系统设计说明书

一、引言

1. 背景与意义

在福州大学，绝大多数的同学为了通过考试、拿到高分，都有复习历年卷的需求。 然而，获取这些试卷的渠道往往效率低下且成本高昂。 根据我们团队的问卷调查，学生们通常通过以下几种方式获取试卷：

- 私下联系学长学姐获取资源
- 通过互助群购买试卷
- 从fuu陈旧的网上试卷列表中寻找资源

- 零散地从各处网络资源中获取

这些方式存在诸多问题，如依赖人际关系、经济支出、时间成本高等。为了优化这一过程，我们团队开发了《卷卷福》——一个福州大学的历年试卷资源共享平台。该平台旨在通过学生之间的互助共享，简化试卷获取流程，降低获取学习资料的经济和时间成本，促进学习资源的公平分配。

《卷卷福》将整合校内学生手中的历年试卷资源，建立一个便捷的共享平台，帮助学生提升学习效率，推动教育公平和可持续发展。该平台不仅为学生提供便利，还能通过社区互动增强学习氛围，促进教学改进。

2. 目的

系统的设计目标是创建一个用户友好、安全可靠、可扩展的历年试卷共享平台，推动学生之间的互助互利，降低学习成本，并提高资源获取效率。

3. 适用范围

本系统主要面向福州大学的学生、教师及校友，帮助学生获取复习资源，同时为教师提供一个发布教学资源和收集学生反馈的渠道。系统适用范围包括移动端和网页端，支持未来的扩展与升级。

4. 定义与缩略语

- **卷卷福**：指本系统，即福州大学历年试卷资源共享平台。
- **用户**：指使用《卷卷福》系统的所有人员，包括福州大学学生、教师、教职工和校友。
- **试卷资源**：指在《卷卷福》平台上共享的历年考试试卷及相关学习资料。
- **资源共享平台**：指用于存储和提供历年试卷下载的线上系统。
- **UI**：用户界面（User Interface），指系统的前端展示界面，用户通过UI与系统进行交互。
- **API**：应用程序编程接口（Application Programming Interface），系统内部或外部子系统之间的通信接口。
- **FUU**：指福州大学的非正式缩写。
- **移动端**：指智能手机、平板等移动设备上的应用程序。

二、 系统概述

1. 系统设计目标

《卷卷福》系统的设计目标如下：

1. **用户友好性**：界面设计简洁、易用，确保用户能快速上手并使用平台功能。
2. **高效性**：系统响应迅速，能够快速处理用户请求，提供流畅的用户体验。
3. **安全性**：保护用户数据的隐私，确保资源共享过程符合相关的版权法规。
4. **可扩展性**：设计架构能够适应未来用户数量增长以及功能扩展的需求。
5. **可靠性**：系统需稳定运行，尽量避免故障和中断。

2. 系统需求分析

2.1功能需求

> 列出系统需实现的主要功能模块，以及每个功能的详细说明。（表格形式，与下面的子模块不同，这里的主要功能模块范围比较大）

主要功能模块	详细说明
社区	以求助帖和资料分享为核心，促进学生之间的学习资源共享、学术交流和社区互动
资料库	包含系统内所有已上传的学习资料供学生使用，并提供AI交互功能
AI助手	接入大语言模型，用户可与AI助手小福进行对话获得帮助
我的	提供个人信息管理、发布中心、下载与收藏记录、上传资源获取奖励、反馈与帮助等功能

2.2 用户需求

2.2.1 学生用户

- **便捷的资源获取**：学生希望能够快速查找到所需的历年试卷和学习资料，以便节省复习时间，提升学习效率。

- **有效的资源分享**：学生愿意分享自己收藏的试卷资源，帮助其他同学，并希望通过平台的福币奖励系统得到认可和鼓励。
- **社区互动**：学生希望能够通过平台与同学交流学习经验，提出问题，分享见解，并在遇到学习困难时得到他人的帮助。
- **个人成就**：学生用户希望通过平台上的贡献排行榜等功能，查看自己的分享贡献，并以此获得成就感和认可。

2.2.2 教师用户

- **教学资源发布**：教师希望通过平台发布课程相关的教学资料和考试试卷，供学生参考和使用，帮助学生更好地复习和学习。
- **学生互动**：教师希望通过平台与学生建立互动关系，回答学生的提问，提供学习指导，从而提升教学效果。
- **教学反馈**：教师希望通过学生的资源使用情况和提出的问题，了解学生在学习过程中遇到的困难，从而改进自己的教学方式或调整课程内容。

2.2.3 管理员用户

- **内容管理**：管理员需要负责审核用户上传的所有资源，确保平台上共享的资料符合质量标准，并且不涉及版权问题，以保证平台的合法性和资源的高质量。
- **用户管理**：管理员需要管理用户账户，包括处理用户注册、分配权限、处理违规行为等，保障平台的安全与秩序。
- **系统维护**：管理员还需负责平台的日常维护，包括监控系统运行状况、故障排查和性能优化，确保平台的稳定性和可用性。
- **数据分析**：通过对用户行为的数据分析，管理员可以了解用户的使用习惯，从而对平台功能进行优化，提升用户体验。

3. 系统的适用范围

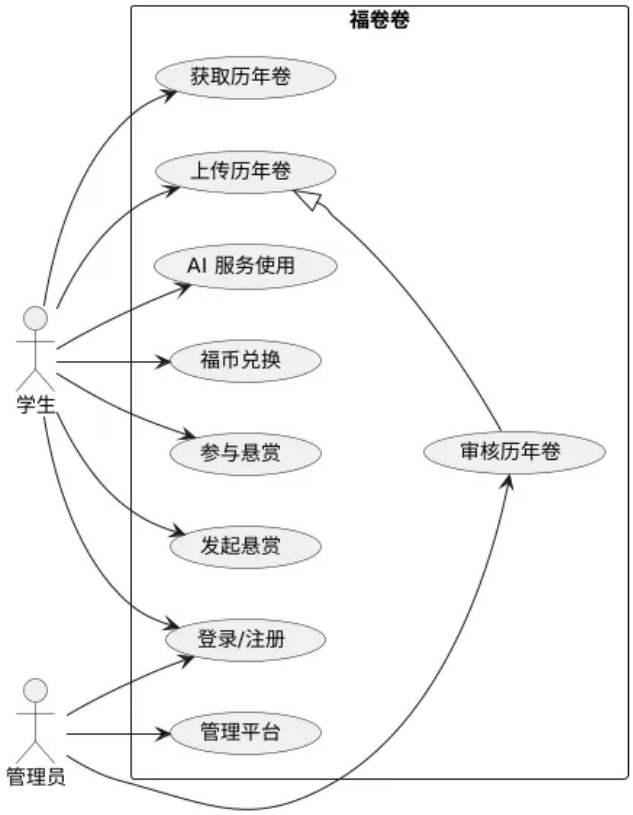
1. **福州大学学生**：系统主要面向在校学生，提供试卷和学习资料的共享服务。
2. **教师和教职工**：教师可以在平台上发布试卷和教学资料，并获取学生反馈以优化教学内容。
3. **校友**：毕业校友也可以通过平台分享他们的学习经验和资源，为在校学生提供帮助。
4. **访问限制**：为了确保资源的合理使用和版权保护，平台可能会对非福州大学用户设置访问限制。

