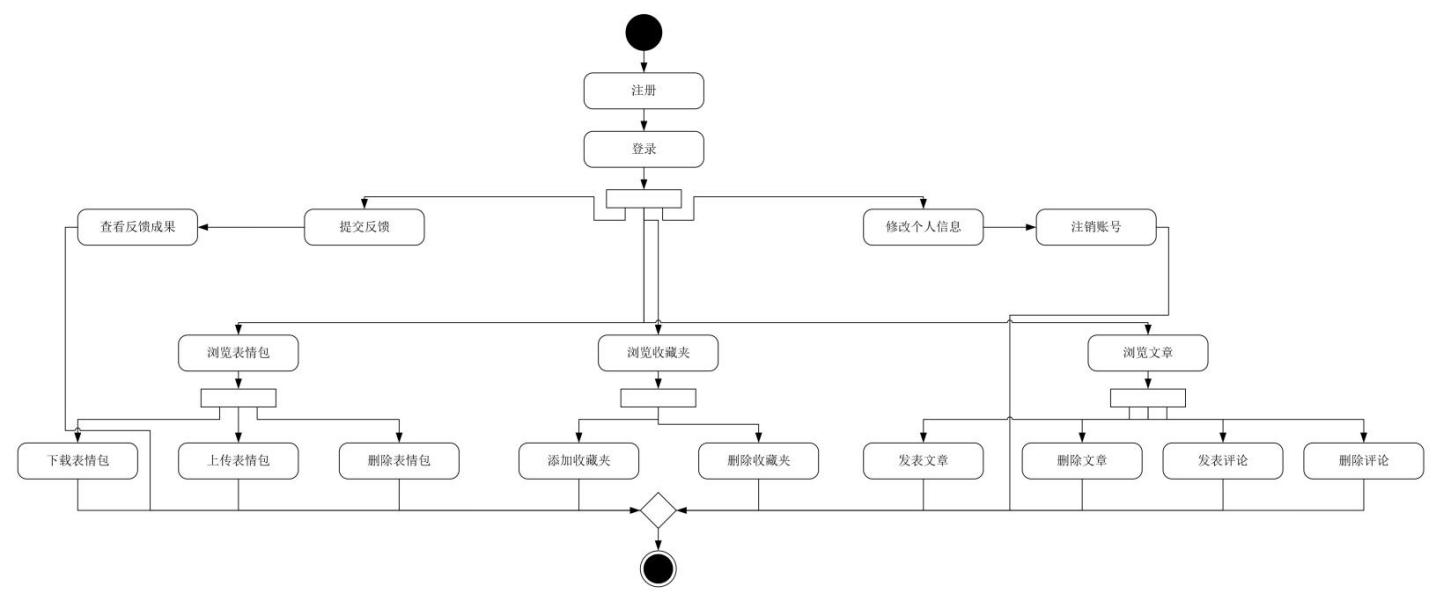


图6：表情包图鉴平台用户端活动图

## 2.10 系统时序图

平台包括用户端和管理端两部分，各部分的时序图如下：

