# 软件需求分析

# 1引言

互联网的普及与应用，使动漫产业得到了难得的发展机遇，随着产业的扩大与升级，动漫产业所影响和覆盖的范围已遍及全球，无比广阔的市场造就了动漫游戏产业所独有的二次元文化。

作为二次元文化的重要组成部分之一，动漫影评游戏评价不仅能反应作品的制作质量与水平，更为体验者提供了表达主观感受的机会，让交流，分享个人理解与感受成为可能。然而，当前市场现存的二次元文化论坛大多存在内容单一、互动性较弱、用户体验欠佳等问题，不足以面对迅速发展与扩大的动漫游戏市场，难以满足二次元爱好者日益增长的线上需求。

鉴于此，我们提出设计一款专注于二次元论坛开发的课程项目。本项目旨在打造一个内容丰富、互动性强、用户体验良好的论坛平台，为二次元爱好者提供一个线上文化交流的场所。通过本项目的实施，我们期望能在一定程度上推动二次元文化的传播与发展，促进二次元文化爱好者之间的沟通与交流，构建一个更加和谐多元的“二次元精神世界”。

# 1.1项目背景

随着互联网技术的发展与二次元文化的兴起，越来越多的年轻人因为诸多动漫影视作品及游戏对二次元文化产生了浓厚的兴趣。而作为影视以及游戏作品的重要组成部分，论坛成为了许多二次元文化爱好者交流观点、分享感受的沟通桥梁。然而，目前市场上的二次元论坛大多存在内容同质化、互动性差、用户体验不佳等问题。因此，设计一款以二次元爱好者为受众群体、内容丰富、互动性强、用户体验良好的影评论坛势在必行。

# 1.2目的

**①普通用户方面：**

帮助用户事先了解番剧或游戏的质量水平，为其是否选择观影或游玩提供可靠的判断依据，并为经验者提供 发表主观体验感受和各自观点的 线上交流平台，从而促进其之间的思想碰撞，加深用户对于某些作品剧情情节等方面的认知，如此在满足用户观影游玩需求的同时，也拓宽了用户的眼界与视野，提升了其鉴赏水平。

