软件工程小组 Part2

软件设计

**小组成员：尹航 + 张锐 + 罗一德 + 李松峪**

**文档介绍：**这是我们软件工程小组，在完成“基于人工智能的宿舍管理系统”的第二部分，即软件设计文档，通过描述子系统/组件职责和需求，以及设计优先级，来规范我们的软件设计，同时通过讨论主要设计问题，我们完成了我们的软件设计，将软件的主要功能制作成Class图，并概述了我们的系统架构，在软件过程设计中发现的一些其他问题一律放到其他细节当中，关于软件设计文档和软件设计有待完善的部分，我们列出了一个TODO来记录，以待之后对我们软件设计的优化。

## Task1 子系统/组件职责和需求

我们将需求文档的功能需求描述出来，来作为我们子系统和组件的主要组成。

|  |  |  |
| --- | --- | --- |
| 子系统/组件名称 | 目的或职责描述 | 对应的功能性需求 |
| 注册登录模块 | 负责用户的身份验证和会话管理，确保只有合法用户（学生或管理员）才能访问系统功能。 | 注册登录 |
| 个人与房间信息模块 | 负责录入、存储和维护学生用户的个人资料以及他们当前或未来入住的宿舍房间的详细信息和数据。 | 个人与房间信息 |
| 房间查询模块 | 为学生用户提供查询宿舍房间具体信息的功能，帮助学生了解宿舍环境，弥补信息差。 | 房间查询 |
| 抢房模块 | 提供特定场景下（如学期初或特定时间段）学生用户在线快速选择并锁定房间的功能。 | 抢房 |
| 换房模块 | 负责处理学生用户提出的换宿舍申请和流程，包括交换房间或交换空房间等操作，简化传统的繁琐程序。 | 换房 |
| 论坛交流模块 | 搭建一个社区交流平台，促进学生互动，使他们能够了解彼此的生活习惯和作息，以便选择更合适的舍友，减少生活矛盾。 | 论坛交流 |
| AI 问答模块 | 提供一个基于人工智能的问答机器人，即时、智能地解答学生关于住宿和系统使用的基础疑问，并整合论坛信息提供更详细的内容。 | AI 问答 |
| 管理模块 | 供系统管理员使用，用于高效地管理学生和房间信息、审批换房等申请流程、处理学生反馈，以及进行黑名单、删除违规信息等宿舍管理工作。 | 管理 |
| 意见反馈模块 | 提供用户提交对系统或宿舍管理方面建议或问题的渠道，支持系统管理员对反馈进行处理。 | 意见反馈 |