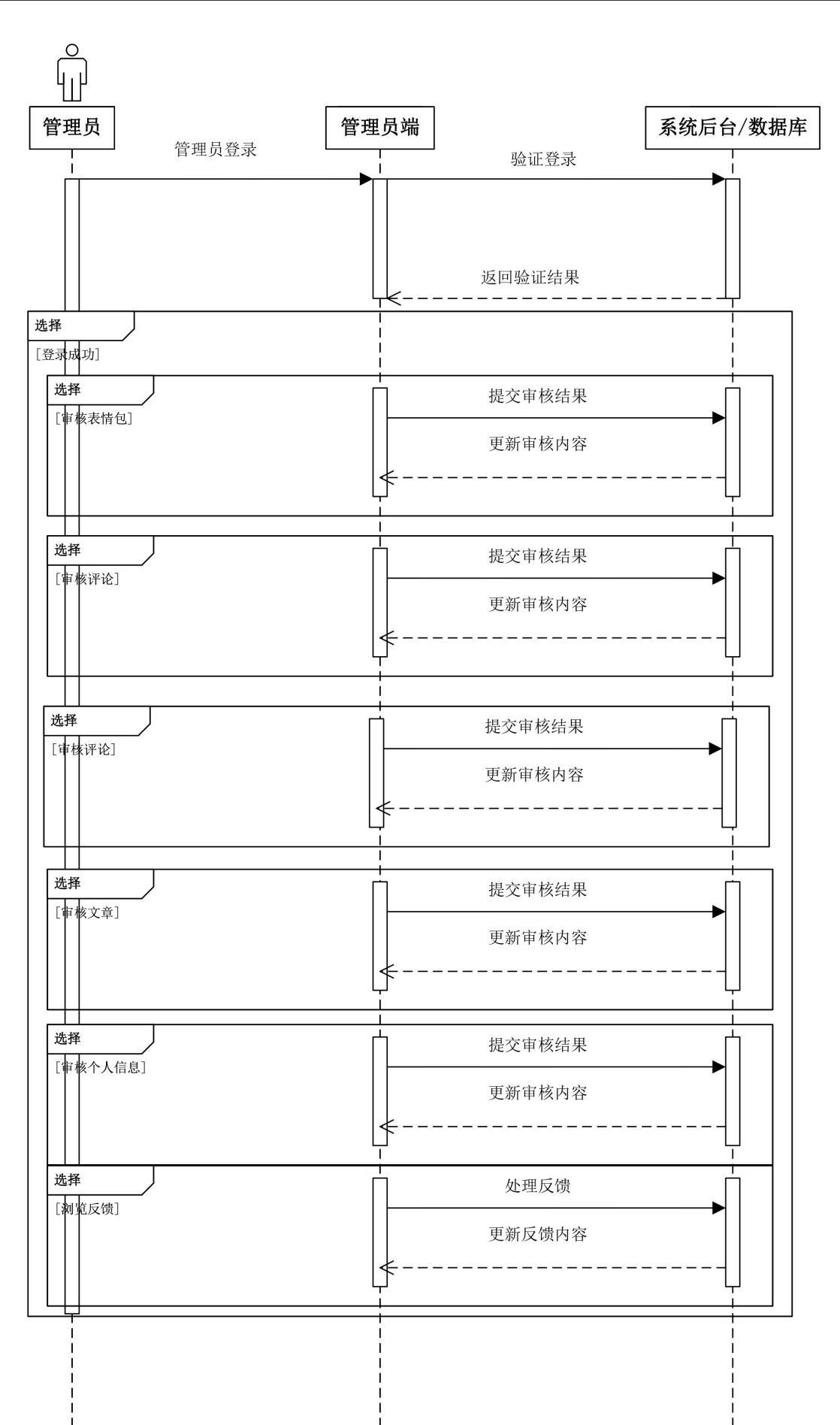
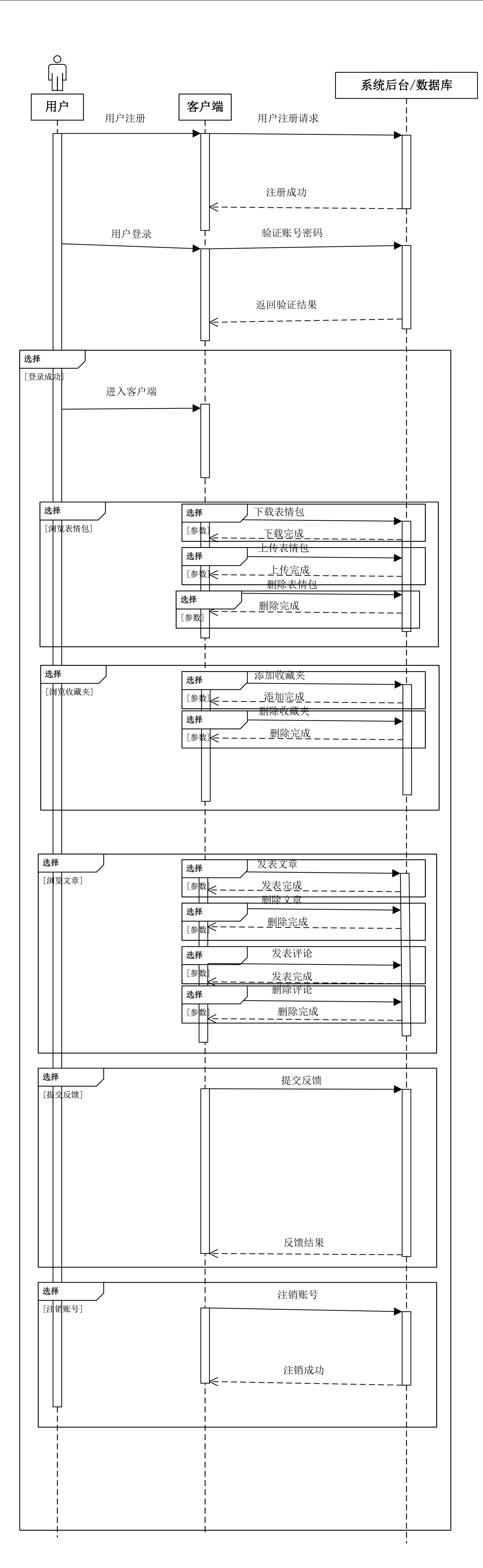
**管理端：**