**②作品创作者方面：**

创作者也可通过大众评论了解自身作品所受到的主体评价，可以据此改良作品，其番剧或游戏也能在平台提供给普通用户的浏览与讨论模块中得到有效的宣传与推广

**③相关行业方面：**

二次元文化行业也能依据平台的大数据把握市场趋势和观众喜好，做出相应的行业规划。

# 2需求概述

**需求1:** 用户在接触陌生的番剧或游戏作品前，希望事先通过查看相关评论了解其质量、风格、主题等信息，以作为是否值得投入精力的判断依据。、

**需求2:** 用户完成动漫或游戏的体验后，存在仍有强烈的欲望表达自己主观感受、见解和评价的群体，他们有与他人分享自身体验经历与自我表达的需求。

# 2.1应用场景

**场景1:**  用户可借此寻找优质内容，根据经验玩家和观众的评价与推荐来选择适合自己的游戏或番剧。而特定兴趣用户也能精准选择符合自己喜好的作品。

**场景2:** 而用户分享体验见解和参与讨论互动的这些模块，可以为创作者和开发者提供有效的依据从而了解用户的需求与建议，从而在后续作品中进行有效调整与改进。

**场景3:** 用户可以通过软件结识志同道合的爱好者，在平台所提供的交流模块中，分享动漫和游戏的乐趣，也可以通过软件了解动漫和游戏行业的最新动态、热门作品和趋势。

# 2.2功能特性

**Ⅰ. 用户注册与登录功能**

**功能名称：** 用户注册与登录

**功能概述：** 允许用户注册、登录论坛，进行个性化操作（如发帖、回帖、资源下载等）。用户可以选择普通用户、资源发布者等身份。

**用户角色：**

​                ●  访客

​                ●  普通用户

​                ●  资源发布者

​                ●  管理员

**前提条件：**

​                ●  访客需要注册才能发布内容或下载资源。

**主要流程：**

​                ●  用户打开注册页面，输入用户名、密码、邮箱等信息。

​                ●  系统验证信息是否符合规范，如用户名唯一，邮箱格式正确。

​                ●  系统向用户发送激活邮件，用户通过点击邮件中的链接激活账户。

​                ●  激活后，用户可以登录论坛。

**边界情况：**

​                ●  如果用户名已存在，系统提示用户更换。

​                ●  如果密码不符合安全标准（如长度不足或缺少特殊字符），提示用户修改。

**输入与输出：**

​                ●  输入：用户名、邮箱、密码

​                ●  输出：注册成功、错误提示、登录页面

**性能要求：**

​                ●  登录请求处理时间不超过3秒。

**安全性：**

​                ●  所有敏感信息（如密码）都需要加密存储。

​                ●  支持两步验证以提高安全性。

**Ⅱ. 发帖与讨论功能**

**功能名称：**发帖与讨论

**功能概述：**

允许用户在论坛中分享和讨论与二次元技术相关的内容（如动画制作、视频剪辑、建模、渲染等技术话题）。

**用户角色：**

​                ●  普通用户

​                ●  吧主

​                ●  管理员

**前提条件：**

​                ●  用户已登录。

**主要流程：**

​                ●  用户进入技术讨论板块，点击“发帖”按钮。

​                ●  用户填写帖子标题、内容，并附加相关图片、代码片段或视频链接。

​                ●  系统验证帖子是否符合论坛规则（如内容是否合法、无恶意代码）。

​                ●  帖子发布成功后，其他用户可以对帖子进行评论、点赞、收藏。

**边界情况：**

​                ●  用户发帖内容不符合要求时，系统提示用户修改。

​                ●  发帖内容包含违法内容时，管理员可直接删除或警告。

**输入与输出：**

​                ●  输入：帖子标题、内容、附件（图片、代码），支持Markdown格式

​                ●  输出：帖子发布成功提示、评论、点赞等交互元素

**性能要求：**

​                ●  发帖请求处理时间不超过3秒。

**安全性：**

​                ●  用户发布的内容经过审核机制（如自动化过滤或管理员审核）。

​                ●  对用户发布的代码片段进行过滤，防止XSS等攻击。

**Ⅲ. 积分与用户等级系统**

**功能名称：** 积分与用户等级系统

**功能概述：** 通过用户的活跃度（如发布资源、发帖、评论、参与讨论等）积累积分，并根据积分提升用户等级，解锁更多权限（如更多下载次数、上传权限等）。

**用户角色：**

* ● 普通用户