三、系统架构设计

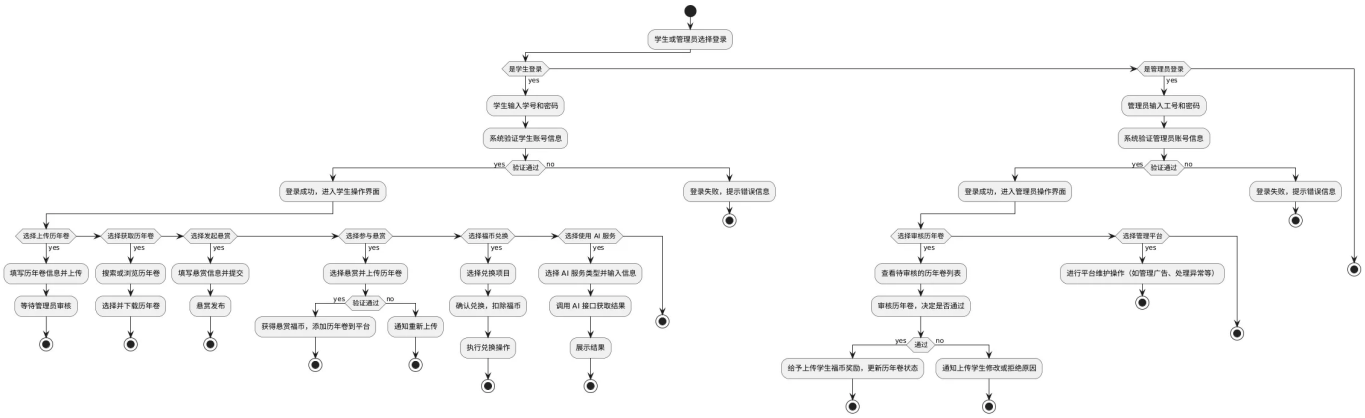
1. 系统架构图（UML图）

> 绘制系统的整体架构图，描述各模块之间的关系。

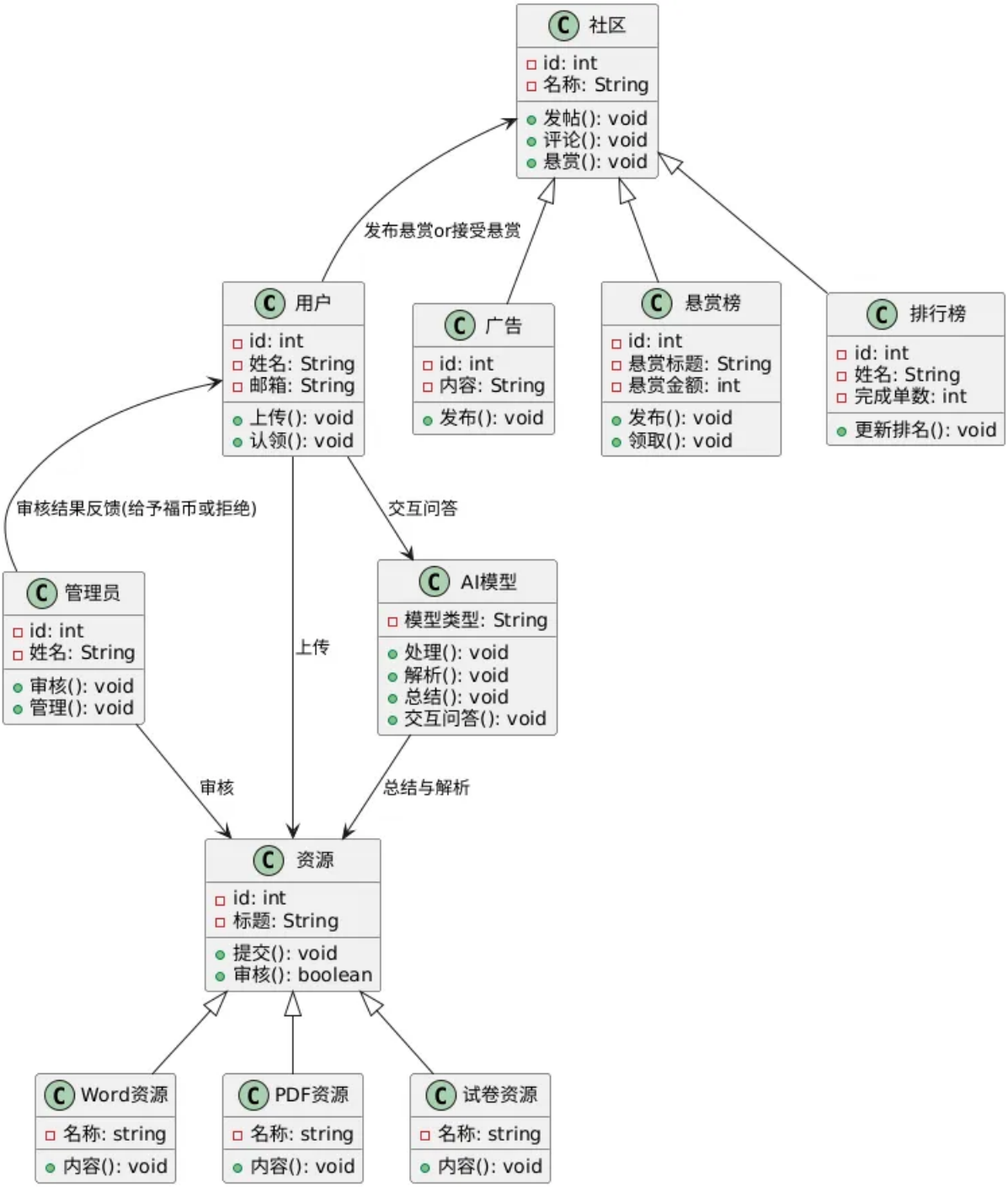
1.1. UML用例图



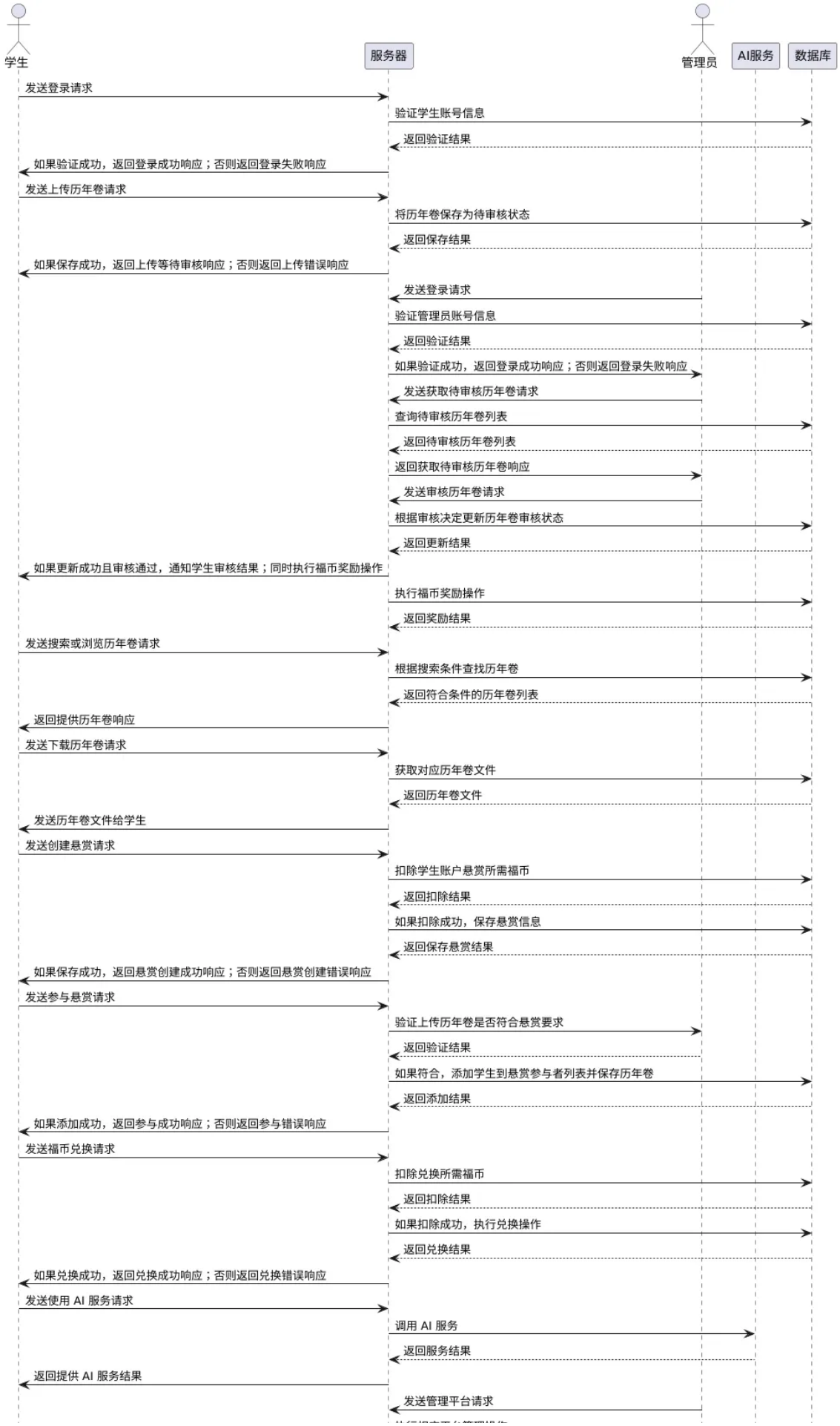
1.2. UML活动图

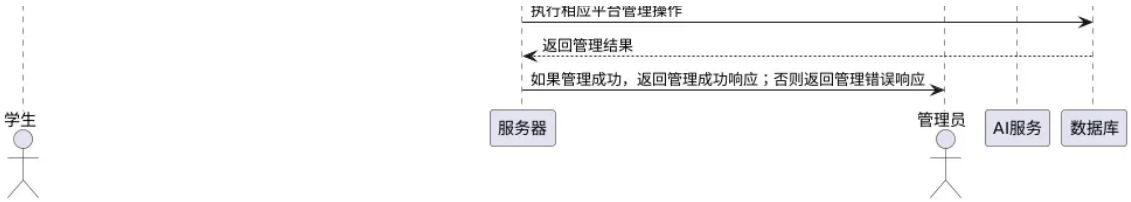

学生或管理员选择登录
是学生登录
是管理员登录
yes
学生输入学号和密码
系统验证学生账号信息
验证通过
yes
no
登录成功，进入学生操作界面
登录失败，提示错误信息
选择上传历年卷
选择获取历年卷
选择发起悬赏
选择参与悬赏
选择福币兑换
选择使用 AI 服务
填写历年卷信息并上传
等待管理员审核
搜索或浏览历年卷
选择并下载历年卷
填写悬赏信息并提交
悬赏发布
选择悬赏并上传历年卷
获得悬赏福币，添加历年卷到平台
通知重新上传
选择兑换项目
确认兑换，扣除福币
执行兑换操作
选择 AI 服务类型并输入信息
调用 AI 接口获取结果
展示结果
管理员输入工号和密码
系统验证管理员账号信息
登录成功，进入管理员操作界面
登录失败，提示错误信息
选择审核历年卷
选择管理平台
查看待审核的历年卷列表
审核历年卷，决定是否通过
通过
给予上传学生福币奖励，更新历年卷状态
通知上传学生修改或拒绝原因
进行平台维护操作（如管理广告、处理异常等）

1.3. UML类图

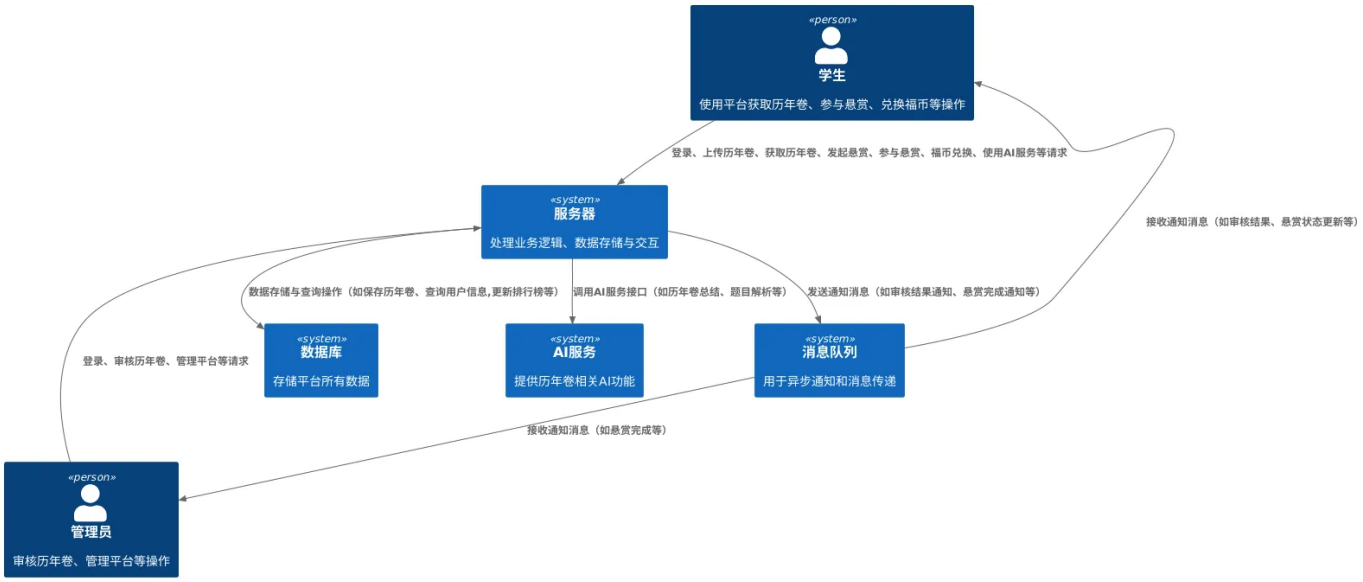

社区
id: int
名称: String
发帖(): void
评论(): void
悬赏(): void
发布悬赏or接受悬赏
用户
id: int
姓名: String
邮箱: String
上传(): void
认领(): void
广告
id: int
内容: String
发布(): void
悬赏榜
id: int
悬赏标题: String
悬赏金额: int
发布(): void
领取(): void
排行榜
id: int
姓名: String
完成单数: int
更新排名(): void