## Task2 设计优先级

我们的项目系统根据设计初衷和需求文档，主要遵循以下五大优先级。

**信息透明：** 这是项目的核心创新点。我们要保障学生能实时、全面地了解宿舍环境信息和潜在舍友的生活作息，从根本上弥补信息差，减少不必要的生活矛盾。

**流程便捷：** 我们必须保障系统能简化和优化换宿舍和入住的程序。设计的重点在于克服传统模式的繁琐，满足学生对高效、便捷流程的需求。

**管理效率：** 我们要保障管理员能够高效地完成审批流程、管理信息、并处理反馈，确保系统能够方便学校进行宿舍管理。

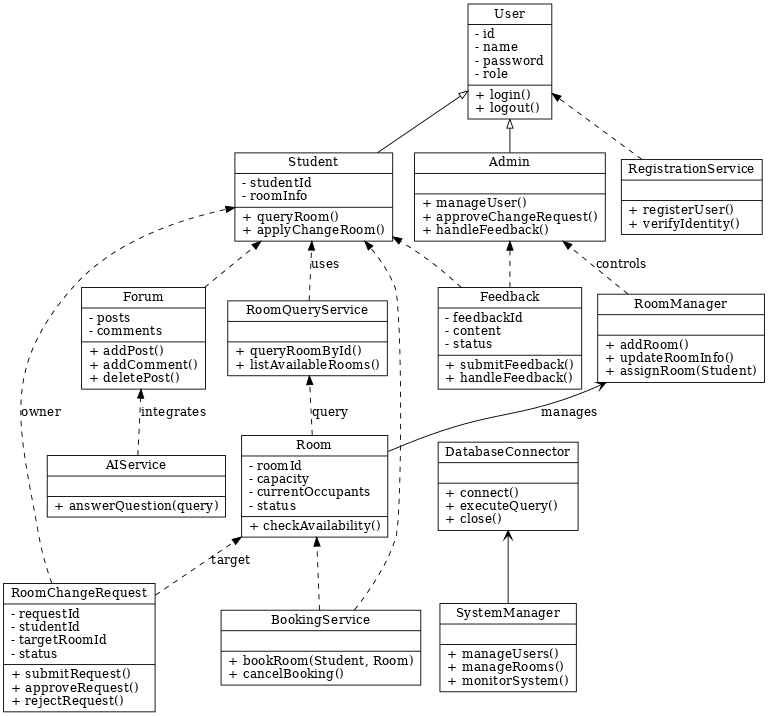
**AI集成：** 我们为了打造智能的宿舍管理系统，需要保障AI功能的成功集成，让AI文答功能辅助我们的系统，提升我们系统的服务体验。

**质量保证：**我们要确立技术规范和保障系统采用vue + spring的流行架构，确保我们的代码的可复用性、可扩展性和可靠性，同时需要注意在登录方面的安全性问题。

## Task3 设计大纲及设计方法

我们采用了混合式的设计方法。

自顶向下：首先从系统整体出发，划分主要子系统（注册登录、房间管理、换房、AI问答、论坛交流等），自底向上：在具体实现中，从可复用的类和组件（如 User、Room、DatabaseConnector）入手，逐步组合成高层模块。



## Task4 主要设计问题

### 系统架构

**问题描述：**

我们决定采用前后端分离的客户端-服务端架构。关键的设计挑战是，选择一个能够最大限度保障系统的可扩展性、可控性、代码质量与规范，并支持我们异步响应式交互需求的流行技术栈。

**可行方案：  
A Vue前端 + Spring后端框架  
B React前端 + Spring后端框架**

**C React前端 + Nodejs后端框架  
最终选取：**

**A Vue前端 + Spring后端框架。**

**选取理由：**1. Vue 的学习曲线相对平缓，且与 Spring 组成的稳定架构易于理解和掌握，能最大化开发效率和项目可控性。2. 该组合是成熟的企业级解决方案，能够保障系统的可扩展性、可维护性和可靠性。3. Spring 提供了强大的依赖注入和事务管理能力，支持数据库和接口交互，且该架构天然支持我们所需的异步响应式交互。同时，流行的框架生态有利于代码仓库管理、代码质量和规范的维护（更容易集成 ESLint 等工具）。

### 界面实现方式

**问题描述：**

在设计“基于人工智能的学生宿舍管理系统”的用户界面时，我们面临选择交付形态的关键挑战。我们必须确定是采用更具跨平台性和便捷性的浏览器交互（Web应用）方式，还是采用在性能和原生体验上有优势的传统软件 GUI (图形用户界面) 方式。同时我们决策必须考虑到用户（学生和管理员）的访问便捷性和系统的部署维护成本。

**可行方案：  
A 浏览器交互（Web 应用）：** 界面通过浏览器访问

**B 独立软件 GUI 交互：** 开发桌面客户端  
**最终选取：**

**A 浏览器交互（Web 应用）**  
**选取理由：**1、浏览器不用额外学习Vue GUI架构的知识，对我们小组的技术能力更加的友好，控制了额外的时间成本。2、浏览器访问更加的便捷，适合于学生和管理员随时的操作，我们只用做手机端浏览器的适配，就能和手机软件实现同样的效果，不仅优化用户体验还能够增强软件能力。3、Web应用的模式更加符合Vue+Spring的交互逻辑，能够规范我们的代码。