* ● 管理员

**前提条件：**

  ● 用户需要登录，并参与论坛活动。

**主要流程：**

● 用户参与论坛的各种活动（如发帖、回帖、点赞、下载等）会积累积分。

● 系统根据用户的积分自动提升等级，并在用户个人资料页面显示当前等级和下一等级的要求。

● 用户等级提升后，系统解锁更多权限，如更高的每日下载上限、资源上传权限等。

**边界情况：**

● 如果用户积分达到上限后，系统发送通知提示用户升级。

● 管理员可根据论坛的运营情况调整积分规则。

**输入与输出：**

● **输入**：用户操作（发帖、评论等）

● **输出**：积分累加、等级提升通知

**性能要求：**

 ● 积分累加应在每次操作后立即更新。

**安全性：**

 ● 防止刷分行为，系统应对异常操作进行检测。

# 2.3非功能需求

1. **性能需求**

响应速度：论坛的页面加载时间应控制在合理范围内，一般不超过 3 秒，以确保用户能够快速浏览和交互。

并发处理能力：能够同时处理大量用户的访问和操作，在高峰时段也能保持稳定运行，避免出现卡顿或崩溃现象

1. **安全性需求**

用户数据安全：严格保护用户的个人信息，包括用户名、密码、邮箱等，采用加密技术存储和传输数据，防止数据泄露。

防范恶意攻击：具备防范常见网络攻击的能力，如 DDoS 攻击、SQL 注入等，确保论坛的安全性和稳定性。

内容审核：建立有效的内容审核机制，过滤不良信息、违规内容和恶意言论，营造健康的交流环境。

1. **可用性需求**

界面友好：论坛的界面设计应简洁、美观、易用，符合二次元风格，方便用户操作和浏览。

跨平台兼容性：能够在不同的操作系统和设备上正常运行，包括电脑、手机、平板等，满足用户多样化的使用需求。

可访问性：考虑到不同用户的需求，确保论坛具有良好的可访问性，例如为视力障碍用户提供辅助功能。

1. **可靠性需求**

数据备份与恢复：定期对论坛的数据进行备份，以防止数据丢失，同时具备快速恢复数据的能力，确保在出现故障时能够尽快恢复正常运行。

系统稳定性：保证论坛系统的稳定运行，减少故障发生的概率，及时修复出现的问题，确保用户能够持续使用论坛。

1. **可维护性需求**

代码可维护性：论坛的代码应具有良好的结构和可读性，便于开发人员进行维护和升级。

系统扩展性：随着用户数量的增加和业务的发展，论坛应具备良好的扩展性，能够方便地添加新功能和模块。

1. **用户体验需求**

个性化设置：允许用户根据自己的喜好进行个性化设置，如主题颜色、字体大小、通知偏好等，提高用户的满意度。

互动体验：提供丰富的互动功能，如点赞、评论、分享、私信等，增强用户之间的交流和互动。

搜索功能：具备高效的搜索功能，方便用户快速找到自己感兴趣的内容和话题。

# 3数据需求分析

**Ⅰ用户数据需求**

**①  用户表（users）：**

用户 ID（user\_id）： 用于系统内部唯一识别用户，方便管理用户信息和关联其他表中的用户数据。

用户名（username）： 用户在系统中的显示名称，用于用户之间的识别和交流。

电子邮箱（email）： 作为用户注册和登录的重要凭证，同时可用于接收系统通知和密码找回等。

角色（role）： 确定用户在系统中的权限级别，例如管理员可以管理内容和用户，普通用户只能进行评论和点赞等操作。

个人简介（bio）： 让用户能够展示自己的兴趣爱好、游戏或动漫经历等，促进用户之间的了解。

头像 URL（avatar\_url）： 为用户提供个性化的展示，增强用户在系统中的辨识度。

创建时间（created\_at）： 记录用户加入系统的时间，可用于分析用户增长趋势等。

最后登录时间（last\_login）： 用于了解用户的活跃程度和登录频率。

密码哈希（password\_hash）： 保障用户密码的安全性，防止密码被直接获取。

**Ⅱ 内容数据需求**

**① 帖子表（posts）：**

帖子 ID（post\_id）： 唯一标识每个帖子，方便系统管理和用户查找。

用户 ID（user\_id）： 关联到用户表，确定帖子的发布者，便于追溯和管理帖子来源。

标题（title）： 简洁地概括帖子的主题，吸引用户点击阅读。

内容（content）： 帖子的具体正文，包含用户对动漫或游戏的评价、讨论等。

浏览量（views）： 统计帖子的被查看次数，反映帖子的受欢迎程度。

创建时间（created\_at）： 记录帖子的发布时间，可用于排序和了解帖子的时效性。

更新时间（updated\_at）： 当帖子内容被修改时更新，方便用户查看最新的讨论内容。

分类 ID （category\_id）： 将帖子分类管理，方便用户据不同的动漫或游戏类型查找相关帖子。

**② 评论表（comments）：**

帖子 ID（post\_id）： 关联到帖子表，确定评论所针对的帖子，形成帖子与评论的关联关系。

用户 ID（user\_id）： 关联到用户表，确定评论的发布者，方便用户之间的交流和互动。

内容（content）： 评论的具体内容，用户对帖子或其他评论的回应和讨论。

创建时间（created\_at）：记录评论的发布时间，用于展示评论的时间顺序。