审核结果反馈(给予福币或拒绝)
交互问答
管理员
id: int
姓名: String
审核(): void
管理(): void
上传
AI模型
模型类型: String
处理(): void
解析(): void
总结(): void
交互问答(): void
审核
总结与解析
资源
id: int
标题: String
提交(): void
审核(): boolean
Word资源
名称: string
内容(): void
PDF资源
名称: string
内容(): void
试卷资源
名称: string
内容(): void

1.4. UML时序图


学生
服务器
管理员
AI服务
数据库
发送登录请求
验证学生账号信息
返回验证结果
如果验证成功，返回登录成功响应；否则返回登录失败响应
发送上传历年卷请求
将历年卷保存为待审核状态
返回保存结果
如果保存成功，返回上传等待审核响应；否则返回上传错误响应
发送登录请求
验证管理员账号信息
返回验证结果
如果验证成功，返回登录成功响应；否则返回登录失败响应
发送获取待审核历年卷请求
查询待审核历年卷列表
返回待审核历年卷列表
返回获取待审核历年卷响应
发送审核历年卷请求
根据审核决定更新历年卷审核状态
返回更新结果
如果更新成功且审核通过，通知学生审核结果；同时执行福币奖励操作
执行福币奖励操作
返回奖励结果
发送搜索或浏览历年卷请求
根据搜索条件查找历年卷
返回符合条件的历年卷列表
返回提供历年卷响应
发送下载历年卷请求
获取对应历年卷文件
返回历年卷文件
发送历年卷文件给学生
发送创建悬赏请求
扣除学生账户悬赏所需福币
返回扣除结果
如果扣除成功，保存悬赏信息
返回保存悬赏结果
如果保存成功，返回悬赏创建成功响应；否则返回悬赏创建错误响应
发送参与悬赏请求
验证上传历年卷是否符合悬赏要求
返回验证结果
如果符合，添加学生到悬赏参与者列表并保存历年卷
返回添加结果
如果添加成功，返回参与成功响应；否则返回参与错误响应
发送福币兑换请求
扣除兑换所需福币
返回扣除结果
如果扣除成功，执行兑换操作
返回兑换结果
如果兑换成功，返回兑换成功响应；否则返回兑换错误响应
发送使用 AI 服务请求
调用 AI 服务
返回服务结果
返回提供 AI 服务结果
发送管理平台请求