图7：表情包图鉴平台管理端时序图

**用户端：**



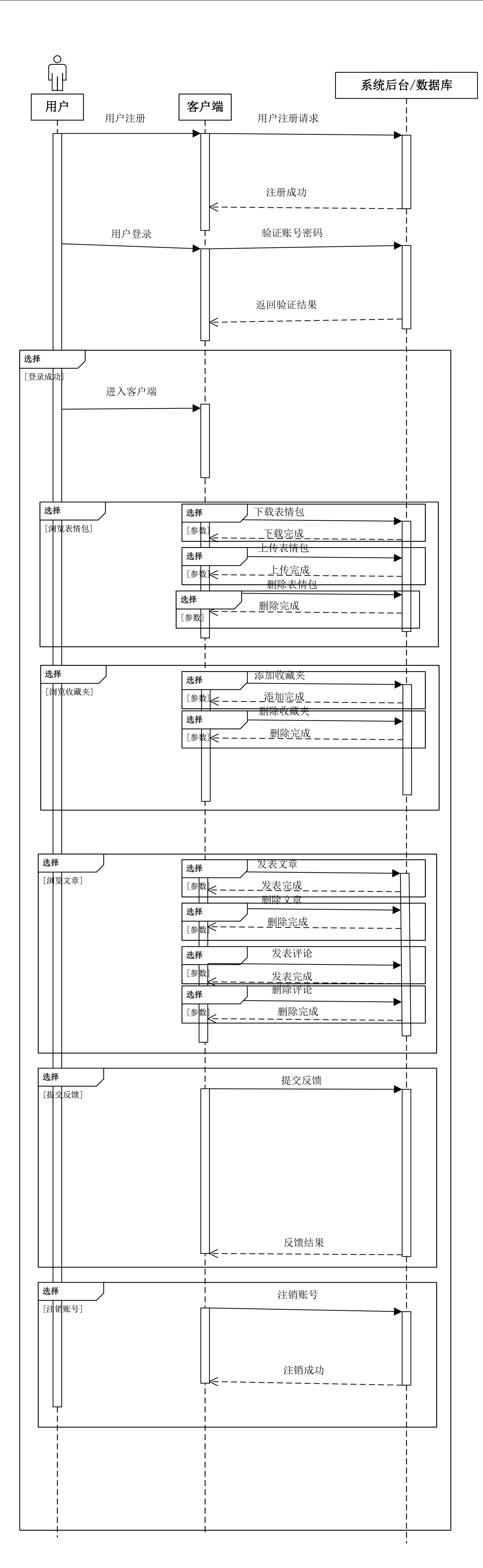
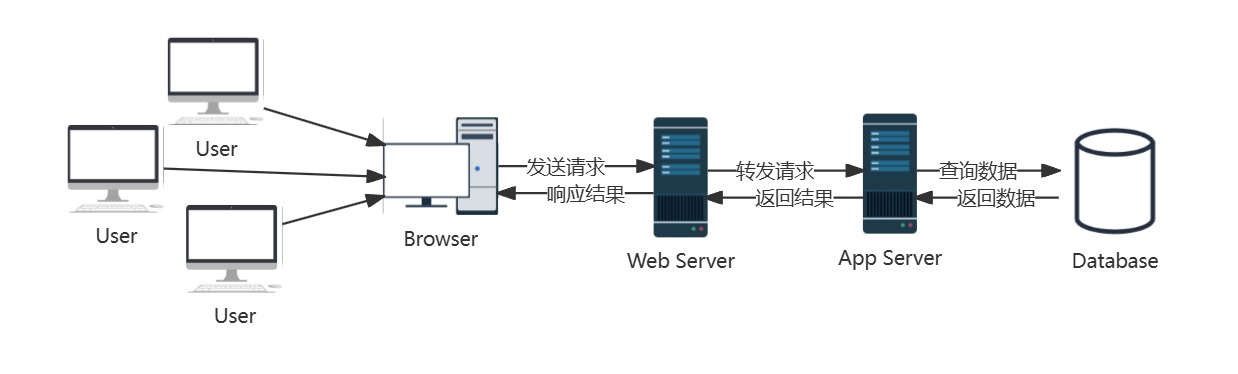


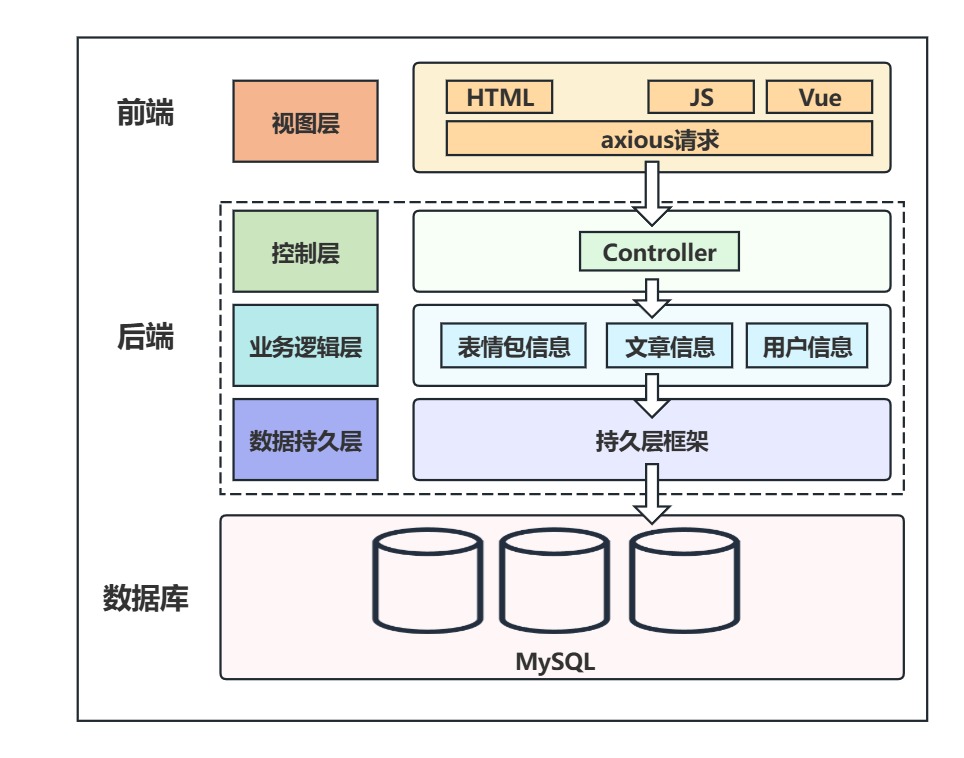
图8：表情包图鉴平台用户端时序图

## 2.11 系统架构

本平台采用B/S架构，使用浏览器请求、浏览器反馈的工作模式。在B/S架构系统中，用户通过浏览器向分散在网络上的服务器发送请求，服务器识别浏览器的请求并做出反馈，将需要的信息发回到浏览器。而其它的如数据请求、加工、结果返回和动态网页的产生、对数据库的调用和应用程序的响应等操作都需要由应用服务器进行完成。B/S架构示意图如下：