评论 ID（comment\_id）：唯一标识每条评论，便于管理和查找。

**Ⅲ 分类与标签数据需求**

**① 分类表（categories）：**

分类名称（name）： 明确分类的主题，如不同的动漫类型、游戏类型等，方便用户快速定位感兴趣的内容。

分类 ID（category\_id）： 唯一标识每个分类，方便系统管理和用户查找。

创建时间（created\_at）： 记录分类的创建时间，可用于分析系统内容的发展和变化。

分类描述（description）： 对分类进行简要说明，帮助用户更好地理解分类的范围和特点。

**② 标签表（tags）：**

标签 ID（tag\_id）： 唯一标识每个标签，方便系统管理和使用。

标签名称（name）： 简洁地描述标签的主题，如“热门动漫”“策略游戏”等，方便用户快速筛选和查找相关内容。

**③ 帖子标签关联表（post\_tags）：**

帖子 ID（post\_id）： 关联到帖子表，确定带有标签的帖子。

标签 ID（tag\_id）： 关联到标签表，确定帖子所关联的标签，实现帖子的多标签分类，方便用户从不同角度查找和筛选内容。

Ⅳ 互动数据需求

1. **私信表（messages）：**

私信 ID（message\_id）： 唯一标识每条私信，方便系统管理和查找。

发送者 ID（sender\_id）： 关联到用户表，确定私信的发送者。

接收者 ID（receiver\_id）： 关联到用户表，确定私信的接收者。

内容（content）： 私信的具体正文，用户之间的私密交流内容。

发送时间（sent\_at）： 记录私信的发送时间，方便用户查看私信的时间顺序。

1. **点赞表（likes）：**

点赞 ID（like\_id）： 唯一标识每条点赞记录，方便系统管理和统计。

用户 ID（user\_id）： 关联到用户表，确定点赞的用户。

帖子 ID（post\_id）： 关联到帖子表，确定被点赞的帖子，可用于统计帖子的受欢迎程度。

评论 ID（comment\_id）： 关联到评论表，确定被点赞的评论，同样可用于统计评论的受欢迎程度。

创建时间（created\_at）： 记录点赞的时间，可用于分析用户的互动行为和趋势。

**Ⅴ 权限数据需求**

1. 权限表（permissions）：

权限 ID（permission\_id）： 唯一标识每个权限，方便系统管理和分配权限。

角色（role）： 定义不同的用户权限类别，如管理员、版主、普通用户等。

权限名称（permission\_name）： 具体描述每个权限的功能，如“管理帖子”“删除评论”等。

权限描述（description）： 对每个权限进行简要说明，帮助管理员更好地理解和分配权限。

# 3.1系统用例分析

# 3.2数据字典

数据库中各个键值的中文解释：

**1. 用户表 (users)**

user\_id: 用户ID，唯一标识每个用户的编号。

username: 用户名，用户的登录名或显示名，必须唯一。

email: 电子邮箱，用户的电子邮箱地址，必须唯一。

password\_hash: 密码哈希，用户的密码经过加密后的值。

bio: 个人简介，用户的个人描述或自我介绍。

avatar\_url: 头像URL，用户头像的图片链接。

role: 角色，用户在论坛中的权限类型（如管理员、版主、普通用户）。

created\_at: 创建时间，用户注册的时间。

last\_login: 最后登录时间，用户上次登录的时间。

**2. 帖子表 (posts)**

post\_id: 帖子ID，唯一标识每个帖子的编号。

user\_id: 用户ID，发帖人的唯一编号，关联到用户表的user\_id。

category\_id: 分类ID，帖子所属分类的编号，关联到分类表的category\_id。

title: 标题，帖子的标题或主题。

content: 内容，帖子的正文内容。

created\_at: 创建时间，帖子发布的时间。

updated\_at: 更新时间，帖子的最后一次修改时间。

views: 浏览量，帖子被浏览的次数。

**3. 评论表 (comments)**

comment\_id: 评论ID，唯一标识每条评论的编号。

post\_id: 帖子ID，评论所对应的帖子编号，关联到帖子表的post\_id。

user\_id: 用户ID，评论发布者的编号，关联到用户表的user\_id。

content: 内容，评论的具体内容。

created\_at: 创建时间，评论发布的时间。

**4. 分类表 (categories)**

category\_id: 分类ID，唯一标识每个分类的编号。

name: 分类名称，分类的名称。

description: 分类描述，对分类的简要说明。

created\_at: 创建时间，分类创建的时间。

**5. 标签表 (tags)**

tag\_id: 标签ID，唯一标识每个标签的编号。

name: 标签名称，标签的名称（如“技术”、“新闻”等）。

**6. 帖子标签关联表 (post\_tags)**

post\_id: 帖子ID，关联的帖子编号，关联到帖子表的post\_id。