1.5. UML协作图



2. 技术选型

> 说明系统使用的开发语言、框架、数据库、第三方库

技术栈：

- vite
- vue3
- ts
- uniCloud
- wotDesignUni
- sp-editor

基于uniapp框架，配合云开发平台：https://unicloud.dcloud.net.cn/，实现前后端均部署到云服务器上。

部署

- Serverless 服务
- 云原生容器平台（支付宝云）

集成插件

- sp-editor 基于官方的富文本编辑器editor组件改良版
- wotDesignUni 基于vue3+Typescript构建的精美ui库
- mp-html 富文本解析器

使用语言：

- **前端框架主力**：Vue.js 是这个项目的核心前端框架，负责管理整个应用的用户界面和交互逻辑。Vue的组件化设计能够帮助模块化开发，提升代码的复用性。
- **编程语言主力**：JavaScript 和 TypeScript 搭配使用，JavaScript 负责业务逻辑，TypeScript 提供类型检查和提高代码质量。二者的结合能够帮助团队在开发时更好地控制代码的健壮性和可扩展性。
- **样式管理**：SCSS 是主要的样式工具，通过预处理器功能能够简化复杂样式的管理，而少量的CSS则可能用于一些基础样式的处理。
- **功能实现**：前端功能模块灵活运用**uniapp下所设计好的插件****，实现软件功能模块快速搭建实现。
- **数据库管理**：使用uniapp下的云服务器，后端管理。

四、详细设计

1. 模块设计

> 列出系统的各个子模块，提供模块的功能说明。（表格形式，与上面的主要功能模块不同，这个更细）

功能模块	子模块	说明
社区	浏览求助帖	用户可以查看其他用户发布的求助帖，并提供自己的历年卷资源以获得福币奖励。
	发布求助帖	用户可以使用福币作为悬赏，发布求助帖以获取其他用户分享的历年卷资源。

	排行榜	根据用户完成的求助单数量进行排名，激励用户积极参与资源共享。
	问题咨询	用户可以提出学习相关的问题，其他用户可以留言提供帮助和建议。
	资料分享	用户可以上传并分享各种学习资料，同时获得福币奖励。
资源库	学科分类	资源库中包含了所有已被用户上传的历年卷和教材资源，并根据学科进行分类。
	AI功能 – 题目总结	用户打开历年卷进行学习时，AI可以总结历年卷的题目，包括包含的知识点和整体难度。
	AI功能 – 题目解析	AI可以对单个题目进行解析，包括题目包含的知识点、解题思路和方法。
	AI功能 – 举一反三	AI可以对单个题目进行举一反三，给出一些相同知识点的题目供用户强化训练。
	AI功能 – 提问获得指点	遇到不懂的地方，用户可以向AI提问获得指点。
AI助手	AI对话	用户可以与接入大语言模型的AI助手小福进行对话，获得帮助。
我的	个人信息	用户可以查看和修改自己的个人信息。

	发布中心	用户可以查看自己已发布的资源。
	我的下载	用户可以查看自己下载的资源。
	我的收藏	用户可以查看自己收藏的资源。
	上传文件	用户可以上传学习资源并获得相应的福币奖励。
	反馈与帮助	用户可以对app的使用感受进行反馈。
	夜间模式	用户可以打开夜间模式以减少眼睛疲劳。

2. 接口设计

> 定义模块之间的接口和外部接口，包括API接口、接口格式、传输协议及安全措施。

2.1. 总体设计思路

- **传输协议**：使用 HTTPS 确保所有数据在传输过程中加密，以保障数据的安全性。
- **数据格式**：采用 JSON 作为数据交换格式，简洁且易于解析。
- **身份验证**：使用 OAuth2.0 或 JWT 令牌进行用户身份认证，每个用户的敏感操作需要验证授权，以防止未授权访问。
- **防护措施**：加入 IP 限制、验证码、数据加密及访问控制，以增强系统安全性。

2.2. 模块接口设计

2.2.1. 社区模块

①浏览求助帖

- **接口：** `GET /community/help-posts`
- **请求参数：**
  - `page` (int): 页码，默认 1
  - `size` (int): 每页显示的求助帖数，默认 10

- **响应：**

```json
{
  "posts": [
    {
      "id": "string",
      "title": "string",
      "author": "string",
      "reward": "int",
      "status": "open/closed",
      "comments": "int",
      "time_posted": "timestamp"
    }
  ],
  "total_pages": "int"
}
```

②发布求助帖

- **接口**：`POST /community/help-posts`
- **请求参数**：

```
{
  "title": "string",
  "description": "string",
  "reward": "int",
  "tags": ["string"]
}
```

- **响应:**

```
{
  "status": "success",
  "post_id": "string"
}
```

③排行榜

- **接口**：`GET /community/leaderboard`

- **请求参数**：无
- **响应**：

```json
{
  "leaderboard": [
    {
      "user_id": "string",
      "username": "string",
      "completed_requests": "int"
    }
  ]
}
```

④问题咨询

- **接口**：`POST /community/consultation`
- **请求参数**：

```json
{
  "question": "string",
  "tags": ["string"]
}
```

- **响应**：

```json
{
  "status": "success",
  "question_id": "string"
}
```

⑤资料分享

- **接口**：`POST /community/share-resource`
- **请求参数**：

```json
{
  "resource_name": "string",
  "description": "string",
  "file_url": "string",
  "tags": ["string"]
}
```

- **响应**：

```json
{
  "status": "success",
  "resource_id": "string"
}
```

2.2.2. 资源库模块

①获取资源列表

- **接口**：`GET /resources`
- **请求参数**：
  - `subject` (string): 学科
  - `page` (int): 页码
- **响应**：

```json
{
  "resources": [
    {
      "id": "string",
      "name": "string",
      "subject": "string",
      "description": "string"
    }
  ],
  "total_pages": "int"
}
```

②AI 功能 – 总结试卷

- **接口：** `POST /resources/ai/summary`
- **请求参数：**

```json
{
  "resource_id": "string"
}
```

- **响应：**

```json
{
  "summary": {
    "knowledge_points": ["string"],
    "difficulty": "int"
  }
}
```

③AI 功能 – 题目解析

- **接口：** `POST /resources/ai/analysis`
- **请求参数：**

```json
{
  "resource_id": "string",
  "question_id": "string"
}
```

- **响应：**

```json
{
  "analysis": {
    "knowledge_points": ["string"],
    "solution": "string",
    "method": "string"
  }
}
```

2.2.3. AI助手模块

①对话

- **接口**：`POST /ai-assistant/conversation`
- **请求参数及响应**：

```json
{
  "message": "string",
  "context_id": "string"
}
```

2.2.4. 我的模块

①个人信息查询

- **接口**：`GET /user/profile`
- **请求参数**：无
- **响应**：

```json
{
  "user_id": "string",
  "username": "string",
  "email": "string",
  "fubi_balance": "int"
}
```

②发布中心

- **接口**：`GET /user/my-posts`
- **请求参数**：
- `page` (int): 页码
- **响应**：

```json
{
  "posts": [
    {
      "id": "string",
      "title": "string",
      "status": "open/closed",
      "time_posted": "timestamp"
    }
  ],
  "total_pages": "int"
}
```

③上传文件

- **接口**：`POST /user/upload`
- **请求参数**：

```json
{
  "file": "binary",
  "description": "string",
  "tags": ["string"]
}
```