图9：B/S架构示意图

结合了所用技术以及前端、后端、数据库的连接，本系统的整体架构图如下：

图10：表情包图鉴平台整体架构图

## 2.12 数据字典

根据上文分析，我们建立了如下数据字典，包括User（用户管理表）、Article（文章管理表）、Meme（表情包管理表）、Feedback（反馈管理表）、Collection（反馈收集表）、Comment（评论管理表）。具体如下表所示：

表7：表情包图鉴平台系统数据字典

User:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否可为空** | **主键** | **描述** |
| **user\_id** | BIGINT | 10 | 否 | 是 | 用户ID |
| **user\_name** | VARCHAR | 50 | 否 |  | 用户名 |
| **user\_pswd** | VARCHAR | 100 | 否 |  | 密码 |
| **email** | VARCHAR | 100 | 是 |  | 用户邮箱 |
| **mobile** | VARCHAR | 30 | 是 |  | 用户手机号 |

Article:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否可为空** | **主键** | **描述** |
| **article\_id** | BIGINT | 10 | 否 | 是 | 文章ID |
| **article\_name** | VARCHAR | 50 | 否 |  | 文章标题 |
| **article\_category** | VARCHAR | 30 | 是 |  | 文章分类 |
| **article\_content** | TEXT |  | 否 |  | 文章内容 |

Meme:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否可为空** | **主键** | **描述** |
| **meme\_id** | BIGINT | 10 | 否 | 是 | 表情包ID |
| **meme\_name** | VARCHAR | 100 | 否 |  | 表情包标题 |
| **meme\_style** | VARCHAR | 30 | 是 |  | 表情包风格 |
| **meme\_image** | VARCHAR | 255 | 否 |  | 表情包图片(地址) |

Feedback:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否可为空** | **主键** | **描述** |
| **feedback\_id** | BIGINT | 10 | 否 | 是 | 反馈ID |
| **feedback\_name** | VARCHAR | 50 | 是 |  | 反馈主题 |
| **feedback\_content** | TEXT |  | 否 |  | 反馈内容 |
| **feedback\_time** | DATETIME |  | 否 |  | 反馈时间 |

Collection:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否可为空** | **主键** | **描述** |
| **collection\_id** | BIGINT | 10 | 否 | 是 | 反馈ID |
| **collection\_name** | VARCHAR | 50 | 是 |  | 反馈主题 |

Comment:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否可为空** | **主键** | **描述** |
| **comment\_id** | BIGINT | 10 | 否 | 是 | 评论ID |
| **comment\_content** | TEXT |  | 否 |  | 评论内容 |
| **comment\_time** | DATETIME |  | 否 |  | 评论时间 |

## 2.13 数据库模式设计

实体集模式：

User(user\_id, user\_name, user\_pswd, email, mobile)

Collection(collection\_id, collection\_name)

Meme(meme\_id, meme\_name, meme\_style, meme\_image)

Comment(comment\_id，comment\_content, comment\_time)

Article(article\_id, article\_name. article\_category, article\_content,)

Feedback(feedback\_id, feedback\_name, feedback\_content, feedback\_time)

关系集模式：

多对多

include(user\_id, collection\_id, meme\_id)

download(user\_id, meme\_id)

多对一/一对多

upload(meme\_id, user\_id)

publish\_comment(article\_id, comment\_id, user\_id)

publish\_article(article\_id, user\_id)

review(article\_id, comment\_id)

submit(feedback\_id, user\_id)

一对一

manage(user\_id, collection\_id)

模式合并后：

User(user\_id, user\_name, user\_pswd, email, mobile, collection\_id)

Collection(user\_id, collection\_id, collection\_name)

Meme(meme\_id, meme\_name, meme\_style, meme\_image, user\_id)

Comment(article\_id, comment\_id, comment\_content, comment\_time, user\_id)

Article(article\_id, article\_name. article\_category, article\_content, user\_id)

Feedback(user\_id, feedback\_id, feedback\_name, feedback\_content, feedback\_time)

include(user\_id, collection\_id, meme\_id)

download(user\_id, meme\_id)