tag\_id: 标签ID，关联的标签编号，关联到标签表的tag\_id。

**7. 私信表 (messages)**

message\_id: 私信ID，唯一标识每条私信的编号。

sender\_id: 发送者ID，发送私信的用户编号，关联到用户表的user\_id。

receiver\_id: 接收者ID，接收私信的用户编号，关联到用户表的user\_id。

content: 内容，私信的具体内容。

sent\_at: 发送时间，私信发送的时间。

**8. 点赞表 (likes)**

like\_id: 点赞ID，唯一标识每条点赞记录的编号。

user\_id: 用户ID，点赞的用户编号，关联到用户表的user\_id。

post\_id: 帖子ID，点赞的帖子编号，关联到帖子表的post\_id，可以为空。

comment\_id: 评论ID，点赞的评论编号，关联到评论表的comment\_id，可以为空。

created\_at: 创建时间，点赞发生的时间。

**9. 权限表 (permissions)**

permission\_id: 权限ID，唯一标识每个权限的编号。

permission\_name: 权限名称，具体权限的名称。

role: 角色，定义用户的权限类别（如管理员、版主、普通用户）。

description: 权限描述，对该权限的简要说明。表字段解释：

**10. 收藏帖子表（Favorites）**

favorite\_id: 每条收藏记录的唯一标识。

user\_id: 该条收藏记录所属的用户，外键关联 users 表的 user\_id。

post\_id: 被收藏的帖子，外键关联 posts 表的 post\_id。

created\_at: 用户收藏帖子时的时间戳。

**11.账户积分表（Points）**

point\_id: 每条积分变动的唯一标识。

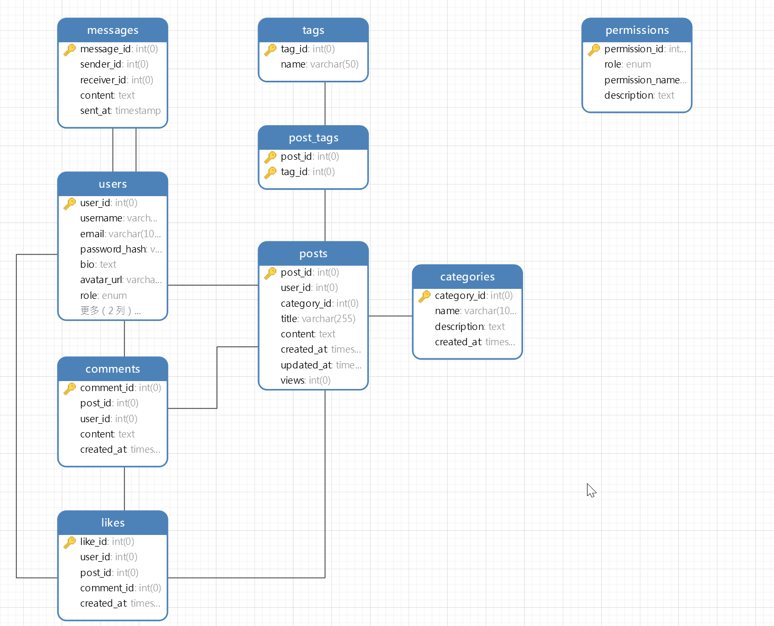
user\_id: 该条积分记录所属的用户，外键关联 users 表的 user\_id。

points: 积分的数值，正数表示增加积分，负数表示扣除积分。

action\_type: 描述该积分记录的来源，例如“发帖”、“评论”、“资源下载”等。

created\_at: 积分变动的时间戳。

ER图：



**用户表 (users)**

**主键**: user\_id

**关联关系**:

posts.user\_id 外键指向 users.user\_id

comments.user\_id 外键指向 users.user\_id

messages.sender\_id 外键指向 users.user\_id

messages.receiver\_id 外键指向 users.user\_id

likes.user\_id 外键指向 users.user\_id

**帖子表 (posts)**

**主键**: post\_id

**外键**:

user\_id 外键指向 users.user\_id

category\_id 外键指向 categories.category\_id

**关联关系**:

comments.post\_id 外键指向 posts.post\_id

post\_tags.post\_id 外键指向 posts.post\_id

likes.post\_id 外键指向 posts.post\_id

**评论表 (comments)**

**主键**: comment\_id

**外键**:

post\_id 外键指向 posts.post\_id

user\_id 外键指向 users.user\_id

**关联关系**:

likes.comment\_id 外键指向 comments.comment\_id

**分类表 (categories)**

**主键**: category\_id

**关联关系**:

posts.category\_外键指向 categories.category\_id

**标签表 (tags)**

**主键**: tag\_id

**关联关系**:

post\_tags.tag\_id 外键指向 tags.tag\_id

**帖子标签关联表 (post\_tags)**

**联合主键**: post\_id, tag\_id

**外键**:

post\_id 外键指向 posts.post\_id

tag\_id 外键指向 tags.tag\_id

**私信表 (messages)**

**主键**: message\_id

**外键**:

sender\_id 外键指向 users.user\_id

receiver\_id 外键指向 users.user\_id

**点赞表 (likes)**

**主键**: like\_id

**外键**:

user\_id 外键指向 users.user\_id

post\_id 外键指向 posts.post\_id (可选)

comment\_id 外键指向 comments.comment\_id (可选)