- **响应**：

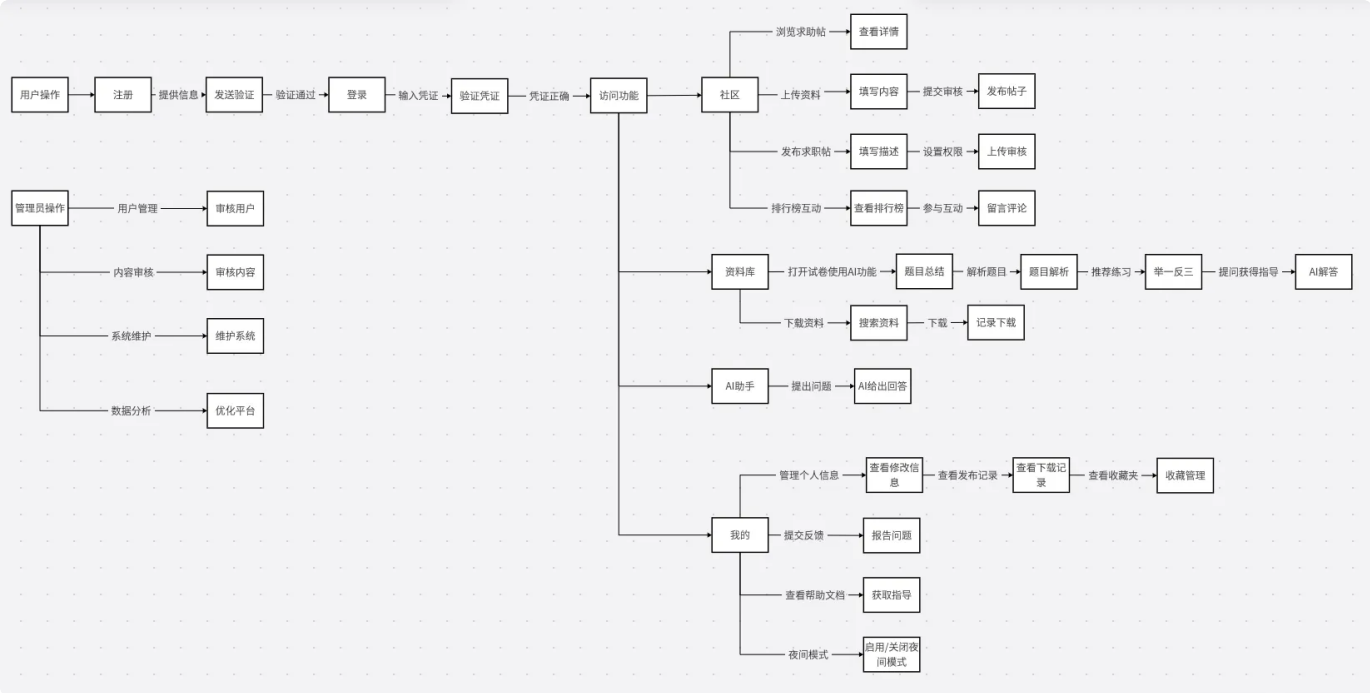
```json
{
  "status": "success",
  "resource_id": "string",
  "reward_fubi": "int"
}
```

2.3. 安全措施

- **数据加密**：所有接口通过 HTTPS 加密。
- **身份认证**：所有需要授权的接口均需提供 OAuth 2.0 / JWT 令牌。
- **访问控制**：细粒度权限控制，敏感接口设有双因子认证。
- **输入验证**：所有输入参数进行类型和内容验证，防止 SQL 注入、XSS 等攻击。

3. 业务流程

> 详细描述系统的主要业务流程，通常配合流程图来说明。



3.1. 用户注册与登录

- **注册流程**：
  - 用户提供用户名和密码。
  - 系统验证信息的完整性。
  - 系统发送验证邮件或短信。
  - 用户确认身份，注册完成。
- **登录流程**：
  - 用户输入用户名和密码。
  - 系统验证凭证是否匹配。
  - 若验证通过，用户可以访问平台功能；若失败，提示重新输入或找回密码。
- **流程图要素**：
  - 开始节点 → 用户输入信息 → 系统验证 → 验证成功/失败 → 结束节点

3.2. 浏览与发布求助帖

- **浏览求助帖**：

- 用户进入求助帖浏览页面。
- 系统显示所有公开的求助帖列表。
- 用户点击查看帖子的详细信息。

- **发布求助帖**：
  - 用户点击发布新帖。
  - 填写求助内容、悬赏福币等信息。
  - 提交后系统自动审核，审核通过后帖子发布。
- **流程图要素**：
  - 开始 → 选择发布求助帖 → 填写信息 → 系统审核 → 发布成功/失败

3.3. 资料分享与下载

- **上传资料**：
  - 用户进入资料上传页面。
  - 提供资料的详细信息和权限设置。
  - 上传资料后，系统审核并加入资源库。
- **下载资料**：
  - 用户浏览或搜索资源库中的资料。
  - 选择下载资料，资料将保存到用户的下载记录中。
- **流程图要素**：
  - 上传：开始 → 填写资料信息 → 提交 → 系统审核 → 上传成功/失败
  - 下载：开始 → 浏览/搜索资料 → 下载 → 添加至记录

3.4. AI功能的使用

- **题目总结**：
  - 用户打开历年试卷。
  - AI助手分析卷子中的题目，给出知识点总结和难度评估。
- **题目解析**：
  - 用户选定单个题目。
  - AI助手提供解题步骤、相关知识点解析。
- **举一反三**：
  - AI助手根据题目推荐相似练习题，用户进行进一步练习。

- **流程图要素**：
  - AI功能整体流程：选择题目 → AI分析 → 总结/解析/举一反三 → 用户反馈结果

3.5. 排行榜与社区互动

- **查看排行榜**：
  - 用户点击排行榜查看活跃用户的求助单完成情况。
- **社区互动**：
  - 用户可在求助帖下进行留言、评论或点赞。
- **流程图要素**：
  - 查看排行榜：开始 → 点击查看 → 排名展示
  - 社区互动：开始 → 选择帖子 → 留言/点赞 → 系统更新

3.6. 个人信息管理

- 用户进入个人中心，查看或修改个人信息，或查看发布与下载记录。

3.7. 反馈与帮助

- 用户可以通过提交反馈或问题，系统管理员会收到并处理。

3.8. 夜间模式

- 用户可以在系统设置中开启或关闭夜间模式。

3.9. 管理员操作

- **用户管理**：
  - 管理员审核新用户、管理用户权限。
- **内容审核**：
  - 管理员审核上传的资料和求助帖，确保合规。
- **流程图要素**：
  - 审核流程：管理员登录 → 审核内容/用户 → 审核结果

五、数据设计

1. 数据结构设计

> 描述系统的数据结构。（表格形式，参考[软工]概要设计说明书(GB8567-88)-CSDN博客）

1.1. 数据库表设计

1.1.1. 用户表（`users`）

- 存储用户的基本信息及账户信息，包括福币余额、角色权限等。

字段名	类型	描述
user_id	UUID (PK)	用户唯一标识
username	VARCHAR	用户名
email	VARCHAR	邮箱
password_hash	VARCHAR	加密的密码
fubi_balance	INT	用户福币余额
role	ENUM	用户角色（如普通、管理员等）
created_at	TIMESTAMP	注册时间

1.1.2. 帖子表（`help_posts`）

- 社区中的求助帖，用户可以发布求助需求，包含悬赏的福币金额。

字段名	类型	描述
post_id	UUID (PK)	帖子唯一标识
user_id	UUID (PK)	发布者用户 ID
title	VARCHAR	帖子标题
description	TEXT	帖子内容

reward	INT	悬赏福币数
status	ENUM	帖子状态（open/closed）
created_at	TIMESTAMP	帖子发布时间

1.1.3. 评论表（`comments`）

- 用户可以在求助帖下方留言或回答。

字段名	类型	描述
comment_id	UUID (PK)	评论唯一标识
post_id	UUID (FK)	所属求助帖 ID
user_id	UUID (FK)	评论者用户 ID
content	TEXT	评论内容
created_at	TIMESTAMP	评论发布时间

1.1.4. 排行榜表（`leaderboard`）

- 存储完成求助帖数量，用于生成排行榜。

字段名	类型	描述
user_id	UUID (PK)	用户唯一标识
completed_requests	INT	用户完成的求助帖数量

1.1.5. 资源库表（`resources`）

- 包含平台上所有学习资料和历年卷信息，支持按照学科分类。

字段名	类型	描述
resource_id	UUID (PK)	资源唯一标识
user_id	UUID (FK)	上传者用户 ID
subject	VARCHAR	学科分类

resource_name	VARCHAR	资源名称
description	TEXT	资源描述
file_url	VARCHAR	文件存储 URL
uploaded_at	TIMESTAMP	上传时间

1.1.6. 题目表（`questions`）

- 存储历年卷中的具体题目，用于 AI 功能的分析和强化训练。

字段名	类型	描述
question_id	UUID (PK)	题目唯一标识
resource_id	UUID (FK)	所属资源 ID
content	TEXT	题目内容
knowledge_points	JSON	题目知识点列表
difficulty	INT	题目难度