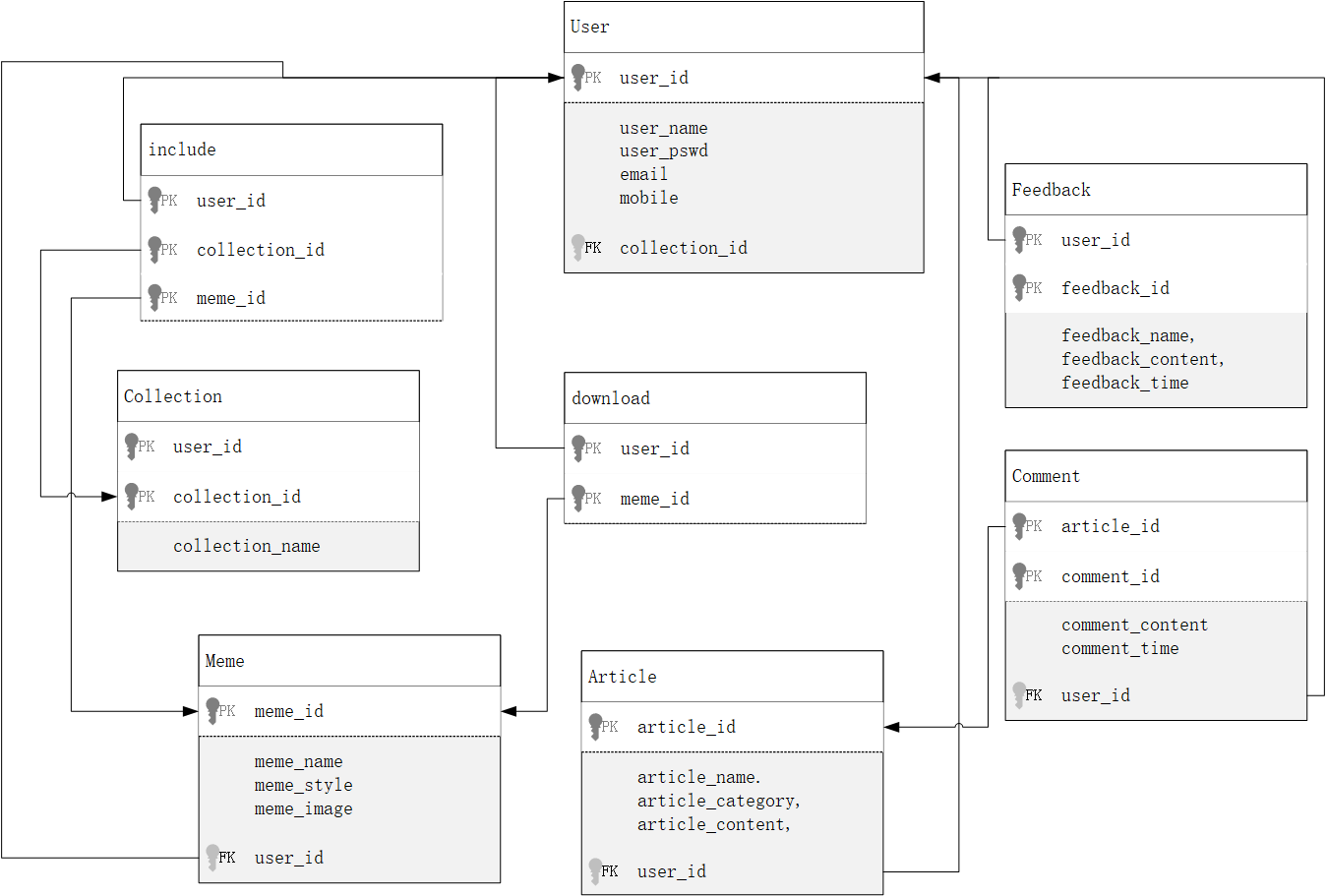
关系模式中每个属性不可分割，没有依赖的传递，且每个非主属性都完全依赖于主码，经此三条判断得出该关系模式属于第三范式。本平台的模式图如下所示：

图11：表情包图鉴平台模式图

## 2.14 数据流设计

**1. 用户注册与登录**

输入: 用户通过前端界面输入 user\_name, user\_pswd, email, mobile。

注册处理流程:

前端将用户输入的数据发送到后端。

后端验证输入数据的合法性（如密码强度、邮箱格式等）。

如果验证通过，后端生成一个唯一的 user\_id，并将 user\_name, user\_pswd, email, mobile 存储到数据库中。

用户注册成功后，系统返回一个成功消息给前端。

输出: 用户注册成功，系统返回成功消息。

登录处理流程:

用户输入 user\_name 和 user\_pswd。

前端将登录信息发送到后端。

后端验证 user\_name 和 user\_pswd 是否匹配数据库中的记录。

如果验证通过，后端生成一个会话令牌（Token）并返回给前端。

前端将令牌存储在本地，用于后续请求的身份验证。

输出: 用户登录成功，系统返回会话令牌。

**2. 用户创建收藏夹**

输入: 用户在前端界面输入 collection\_name。

处理流程:

前端将 collection\_name 和当前用户的 user\_id 发送到后端。

后端生成一个唯一的 collection\_id，并将 collection\_name 和 user\_id 存储到数据库中。

系统返回创建成功的消息给前端。

输出: 收藏夹创建成功，系统返回成功消息。

**3. 用户上传表情包**

输入: 用户在前端界面输入 meme\_name, meme\_style，并上传 meme\_image。

处理流程:

前端将 meme\_name, meme\_style, meme\_image 和当前用户的 user\_id 发送到后端。

后端生成一个唯一的 meme\_id，并将 meme\_name, meme\_style, meme\_image, user\_id 存储到数据库中。

系统返回上传成功的消息给前端。

输出: 表情包上传成功，系统返回成功消息。

**4. 用户评论文章**

输入: 用户在前端界面输入 comment\_content。

处理流程:

前端将 comment\_content 和当前用户的 user\_id 以及目标 article\_id 发送到后端。

后端生成一个唯一的 comment\_id，并将 comment\_content, user\_id, article\_id 存储到数据库中。

系统返回评论成功的消息给前端。

输出: 评论成功，系统返回成功消息。

**5. 用户反馈**

输入: 用户在前端界面输入 feedback\_name, feedback\_content。

处理流程:

前端将 feedback\_name, feedback\_content 和当前用户的 user\_id 发送到后端。

后端生成一个唯一的 feedback\_id，并将 feedback\_name, feedback\_content, user\_id 存储到数据库中。

系统返回反馈成功的消息给前端。

输出: 反馈成功，系统返回成功消息。

**6. 用户下载表情包**

输入: 用户在前端界面选择要下载的 meme\_id。

处理流程:

前端将 meme\_id 和当前用户的 user\_id 发送到后端。

后端根据 meme\_id 从数据库中获取 meme\_image。

后端将 meme\_image 返回给前端。

前端将 meme\_image 提供给用户下载。

输出: 用户成功下载表情包。

通过上述数据流设计，系统能够有效地处理用户注册、登录、创建收藏夹、上传表情包、评论文章、反馈、下载表情包等操作，并将相关数据存储在数据库中。用户在前端界面可以直观地看到操作结果，系统也会记录操作日志以供后续分析。

## 2.15 需求追踪

我们设计了一系列需求追踪*样例*，拟在设计过程中采用该样例进行需求的追踪与进度的评估。具体方案如下：

表8：表情包图鉴平台需求追踪方案

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **需求编号** | **需求描述** | **需求来源** | **优先级** | **当前状态** | **验证方法** | **责任人** |
| **R1** | 上传、浏览、收藏和下载表情包 | 用户 | 高 | 未实现 | 功能测试、用户体验测试 | 开发团队 |
| **R2** | 表情包的快速搜索、分类和个性化管理 | 用户 | 高 | 未实现 | 功能测试、用户体验测试 | 开发团队 |
| **R3** | 查看文档的详细信息，如创建时间、大小、修改历史 | 用户 | 高 | 未实现 | 功能测试、用户体验测试 | 开发团队 |
| **R4** | 用户可以对文档进行评论，并能够发布和修改自己的评论 | 用户 | 高 | 未实现 | 功能测试、用户体验测试 | 开发团队 |
| **R5** | 用户注册、登录与注销功能 | 用户 | 高 | 未实现 | 功能测试、用户体验测试 | 开发团队 |
| **R6** | 通过评论和文章进行社交 | 用户 | 高 | 未实现 | 功能测试、用户体验测试 | 开发团队 |
| **R7** | 管理者能够设置和修改用户的访问权限 | 用户 | 中 | 未实现 | 功能测试、用户体验测试 | 开发团队 |
| **R8** | 用户能够管理自己上传的文档，包括删除和修改 | 用户 | 高 | 未实现 | 功能测试、用户体验测试 | 开发团队 |
| **R9** | 支持多种类型文件的上传和下载 | 开发团队 | 中 | 未实现 | 功能测试 | 开发团队 |
| **R10** | 允许用户创建文件夹以分类和组织文件 | 开发团队 | 中 | 未实现 | 功能测试 | 开发团队 |
| **R11** | 提供版本控制功能，跟踪文件的修改历史 | 开发团队 | 低 | 未实现 | 功能测试 | 开发团队 |
| **R12** | 具备快速搜索引擎，根据文件名、标签和上传时间进行查询 | 用户 | 高 | 未实现 | 功能测试、用户体验测试 | 开发团队 |
| **R13** | 管理员能够定义不同用户或用户组的权限 | 开发团队 | 高 | 未实现 | 功能测试 | 开发团队 |
| **R14** | 响应迅速，支持高并发访问 | 开发团队 | 中 | 未实现 | 性能测试 | 开发团队 |
| **R15** | 保证文件上传和下载速度在可接受范围内 | 用户 | 高 | 未实现 | 性能测试、用户体验测试 | 开发团队 |
| **R16** | 实现强密码保护和用户身份验证机制 | 开发团队 | 高 | 未实现 | 功能测试 | 开发团队 |
| **R17** | 实施数据加密和备份策略，避免数据泄露或丢失 | 开发团队 | 高 | 未实现 | 功能测试 | 开发团队 |
| **R18** | 界面易于使用，确保用户能够迅速上手操作 | 开发团队 | 中 | 未实现 | 用户体验测试 | 开发团队 |
| **R19** | 提供在线帮助和完善的文档支持，解答用户的常见问题 | 开发团队 | 中 | 未实现 | 用户体验测试 | 开发团队 |

# 3 校园表情包图鉴平台系统的软件过程成熟度深度评估

## 3.1 项目背景与评估方法

校园表情包图鉴平台系统是一个面向大学生群体的Web应用系统，是我在《软件需求分析》课程中的大作业项目。作为一个提供集中存储和展示校园表情包的平台，允许用户上传、存储和管理校园相关的表情包，探索表情包在校园文化中的传播与接受方式，也可以发布自己有关表情包的推荐文章，与其他用户进行交流。我们希望通过此平台让大学生共享和传播具有校园特色的表情包，提高校园表情包的可见度和传播效率，促进校园文化的多样性和创新，鼓励学生通过表情包表达自我，也增加学生之间的交流与互动。

结合CMMI的评估标准，我将从多个维度对该项目的软件过程成熟度进行全面评估。评估方法采用了文档分析、过程域映射和成熟度特征对比三种方式。首先，通过仔细研读项目的需求说明书，提取与CMMI过程域相关的证据。其次，将项目的实践与CMMI各个过程域的目标和实践进行映射，判断满足程度。最后，将项目的整体特征与各成熟度级别的特征进行对比，确定最符合的成熟度级别。

## 3.2 需求管理能力评估

从需求管理的角度来看，该项目展现出了较为成熟的管理能力。项目团队系统地收集了需求，采用了访谈、需求研讨会、问卷调查和场景分析等多种需求收集方法。这种多样化的需求收集方式表明团队认识到了不同利益相关者的重要性，并努力从多个角度理解系统需求。