**收藏帖子表 (favorites)**

**主键**: favorite\_id

**外键**: user\_id 外键指向 users.user\_id

      post\_id 外键指向 posts.post\_id

**关联关系**:

用户和帖子之间的多对多关系（用户可以收藏多个帖子，帖子可以被多个用户收藏）

**积分表 (points)**

**主键**: point\_id

**外键**: user\_id 外键指向 users.user\_id

**字段**:   points 记录积分数值

      action\_type 表示积分变动的类型（如发帖、评论、下载资源等）

**关联关系**:

用户和积分记录之间的一对多关系（一个用户可以有多个积分变动记录）

# 4需求分析

# 功能需求

1. **用户管理**

用户注册与登录： 允许用户通过邮箱注册账号并登录系统，确保账号的唯一性和安全性。

个人资料管理： 用户可以编辑自己的用户名、头像、个人简介等信息，展示个性。

角色与权限管理： 区分不同用户角色（如普通用户、管理员等），赋予相应的权限，如管理员可以管理内容和用户。

1. **内容管理**

帖子发布与编辑： 用户能够发布关于动漫或游戏的帖子，包括标题、内容和选择分类，同时可以对自己发布的帖子进行编辑。

评论发布与编辑： 用户可以在帖子下方发布评论，对其他用户的评论进行回复，并且能够编辑自己的评论。

内容搜索与筛选： 提供搜索功能，用户可以通过关键词搜索帖子和评论，同时可以根据分类、标签等进行筛选。

1. **分类与标签管理**

分类创建与编辑： 管理员可以创建不同的动漫和游戏分类，如动作类、角色扮演类等，并对分类进行编辑和管理。

标签添加与管理： 用户和管理员可以为帖子添加标签，方便用户快速找到感兴趣的内容，同时可以对标签进行编辑和管理。

1. **互动管理**

私信功能： 用户之间可以发送私信进行交流，确保私信的隐私性和及时性。

通知功能： 当用户收到私信、被回复或帖子被点赞时，系统能及时通知用户。

点赞： 用户可以对帖子和评论进行点赞，表达自己的喜好和态度，同时可以查看点赞数量的数量。

1. **数据统计与分析**

浏览量统计： 统计帖子和评论的浏览量，了解用户的关注度和热门内容。

用户行为分析： 通过分析用户的注册、登录、发布内容、互动等行为，了解用户需求和行为习惯，为系统优化提供依据。

点赞统计： 分析帖子和评论的点赞和踩的数量，评估内容的质量和受欢迎程度。

# 性能需求

响应速度： 确保用户在访问网站或APP时，页面加载速度快，操作流畅。

数据安全： 确保用户数据的安全性和隐私性，防止数据泄露和非法访问。

并发处理能力： 在高并发情况下，系统能够稳定运行，不会出现卡顿或崩溃现象。

# ③接口需求

**（一）用户接口需求**

用户注册接口功能：允许用户注册新账号。

输入参数：用户名、密码、邮箱等。

输出参数：注册成功状态码、用户 ID。

调用方式：POST 请求，发送到 /api/user/register。

用户登录接口功能：验证用户登录信息。

输入参数：用户名、密码。

输出参数：登录成功状态码、用户信息（包括用户 ID、用户名、头像等）、令牌（用于后续接口调用的身份验证）。调用方式：POST 请求，发送到 /api/user/login。