1.1.7. 收藏表（`favorites`）

- 用户收藏的资源记录。

字段名	类型	描述
favorite_id	UUID (PK)	收藏记录 ID
user_id	UUID (FK)	用户 ID
resource_id	UUID (FK)	收藏资源 ID
added_at	TIMESTAMP	收藏时间

1.1.8. AI 问答表（`ai_interactions`）

- 用户向 AI 进行提问和互动时的记录，用于帮助学生解答疑问。

字段名	类型	描述

interaction_id	UUID (PK)	问答唯一标识
user_id	UUID (FK)	用户 ID
question_id	UUID (FK)	题目 ID
question	TEXT	用户提问内容
answer	TEXT	AI 返回的答案
asked_at	TIMESTAMP	提问时间

1.1.9. 下载表（`downloads`）

- 用户下载的资源记录，记录下载的资源及下载时间。
- 用户收藏的资源记录。

字段名	类型	描述
download_id	UUID (PK)	下载记录 ID
user_id	UUID (FK)	用户 ID
resource_id	UUID (FK)	下载的资源 ID
downloaded_at	TIMESTAMP	下载时间

1.2. 数据结构逻辑关系

- **`users`** 与 **`help_posts`**：一对多关系，用户可以发布多个求助帖。
- **`help_posts`** 与 **`comments`**：一对多关系，每个求助帖可以有多个评论。
- **`users`** 与 **`leaderboard`**：一对一关系，每个用户都有一个对应的排行榜记录。
- **`users`** 与 **`resources`**：一对多关系，用户可以上传多个资源。
- **`resources`** 与 **`questions`**：一对多关系，每个资源可以包含多个题目。
- **`users`** 与 **`ai_interactions`**：一对多关系，用户可以进行多次 AI 提问互动。
- **`users`** 与 **`favorites`**：一对多关系，用户可以收藏多个资源。
- **`users`** 与 **`downloads`**：一对多关系，用户可以下载多个资源。

2. 数据存储设计

> 说明数据的存储方式、存储策略（参考同上）

2.1. 数据存储方式

2.1.1. 关系型数据库（MySQL / PostgreSQL）

- **适用范围**：用户信息、帖子、评论、资源、下载记录、反馈等结构化数据。
- **优点**：支持复杂查询、事务处理、关系约束，适合高一致性需求的数据存储。
- **数据表**：主要数据表如 `users`、`help_posts`、`comments`、`resources` 等都存储在关系型数据库中，以保证数据的完整性和一致性。

2.1.2. NoSQL 数据库（MongoDB / DynamoDB）

- **适用范围**：用户与 AI 的互动数据、用户的历史操作记录。
- **优点**：灵活的文档结构适合存储不固定格式或不规则的数据，支持快速查询和高并发访问。
- **数据表**：`ai_interactions` 和部分用户行为日志可以存储在 NoSQL 数据库中，便于快速响应和灵活扩展。

2.1.3. 对象存储（AWS S3 / 阿里云 OSS）

- **适用范围**：用户上传的资源文件（如历年卷和教材 PDF）。
- **优点**：支持大文件存储、访问频率灵活、成本较低，且可以根据需求进行动态扩展。
- **数据表**：`resources` 表中的文件路径会存储为对象存储的链接，数据库中只存储文件 URL 信息，而文件本身存储在对象存储服务上。

2.1.4. 缓存数据库（Redis）

- **适用范围**：热门资源、社区排行榜、热门求助帖等数据。
- **优点**：高速读取，支持数据过期和自动清除，适合高频查询场景。
- **数据表**：排行榜（如 `leaderboard`）、热门求助帖等将被临时存入 Redis，提升访问速度并减轻数据库压力。

2.2. 数据存储策略

2.2.1. 分库分表策略

- **策略描述**：对于用户量较大、访问频率较高的表（如 `users` 和 `resources` 表），可以采用分库分表的方式，通过用户 ID 或资源 ID 进行分区，减少单个数据库的负载并提升读写性能。
- **适用表**： `users` 、 `resources` 表。

2.2.2. 数据分级存储

- **策略描述**：根据数据的重要性、访问频率、数据生命周期进行分级存储。频繁访问的数据（如排行榜、热门求助帖等）存储在 Redis 等缓存数据库中，冷数据或历史数据（如下载记录、AI 问答记录等）可以定期归档到低成本的对象存储或文件系统中。
- **适用表**： `leaderboard` 、 `downloads` 、 `ai_interactions` 等数据表。

2.2.3. 备份与灾难恢复

- **策略描述**：每天定时对关系型数据库中的关键数据（如用户表、资源表等）进行备份，备份文件存储在不同的数据中心（或多区域云存储）以提升灾难恢复能力。同时，对象存储服务配置多副本，确保文件的高可用性。
- **备份频率**：用户数据、求助帖、资源库数据每天备份，文件存储采用自动多副本存储。

2.2.4. 数据一致性策略

- **策略描述**：采用事务处理（ACID）确保关键数据（如福币奖励、用户信息更新等）的数据一致性。使用主从复制和多节点一致性协议（如 Raft 协议）保证数据同步，防止因服务中断导致数据不一致问题。
- **适用场景**：悬赏福币支付、资源上传和下载等重要交易场景。

2.2.5. 安全策略

- **加密存储**：用户敏感信息（如密码）采用加密存储，文件 URL 中的访问 Token 进行加密。
- **权限控制**：使用 RBAC（基于角色的访问控制）模型管理数据访问权限，不同角色（如管理员、普通用户）只能访问和操作特定数据。
- **审计日志**：记录用户对数据的访问和操作日志（如用户上传、下载、评论等），确保数据操作可追溯。

2.3. 数据生命周期管理

- **短期数据**：如用户行为日志、排行榜数据，保留时间较短（1–3 个月），定期删除或转存到低频存储。

- **长期数据**：如用户个人信息、资源库、求助帖及其评论等长期数据，长期保留并进行周期性备份。
- **归档数据**：对于较老的 AI 问答记录、下载记录等低访问数据，归档到对象存储或冷存储服务中，节省存储成本。

六、安全设计

1. 用户权限管理

> 定义系统的权限管理策略，包括不同用户角色的权限分配。

1.1学生用户权限

- **资源浏览**：可以浏览所有公开的学习资源和求助帖。
- **资源下载**：可以下载公开的和有权限的资源。
- **资源上传**：可以上传学习资料并获得福币奖励。
- **发布求助帖**：可以发布求助帖以获取所需的资源。
- **参与互动**：可以在社区中留言、评论。
- **个人资料管理**：可以查看和修改个人信息。
- **AI助手使用**：可以与AI助手小福进行对话，获取学习帮助和指导。
- **AI功能使用**：可以使用AI功能对历年卷进行题目总结、解析、举一反三和提问获得指点。
- **我的资源管理**：可以查看自己发布的资源、下载记录和收藏夹。
- **反馈提交**：可以提交使用反馈。
- **夜间模式**：可以启用和关闭夜间模式。

1.2管理员用户权限

- **用户管理**：可以管理所有用户账户，包括审核新用户、权限分配等。
- **内容审核**：可以审核用户上传的资源，确保内容的质量和版权合规。
- **系统维护**：可以执行平台的日常维护和故障排除。
- **数据分析**：可以访问和分析用户行为数据，以优化平台功能和用户体验。
- **反馈处理**：可以处理用户反馈，提供技术支持和解决方案。
- **权限分配**：可以分配和调整用户权限。

- **资源管理**：可以管理所有资源，包括删除违规内容。

2. 系统安全防护

> 描述身份认证安全防护措施。

2.1. 强密码策略

- **密码复杂度**：要求用户设置包含大小写字母、数字和特殊字符的强密码。
- **密码长度**：设定最小密码长度，例如至少8个字符。
- **密码历史**：防止用户重复使用最近使用过的密码。