需求文档的结构化程度很高，明确区分了功能性需求和非功能性需求。功能性需求涵盖了个人信息维护、表情包管理、文章管理、系统反馈和管理员功能五大类，每类需求都有详细的子功能描述。非功能性需求则从性能、质量、可用性、安全性、可靠性、可维护性、可扩展性等多个维度进行了定义，并对每个需求进行了优先级评定。

特别值得注意的是，项目建立了需求追踪机制。在需求追踪表中，每个需求都有唯一的编号、明确的描述、来源、优先级、当前状态、验证方法和责任人。这种系统化的需求管理方法不仅有助于确保需求的完整性和一致性，还为后续的变更管理提供了基础。

然而，需求管理也存在一些不足。首先，需求变更管理流程不够明确。虽然有需求追踪表，但没有明确说明当需求发生变更时的处理流程，包括变更的提出、评估、批准和实施等环节。其次，需求的可验证性有待加强。虽然制定了验收准则，但部分需求的验收准则过于笼统，缺乏具体的量化指标。

## 3.3 项目计划与监控评估

项目在计划方面表现出了一定的成熟度。团队制定了基于迭代式开发思想的需求收集计划，将整个过程分为6周的迭代周期，每个周期都有明确的目标和交付物。这种迭代式的开发方法符合现代软件工程的最佳实践，有助于降低项目风险，提高灵活性。

开发环境和技术栈的选择也经过了仔细考虑。项目明确定义了前端使用Vue框架，后端使用SpringBoot框架，数据库采用MySQL和Redis的技术架构。这种技术选择不仅考虑了系统的功能需求，还兼顾了团队的技术能力和项目的约束条件。

然而，项目计划中缺少一些关键要素。首先，没有明确的项目进度计划，包括各个阶段的开始和结束时间、关键里程碑、依赖关系等。其次，资源计划不够详细，虽然列出了团队成员，但没有说明各成员的角色、职责和工作量分配。第三，缺乏风险管理计划，虽然在非功能需求中提到了一些风险，但没有系统的风险识别、评估和应对策略。

项目监控方面的信息较少。文档中没有描述如何跟踪项目进度、如何识别偏差、如何采取纠正措施等。这可能导致项目在执行过程中缺乏有效的控制，增加项目失败的风险。

## 3.4 软件工程实践评估

在软件工程实践方面，项目展现出了向已定义级发展的特征。项目采用了多种UML图来描述系统，包括用例图、ER图、领域模型、活动图、时序图等。这些模型从不同的视角描述了系统的结构和行为，有助于团队成员和利益相关者理解系统。

系统架构设计相对完整，采用了B/S架构，明确了前端、后端和数据库的职责划分。数据库设计遵循了规范化原则，达到了第三范式的要求。这种规范化的设计有助于减少数据冗余，提高数据一致性。

测试方案的设计体现了分层测试的思想，包括模块测试、子系统测试和总体测试三个层次。这种分层的测试策略有助于尽早发现和修复缺陷，提高系统质量。测试覆盖率分析表明团队意识到了测试完整性的重要性。

但是，软件工程实践仍有改进空间。首先，缺乏详细的设计规范和编码标准。虽然选定了技术栈，但没有制定相应的设计原则、编码规范、命名约定等。其次，测试用例设计过于简略，缺乏具体的测试数据、测试步骤和预期结果。第三，没有提及代码审查、持续集成等现代软件工程实践。

## 3.5 质量保证与配置管理评估

质量保证方面，项目的表现相对薄弱。虽然设计了测试方案和验收准则，但缺乏系统的质量保证计划。没有明确的质量目标、质量标准、质量保证活动和质量控制措施。质量保证似乎主要依赖于最后的测试阶段，而不是贯穿整个开发过程。

配置管理是项目的另一个薄弱环节。文档中虽然有版本号和修改记录，但这只是最基本的版本控制。没有完整的配置管理计划，包括配置项识别、版本控制策略、基线管理、变更控制等。在多人协作的项目中，缺乏有效的配置管理可能导致版本混乱、集成困难等问题。

## 3.6 过程标准化程度评估

项目在过程标准化方面显示出了积极的迹象。需求说明书的结构规范，包含了从引言到附录的完整内容。文档中使用了大量的表格来组织信息，如需求表、数据字典、追踪矩阵等，这种结构化的表达方式有助于信息的理解和管理。

UML的使用也体现了对标准建模语言的采用。项目不仅使用了常见的用例图和类图，还使用了活动图、时序图等来描述系统的动态行为。这种多视角的建模方法有助于全面理解系统。

然而，过程标准化还未达到组织级别。当前的标准化更多是项目级的，没有证据表明存在组织级的标准过程、过程资产库、裁剪指南等。这是达到已定义级所必需的要素。

## 3.7 度量与分析能力评估

项目在度量与分析方面的能力较为有限。虽然在非功能需求中定义了一些性能指标，如响应时间、并发用户数等，但缺乏系统的度量计划。没有明确说明如何收集这些数据、如何分析、如何使用分析结果来指导决策。