用户信息查询接口功能：获取用户详细信息。

输入参数：用户 ID（或令牌中的用户信息）。输出参数：用户详细信息，如用户名、邮箱、注册时间、发帖数量、回帖数量等。

调用方式：GET 请求，发送到 /api/user/{userId}。

用户信息修改接口功能：允许用户修改个人信息。

输入参数：用户 ID（或令牌中的用户信息）、要修改的信息（如用户名、密码、邮箱、头像等）。

输出参数：修改成功状态码。

调用方式：PUT 请求，发送到 /api/user/{userId}。

**（二）帖子接口需求**

帖子列表查询接口功能：获取论坛中的帖子列表。

输入参数：分页参数（页码、每页数量）、筛选条件（如分类、标签、发布时间等）。

输出参数：帖子列表，包括帖子 ID、标题、作者、发布时间、内容摘要等。

调用方式：GET 请求，发送到 /api/post/list。

帖子详情查询接口功能：获取特定帖子的详细内容。

输入参数：帖子 ID。

输出参数：帖子详细信息，包括标题、作者、发布时间、内容、回复列表等。

调用方式：GET 请求，发送到 /api/post/{postId}。

发帖接口功能：用户发布新帖子。

输入参数：用户 ID（或令牌中的用户信息）、标题、内容、标签等。

输出参数：发布成功状态码、帖子 ID。

调用方式：POST 请求，发送到 /api/post/create。

回帖接口功能：用户回复特定帖子。

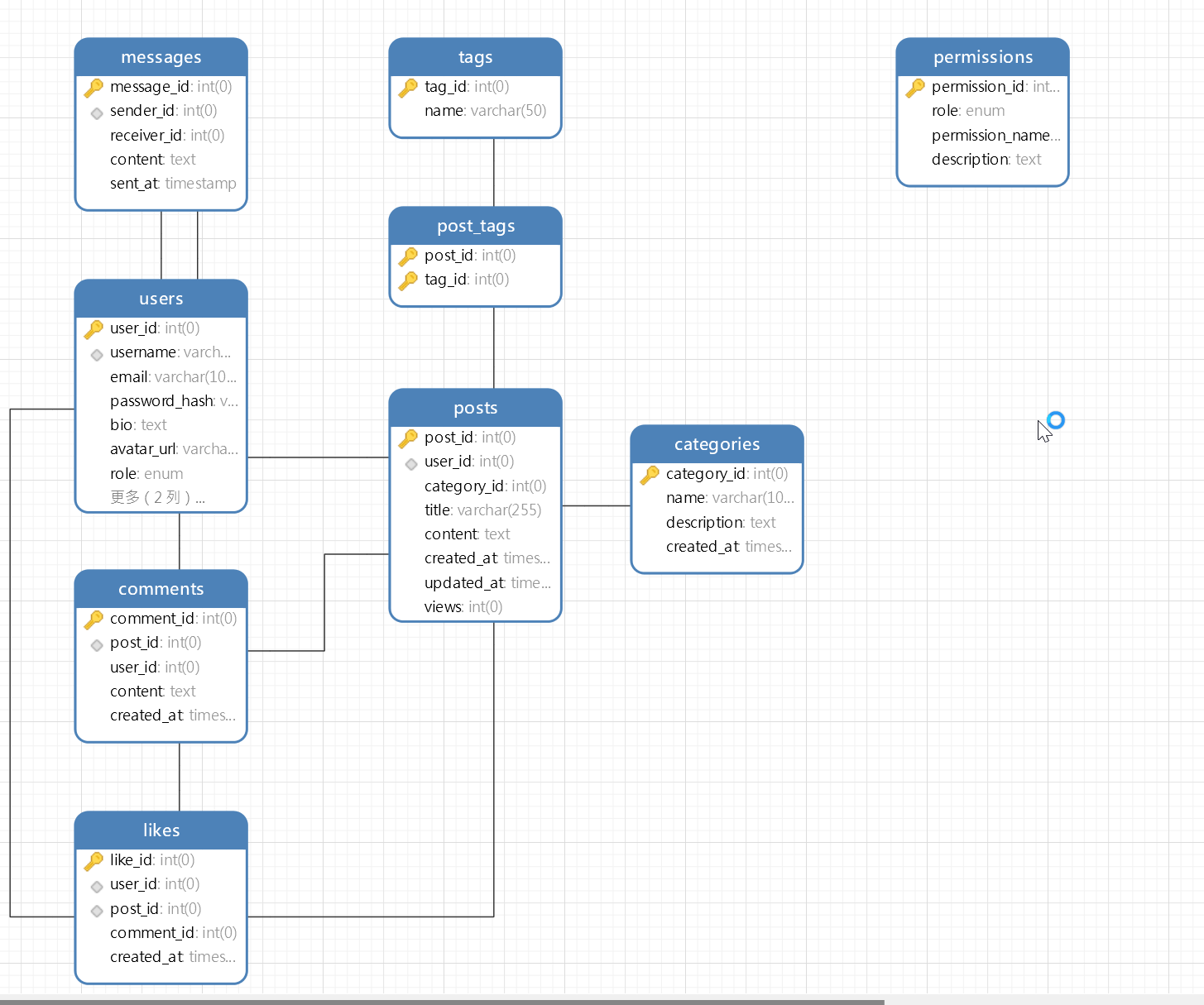
输入参数：用户 ID（或令牌中的用户信息）、帖子 ID、回复内容。输出参数：回复成功状态码。

调用方式：POST 请求，发送到 /api/post/{postId}/reply

# 5系统设计

# 系统架构图（暂缺）

# 数据库设计



（数据库已改振华需要再给er图，不要有鼠标箭头类等er图之外的图标）

### 数据库结构：

#### 用户表 (users)

存储论坛的用户信息。

CREATE TABLE users (

  user\_id INT AUTO\_INCREMENT PRIMARY KEY,

  username VARCHAR(50) NOT NULL UNIQUE,

  email VARCHAR(100) NOT NULL UNIQUE,

  password\_hash VARCHAR(255) NOT NULL,

  bio TEXT,

  avatar\_url VARCHAR(255),

  role ENUM('admin', 'moderator', 'user') DEFAULT 'user',

  created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

  last\_login TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

#### 帖子表 (posts)

存储用户发布的帖子信息。

CREATE TABLE posts (

  post\_id INT AUTO\_INCREMENT PRIMARY KEY,

  user\_id INT NOT NULL,

  category\_id INT,

  title VARCHAR(255) NOT NULL,

  content TEXT NOT NULL,

  created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

  updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

  views INT DEFAULT 0,

  FOREIGN KEY (user\_id) REFERENCES users(user\_id),

  FOREIGN KEY (category\_id) REFERENCES categories(category\_id)

#### 评论表 (comments)

存储用户在帖子中的评论信息。

CREATE TABLE comments (

  comment\_id INT AUTO\_INCREMENT PRIMARY KEY,

  post\_id INT NOT NULL,

  user\_id INT NOT NULL,

  content TEXT NOT NULL,

  created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

  FOREIGN KEY (post\_id) REFERENCES posts(post\_id),

  FOREIGN KEY (user\_id) REFERENCES users(user\_id)