2.2. 多因素认证（MFA）

- **第二认证因素**：除了密码之外，要求用户提供第二种认证方式，如短信验证码、邮箱验证码、移动应用生成的动态口令或生物识别技术（如指纹、面部识别）。

2.3. 防止暴力破解

- **登录尝试限制**：限制连续登录失败的次数，例如5次失败后锁定账户一段时间。
- **验证码**：在登录表单中加入验证码，防止自动化脚本攻击。

2.4. 会话管理

- **安全会话**：使用安全的会话管理机制，如安全的cookie属性（HttpOnly, Secure），防止会话劫持。
- **会话超时**：设置会话超时时间，例如15分钟无活动后自动登出。
- **登出功能**：提供明确的登出选项，确保用户能够安全地结束会话。

2.5. 加密传输

- **SSL/TLS**：使用SSL/TLS加密所有传输中的数据，包括登录凭证。
- **HSTS**：实施HTTP严格传输安全（HSTS）策略，强制所有连接使用HTTPS。

2.6. 账户恢复

- **安全恢复流程**：提供安全的账户恢复选项，如通过电子邮件或手机短信重置密码。
- **身份验证问题**：避免使用容易猜测的安全问题作为账户恢复手段。

2.7. 监控与审计

- **登录活动监控**：监控登录活动，检测异常登录行为。
- **审计日志**：记录所有身份验证相关的操作日志，以便于追踪和调查。

2.8. 防止跨站请求伪造（CSRF）

- **CSRF令牌**：在表单中加入CSRF令牌，确保请求的合法性。
- **SameSite属性**：为cookie设置SameSite属性，防止CSRF攻击。

2.9. 防止跨站脚本攻击（XSS）

- **输入验证**：对所有用户输入进行严格的验证和清理。
- **输出编码**：对输出数据进行适当的编码，防止脚本注入。

七、测试计划

1. 测试范围

1.1. 功能测试（Functional Testing）

  - **核心功能**：验证所有核心功能如浏览求助帖、发布求助帖、排行榜、问题咨询、资料分享等是否按预期工作。
  - **用户交互**：测试用户注册、登录、个人信息管理、资源上传与下载、收藏等交互流程。
  - **AI功能**：测试AI助手的功能，包括对历年卷的题目总结、解析、举一反三以及AI提问功能。
  - **社区互动**：确保用户之间的互动如评论、点赞、分享等功能的正确性。

1.2. 性能测试（Performance Testing）

- **负载测试**：评估系统在高并发用户访问时的性能表现。
- **压力测试**：测试系统在高负载下的稳定性和恢复能力。
- **响应时间**：测量系统对用户请求的响应时间，确保在可接受范围内。

1.3. 安全测试（Security Testing）

- **数据保护**：确保用户数据的安全存储和传输，包括加密措施。
- **身份验证**：验证用户身份验证机制的有效性，防止未授权访问。
- **漏洞扫描**：进行安全漏洞扫描，识别并修复潜在的安全风险。

1.4. 兼容性测试（Compatibility Testing）

- **浏览器兼容性**：测试系统在不同浏览器（如Chrome, Firefox, Safari, Edge）上的表现。
- **设备兼容性**：确保系统在不同设备（桌面、平板、手机）上的兼容性和响应式设计。
- **操作系统兼容性**：验证系统在不同操作系统（如Windows, macOS, Linux, iOS, Android）上的运行情况。

1.5. 用户体验测试（User Experience Testing）

- **界面设计**：评估用户界面的直观性和易用性。
- **导航流程**：测试用户导航的流畅性和逻辑性。
- **反馈机制**：确保用户反馈机制的有效性，便于用户报告问题和提供建议。

1.6. 可靠性测试（Reliability Testing）

- **错误处理**：测试系统在遇到错误或异常情况时的处理能力。
- **数据备份与恢复**：验证数据备份和恢复机制的有效性。
- **系统稳定性**：长时间运行测试，监控系统的稳定性和性能。

1.7. 可用性测试（Usability Testing）

- **易用性**：评估系统对不同用户群体的易用性，包括新手用户和高级用户。
- **学习曲线**：评估用户学习使用系统所需的时间和努力。
- **用户满意度**：通过问卷调查或访谈收集用户对系统的满意度反馈。

1.8. 文档和帮助测试（Documentation and Help Testing）

- **用户文档**：验证用户手册、FAQ和帮助文档的准确性和完整性。
- **在线帮助**：测试在线帮助系统的有效性和易用性。

2. 测试方法

2.1. 单元测试（Unit Testing）

- **目的**：验证每个最小的软件组件或模块是否按预期工作。
- **方法**：编写测试用例来测试每个函数或方法，确保它们在各种输入下都能正确运行。

2.2. 集成测试（Integration Testing）

- **目的**：确保不同模块或服务之间的接口和交互正常工作。
- **方法**：将多个单元组合在一起进行测试，检查它们之间的数据流和控制流。

2.3. 系统测试（System Testing）

- **目的**：验证整个系统的完整性和功能性。
- **方法**：在模拟的生产环境中运行完整的系统，测试所有端到端的功能。

2.4. 性能测试（Performance Testing）

- **目的**：评估系统在不同负载条件下的性能。
- **方法**：使用负载测试工具模拟大量用户同时访问系统，测试响应时间和系统稳定性。

2.5. 安全测试（Security Testing）

- **目的**：识别系统中的安全漏洞和弱点。
- **方法**：进行渗透测试，模拟攻击者的行为，尝试突破系统的安全措施。

2.6. 用户接受测试（User Acceptance Testing, UAT）

- **目的**：确保系统满足最终用户的需求和期望。

- **方法**：邀请实际用户参与测试，收集他们的反馈并根据反馈进行调整。

2.7. 回归测试（Regression Testing）

- **目的**：确保新代码的更改没有引入新的错误或破坏现有功能。
- **方法**：在每次代码更改后，重新运行之前通过的测试用例。

2.8. 兼容性测试（Compatibility Testing）

- **目的**：确保系统在不同设备和浏览器上都能正常工作。
- **方法**：在各种操作系统、浏览器和设备上测试系统，检查界面和功能的兼容性。

2.9. 可用性测试（Usability Testing）

- **目的**：评估系统的用户友好性和易用性。
- **方法**：邀请用户进行测试，观察他们的使用过程，收集关于界面设计和用户体验的反馈。

2.10. 自动化测试（Automated Testing）

- **目的**：提高测试效率，减少人为错误。
- **方法**：使用自动化测试工具编写脚本，执行重复性的测试任务。

3. 测试用例设计

3.1. 主要的功能测试用例

1. **用户注册与登录**
   - 测试用例1：验证新用户能否成功注册。
   - 测试用例2：验证用户使用正确的凭证能否成功登录。
   - 测试用例3：验证用户使用错误的凭证能否被拒绝登录。
2. **浏览与发布求助帖**
   - 测试用例4：验证用户能否成功浏览所有求助帖。
   - 测试用例5：验证用户能否根据关键词搜索特定的求助帖。
   - 测试用例6：验证用户能否成功发布新的求助帖。

3. **资料分享与下载**
    - 测试用例7：验证用户能否成功上传学习资料。
    - 测试用例8：验证用户能否下载其他用户分享的资料。
    - 测试用例9：验证用户能否对下载的资料进行分类和标记。
    - 测试用例10：验证用户能否删除自己上传的资料。
4. **排行榜**
    - 测试用例11：验证排行榜是否根据用户完成的求助单数量正确排序。
    - 测试用例12：验证排行榜的更新机制是否实时。
5. **问题咨询**
    - 测试用例13：验证用户能否成功提出新的问题咨询。
    - 测试用例14：验证用户能否对问题进行回复和评论。
    - 测试用例15：验证用户能否编辑和删除自己的回复。
6. **资源库 – 学科分类**
    - 测试用例16：验证所有上传的历年卷和教材资源是否正确分类到相应的学科下。
    - 测试用例17：验证用户能否根据学科类别筛选和浏览资源。
    - 测试用例18：验证资源库中学科分类的更新机制是否能够及时反映新上传的资源。
7. **AI功能 – 题目总结**
    - 测试用例19：验证AI对历年卷题目的总结是否准确，包括知识点和整体难度的评估。
    - 测试用例20：验证AI总结的知识点是否覆盖了历年卷中的所有重要内容。
    - 测试用例21：验证AI对题目难度的评估是否与用户反馈一致。
    - 测试用例22：验证AI总结功能在不同类型的题目（如选择题、简答题、论述题）上的表现。
8. **AI功能 – 题目解析**
    - 测试用例23：验证AI对单个题目的解析是否准确，包括知识点、解题思路和方法的解释。
    - 测试用例24：验证AI解析的解题思路是否清晰易懂。
    - 测试用例25：验证AI解析功能在不同难度级别的题目上的表现。
9. **AI功能 – 举一反三**
    - 测试用例26：验证AI能否根据单个题目提供相关的强化训练题目。
    - 测试用例27：验证提供的强化训练题目是否覆盖了相同的知识点。
    - 测试用例28：验证AI举一反三功能在不同学科和题型上的表现。
10. **AI功能 – 提问获得指点**
    - 测试用例29：验证用户能否向AI提问，并获得相应的指点。