### AI功能集成

**问题描述：**为了保障系统在新时代的竞争力，我们小组准备为我们的系统集成AI功能，建立一个AI问答模块，考虑到成本和可靠性，我们也要对集成方式权衡。

**可行方案：  
A 将微调模型部署到服务器**  
**B 调用第三方API**

**最终选取：  
B 调用第三方API  
选取理由：**1、考虑到deepseek V3.2模型的降价，此时选取deepseek api的成本可控。2、我们小组的计算机算力不支持我们为我们的项目独立微调出一个模型，且调用模型对服务器的压力过大，成本反而更高，同时远没有第三方api可靠。

### 信息透明和信息交流

**问题描述：**如果我们的系统要注重用户隐私的同时，就必然会导致一部分的信息不可见和无法交流，所以为了保持我们系统的竞争力和特点，我们要权衡信息交互方法。

**可行方案：  
A 完全公开用户和宿舍信息**  
**B 最大程度保护用户隐私的情况下公开信息  
C 采取评价系统**

**最终选取：  
C 采取评价系统  
选取理由：**1. 评价系统是实现信息透明的最佳折中方案。它允许学生对宿舍环境、作息习惯等进行匿名或半匿名的量化打分和评论。这取代了直接公开学生个人信息的需求，成功地在保障用户隐私和提供有效交流信息之间取得了平衡，保持了我们产品的竞争力。2. 通过建立统一的评价标签（例如“早睡党”、“夜猫子”、“爱干净”等），系统能够将主观信息转化为可量化的数据。这些数据可供换房模块和房间查询模块作为舍友互适性的参考依据，从而直接支持了项目“选择更合适的舍友”的核心目标。3. 评价系统的内容由真实住户生成，具有更高的可信度和实时性，形成了社区反馈闭环，增强了系统的数据可靠性和用户粘性，这是简单地公开原始数据所不具备的优势。4、考虑到我们为系统竞争力而设立的论坛功能，评价系统和论坛功能的实现逻辑相符，所以能够实现在最小支出的情况下实现我们产品的竞争力。

## Task5 架构模式

### 系统架构

**Vue前端 + Spring后端 + MySQL数据库（客户端-服务端架构）**

**前端：** Vue.js + antD Vue UI组件库 + TypeScript

**构建/路由：** Vite/Vue router

**网络通信：** Websocket  
**后端：**Java + Spring  
**AI集成：**Deepseek api

### 开发环境与工具

**代码仓库/代码提交：**github/git

**代码格式化工具：**ESLint+Pretter  
**测试工具：（待补充）  
文档工具：**腾讯文档  
**项目管理：**腾讯文档

### 选择理由

前后端分离是典型的分治思想，同时Vue组件化和Spring聚焦于后端的业务逻辑，保证了系统的内聚性原则，采用Spring API则是很好的降低耦合度的方法，采用TypeScript编写代码使得我们的项目灵活性更强，而Vue的组件化也符合可复用的原则，所以我们的架构符合需要的设计原则。

## Task6 其他细节

**关于主要设计问题：**在我们进行软件设计的时候，如代码仓库、代码格式化工具、web组件库、接口管理工具采取了遵循过往经验的判断，虽然这些工具都经过了诸多用户和企业的检验，但我们小组并没有细致的讨论其他可行方案，这可能成为软件之后的风险。

**文明规范：**本文未涉及对用户发言和评论的设置，所以可能在之后的系统中出现恶意和违规发言，系统采取的是由管理员统一管理的形式，来处理不规范的发言。

## TODO

### ~~补充测试工具~~

低优先级

暂不考虑测试内容

### 优化Class图（数据标签）

为Task3的Class图补充数据标签，PPT17页之后的部分

### 需求对应

链接Task1和需求文档，在需求文档更新之后再改

### 架构图

PPT56页之后的部分

### 对应子系统/组件和优先级

为Task2添加一个新表，列出子系统/组件和优先级的对应关系