#### 分类表 (categories)

存储论坛的帖子分类信息。

CREATE TABLE categories (

  category\_id INT AUTO\_INCREMENT PRIMARY KEY,

  name VARCHAR(100) NOT NULL UNIQUE,

  description TEXT,

  created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

#### 帖子标签表 (tags)

存储帖子标签信息，用于给帖子添加标签。

CREATE TABLE tags (

  tag\_id INT AUTO\_INCREMENT PRIMARY KEY,

  name VARCHAR(50) NOT NULL UNIQUE

#### 帖子标签关联表 (post\_tags)

用于关联帖子和标签的多对多关系。

CREATE TABLE post\_tags (

  post\_id INT NOT NULL,

  tag\_id INT NOT NULL,

  PRIMARY KEY (post\_id, tag\_id),

  FOREIGN KEY (post\_id) REFERENCES posts(post\_id),

  FOREIGN KEY (tag\_id) REFERENCES tags(tag\_id)

#### 私信表 (messages)

用于用户之间的私信交流。

CREATE TABLE messages (

  message\_id INT AUTO\_INCREMENT PRIMARY KEY,

  sender\_id INT NOT NULL,

  receiver\_id INT NOT NULL,

  content TEXT NOT NULL,

  sent\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

  FOREIGN KEY (sender\_id) REFERENCES users(user\_id),

  FOREIGN KEY (receiver\_id) REFERENCES users(user\_id)

#### 点赞表 (likes)

用于存储帖子和评论的点赞信息。

CREATE TABLE likes (

  like\_id INT AUTO\_INCREMENT PRIMARY KEY,

  user\_id INT NOT NULL,

  post\_id INT DEFAULT NULL,

  comment\_id INT DEFAULT NULL,

  created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

  FOREIGN KEY (user\_id) REFERENCES users(user\_id),

  FOREIGN KEY (post\_id) REFERENCES posts(post\_id),

  FOREIGN KEY (comment\_id) REFERENCES comments(comment\_id)

#### 用户权限表 (permissions)

用于存储用户角色和权限的关联信息（扩展功能）。

CREATE TABLE permissions (

  permission\_id INT AUTO\_INCREMENT PRIMARY KEY,

  role ENUM('admin', 'moderator', 'user') NOT NULL,

  permission\_name VARCHAR(50) NOT NULL,

description TEXT

#### 收藏帖子表 (permissions)

户可以收藏自己感兴趣的帖子，收藏的帖子可以在用户的个人中心中查看和管理。

CREATE TABLE favorites (

    favorite\_id INT AUTO\_INCREMENT PRIMARY KEY,

    user\_id INT NOT NULL,

    post\_id INT NOT NULL,

    created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

    FOREIGN KEY (user\_id) REFERENCES users(user\_id),  -

FOREIGN KEY (post\_id) REFERENCES posts(post\_id)   );

#### 收藏帖子表 (permissions)

该表记录用户在论坛中的积分变动情况。用户通过发帖、评论、下载资源等行为获得积分，并通过该表记录具体的积分来源。

CREATE TABLE points (

    point\_id INT AUTO\_INCREMENT PRIMARY KEY,

    user\_id INT NOT NULL,

    points INT NOT NULL,

    action\_type VARCHAR(100) NOT NULL,

    created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

    FOREIGN KEY (user\_id) REFERENCES users(user\_id)

);

### 数据库设计说明：

1. **users** 表存储用户的基本信息，包括用户名、邮箱、密码和头像等。role字 段用于区分用户、版主和管理员。
2. **posts** 表存储用户发布的帖子，支持分类功能（category\_id），可以按分类查看帖子。
3. **comments** 表存储对帖子的评论，每条评论与一个用户和一个帖子相关联。
4. **categories** 表提供帖子分类功能，管理员可以管理不同的帖子分类。
5. **tags** 和 **post\_tags** 表用于给帖子添加多个标签，增强帖子管理和搜索功能。
6. **messages** 表允许用户之间发送私信，支持基本的用户交互。
7. **likes** 表支持对帖子和评论的点赞功能。
8. **permissions** 表可以扩展角色权限管理，管理员可以管理不同角色的权限。
9. **favorites**表用于记录用户对帖子的收藏，实现用户和帖子之间的多对多关系。
10. **point**s表用于记录用户的积分变动，记录用户在论坛中的各种操作（如发帖、下载资源） 带来的积分变化。

# 6性能测试

# 6.1资源占用评估

# 6.2响应速度评估

# 7安全性措施

# 7.1数据安全

# 7.2访问控制

# 7.3漏洞修复

# 8部署与维护

# 8.1项目部署概述

# 8.2数据库导入

# 8.3开发环境配置

# 8.4 JDK版本调整

# 8.5数据库连接池设置

# 8.6项目启动与测试

# 9网络请求

# 9.1技术概述

# 9.2技术实现

# 10总结