项目也缺乏过程度量，如生产率、缺陷密度、返工率等。这些度量对于理解和改进开发过程至关重要。没有历史数据的积累，很难进行准确的估算和计划。

## 3.8 团队能力与组织支持评估

团队具备基本的软件开发能力，能够进行需求分析、系统设计、数据库设计等活动。团队成员分工明确，每个人都有特定的职责。这种明确的分工有助于提高工作效率。

但是，团队在某些先进实践方面可能存在经验不足。例如，文档中提到"项目成员开发框架与开发语言掌握不够熟练"，这可能影响项目的进度和质量。另外，没有提到团队的培训计划，如何提升团队成员的技能。

组织支持方面的信息较少。不清楚项目是否得到了充分的资源支持，是否有明确的项目发起人和指导委员会，是否有组织级的过程改进活动等。

## 3.9 综合评估结论

综合以上各方面的分析，我认为我们的校园表情包图鉴平台系统项目的软件过程成熟度处于**CMMI第2级（已管理级）的较高阶段**，正在向第3级（已定义级）过渡。

支持这一判断的主要证据包括：项目建立了基本的需求管理、项目计划、测量分析等管理过程；采用了迭代式开发方法，有基本的计划和跟踪机制；使用了UML等标准建模方法，文档结构规范；设计了分层的测试策略，有基本的质量控制意识。

阻碍项目达到第3级的主要差距包括：缺乏组织级的标准过程和过程资产库；配置管理和风险管理过程不够完善；质量保证活动不够系统化；缺乏定量的项目管理和过程改进机制；团队能力建设和知识管理有待加强。

这个评估结果表明，项目团队已经具备了基本的项目管理能力，但要达到更高的成熟度级别，还需要在过程标准化、风险管理、质量保证等方面进行系统性的改进。

# 3 校园表情包图鉴平台系统的软件过程改进计划

基于当前的成熟度评估，我制定了以下过程改进计划，目标是在未来6-12个月内将软件过程成熟度提升到CMMI第3级（已定义级）。

## 3.1 第一阶段：完善基础管理过程（1-3个月）

首要任务是建立完整的配置管理体系。我们需要选择合适的版本控制工具（如Git），制定分支管理策略，明确代码提交规范。同时建立文档版本管理机制，确保需求文档、设计文档、测试文档等与代码版本保持同步。配置管理不仅包括源代码，还应该包括构建脚本、配置文件、数据库脚本等所有项目相关的工件。

接下来要强化风险管理流程。建立风险登记册，定期进行风险识别和评估。对于每个识别出的风险，制定相应的应对策略，包括规避、转移、缓解或接受。将风险管理融入到项目的日常管理中，在每次迭代计划会议上回顾和更新风险状态。

质量保证体系的建立也是这一阶段的重点。引入代码审查机制，制定代码规范和审查检查单。建立持续集成环境，实现自动化构建和基础的自动化测试。完善测试策略，不仅包括功能测试，还要考虑性能测试、安全测试等非功能性测试。

## 3.2 第二阶段：标准化和优化过程（4-6个月）

首要任务是建立完整的配置管理体系。我们需要选择合适的版本控制工具（如Git），制定分支管理策略，明确代码提交规范。同时建立文档版本管理机制，确保需求文档、设计文档、测试文档等与代码版本保持同步。配置管理不仅包括源代码，还应该包括构建脚本、配置文件、数据库脚本等所有项目相关的工件。

接下来要强化风险管理流程。建立风险登记册，定期进行风险识别和评估。对于每个识别出的风险，制定相应的应对策略，包括规避、转移、缓解或接受。将风险管理融入到项目的日常管理中，在每次迭代计划会议上回顾和更新风险状态。

质量保证体系的建立也是这一阶段的重点。引入代码审查机制，制定代码规范和审查检查单。建立持续集成环境，实现自动化构建和基础的自动化测试。完善测试策略，不仅包括功能测试，还要考虑性能测试、安全测试等非功能性测试。

## 3.3 第三阶段：持续改进和创新（7-12个月）

进入第三阶段，重点转向过程的持续改进和创新。建立过程改进机制，定期回顾项目执行情况，识别过程中的问题和改进机会。可以采用回顾会议、根因分析等方法来发现问题的深层原因。

探索和引入新的技术和方法也是这一阶段的重要内容。例如，可以尝试采用DevOps实践，实现开发和运维的紧密协作；引入微服务架构，提高系统的可扩展性和可维护性；采用容器化技术，简化部署和环境管理。

建立创新激励机制，鼓励团队成员提出改进建议和创新想法。可以设立创新基金，支持有价值的改进项目。定期举办黑客马拉松等活动，激发团队的创新热情。

## 3.4 改进计划的实施保障

为确保改进计划的顺利实施，需要建立相应的保障机制。首先是组织支持，需要获得管理层的认可和支持，确保有足够的资源投入到过程改进中。其次是建立改进团队，指定专人负责推动各项改进措施的落实。定期召开改进进展评审会议，跟踪改进措施的实施情况，及时调整改进策略。

改进过程中要注意平衡，不能因为过程改进而影响项目的正常交付。改进措施应该逐步推进，先在小范围试点，取得成功后再推广到整个组织。同时要注意收集反馈，了解团队成员对改进措施的接受程度，及时解决实施过程中遇到的问题。

通过这个系统化的改进计划，我相信能够将软件开发过程的成熟度提升到一个新的水平。这不仅能够提高项目的成功率，降低项目风险，还能够提升团队的整体能力，为组织的长期发展奠定坚实的基础。过程改进是一个持续的过程，需要全体团队成员的共同努力和不断坚持。只有这样，才能真正实现软件开发过程的规范化、标准化和持续优化。