- 测试用例30：验证AI指点的准确性和相关性。
- 测试用例31：验证AI提问功能在不同学科上的表现。

11. **个人信息管理**
    - 测试用例32：验证用户能否成功修改个人信息（如昵称、头像、联系方式等）。
    - 测试用例33：验证用户能否查看和管理上传和下载历史记录。
    - 测试用例34：验证用户能否安全地注销账户。
12. **反馈与帮助文档**
    - 测试用例35：验证用户能否提交反馈并获得确认。
    - 测试用例36：验证用户能否轻松访问帮助文档。
    - 测试用例37：验证帮助文档是否更新到最新的系统信息和使用指南。

3.2. 性能测试用例

1. **负载测试用例**
    - 测试用例1：模拟1000名用户同时注册账户并记录响应时间。
    - 测试用例2：模拟1000名用户同时发布求助帖并记录响应时间。
    - 测试用例3：模拟1000名用户同时下载学习资料并记录响应时间。
2. **压力测试用例**
    - 测试用例4：持续发送请求，直到系统崩溃，记录最大承载用户数。
    - 测试用例5：在负载测试期间引入突发流量，观察系统的稳定性。
3. **响应时间测试用例**
    - 测试用例6：测试每个功能（注册、登录、上传、下载等）的平均响应时间，确保在可接受范围内。

3.3 安全测试用例

1. **身份验证测试用例**
    - 测试用例1：尝试使用无效凭证进行登录，确认系统拒绝访问。
    - 测试用例2：确认多次错误登录后账户被锁定。
2. **数据保护测试用例**
    - 测试用例3：验证用户数据在传输中的加密是否有效。
    - 测试用例4：确认用户密码以安全方式存储，无法被明文访问。
3. **漏洞扫描用例**

- 测试用例5：使用自动化工具进行漏洞扫描，记录并修复识别到的风险。

4. 测试计划

1. **测试准备阶段**
   - 设定测试目标和范围。
   - 准备测试环境，包括硬件、软件和网络设置。
   - 确定测试工具和测试框架。
2. **测试设计阶段**
   - 编写测试用例和测试脚本。
   - 确定测试数据，准备模拟数据。
3. **测试执行阶段**
   - 按照设计的测试用例执行测试。
   - 记录测试结果和异常情况。
4. **缺陷管理阶段**
   - 跟踪发现的缺陷，并分类和优先级排序。
   - 与开发团队沟通，确认缺陷修复情况。
5. **测试总结阶段**
   - 编写测试报告，总结测试结果和发现。
   - 根据测试结果提出改进建议。
6. **回归测试阶段**
   - 对修复的缺陷进行回归测试，确保不会影响其他功能。

八、项目管理

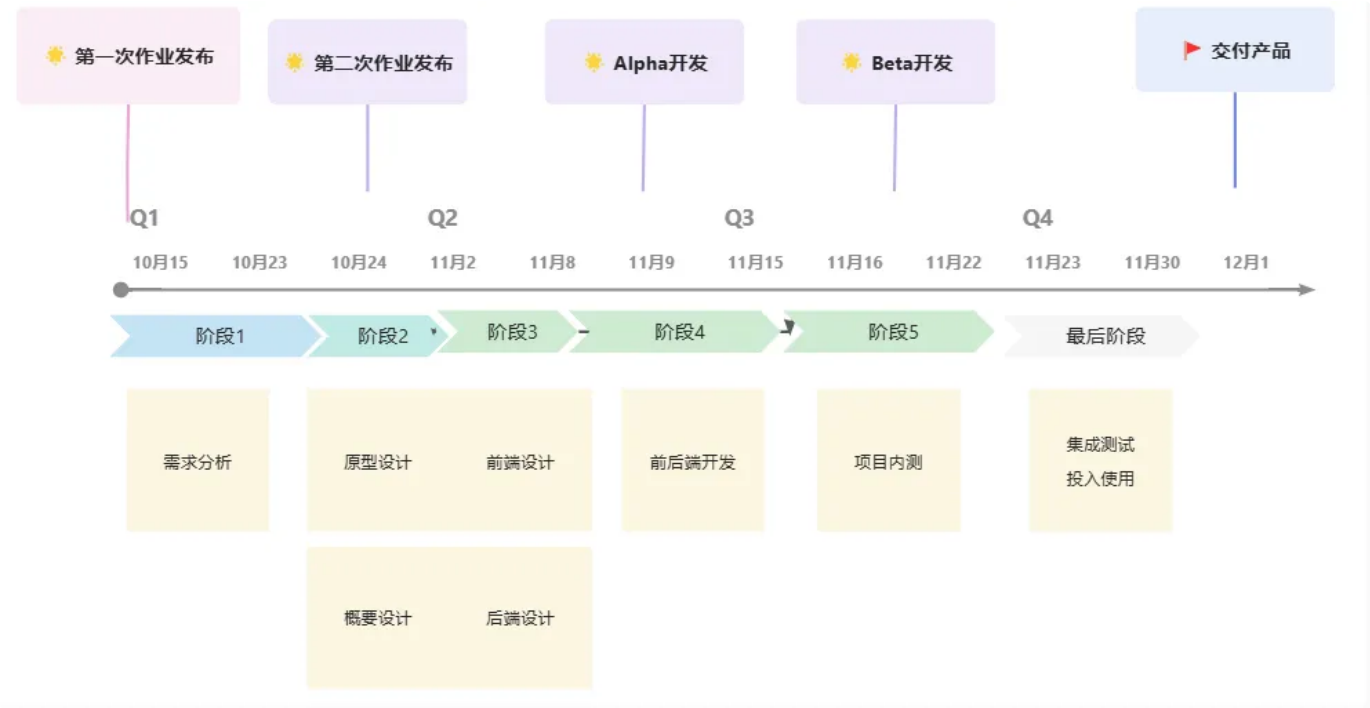
1. 开发计划

> 列出系统的开发周期及主要开发里程碑

周次	时间	计划

第一周	10月15–10月23	需求分析
第二周	10月24–11月02	原型设计+概要设计
第三周	11月02–11月08	前端页面设计、后端基础功能实现
第四周	11月09–11月15	完成前后端开发，撰写相关文档
第五周	11月16–11月22	前后端集成，系统功能初步演示，完成测试报告
第六周	11月23–11月30	审查代码，修复bug并优化性能，交予用户使用，收集反馈再度优化

里程碑



2. 人员安排

> 说明项目成员的角色和分工

姓名	角色	负责的开发部分
陈言泷	组长	项目整体规划与管理，协作沟通，统筹总结

柯鸿毅、周炳辉	前端开发人员	前端界面设计，用户交互功能实现
张硕	前端总负责人	统筹前端开发，协助管理
郑奇键、陈思宇	后端开发人员	API设计与实现，数据库管理
陈尚冰	后端总负责人	后端功能实现，编写相关文档
何愉心、邱雨涵	测试人员	负责测试计划的制订与实现，后续运行的反馈与改进，编写测试相关文档
赖锴彬	运维及对接员	负责运行维护及前后端对接

九、文档清单

> 列出所有相关文档及其用途（如开发文档、API文档、用户手册等）。

《数据库设计说明书》

十、参考资料

- [软工]概要设计说明书(GB8567-88)-CSDN博客
- 《系统设计说明书》国标规范
- 《人月神话》（Frederick P. Brooks Jr.）
- IEEE 1016-2009: 软件设计描述标准