

# 基于多智能体的试卷生成系统

Test paper generation system based on multi-agent

所在赛道与赛项：A

中国高校计算机大赛-网络技术挑战赛

## 目录

基于多智能体的试卷生成系统 .....	1
一、 目标问题与意义价值 4	
1.1 背景分析 .....	4
1.1.1 人工出题现状 .....	4
1.1.2 人工智能的发展 .....	4
1.1.3 数字化教育改革 .....	5
1.2 目标与基本功能 .....	6
1.2.1 实现目标 .....	6
1.2.2 基本功能 .....	7
1.3 意义与应用价值 .....	8
二、 计思路与方案 9	
2.1 主要设计思路 .....	9
2.2 技术路线 .....	10
2.3 设计方案 .....	11
2.3.1 多智能体协同 .....	11
2.3.2 试题难度定义 .....	13
2.3.3 题库管理 .....	15
2.3.4 算法组合试卷 .....	18
2.3.5 多租户模式 .....	23
三、 方案实现 25	
3.1 网页技术架构 .....	25
3.1.1 总体技术架构 .....	25
3.1.2 前端与后端技术 .....	26
3.1.4 前后端交互 .....	27
3.1.5 环境部署 .....	28
3.1.6 云数据库 .....	29
3.3 多智能体实现 .....	29
3.3.1 提示词工程 .....	29
3.3.2 动作实现 .....	30
3.3.3 角色实现 .....	31
3.3.4 记忆实现 .....	32
3.4 题目生成实现 .....	33
3.4.1 文档读取与处理 .....	33
3.4.2 摘要生成 .....	33
3.3.3 题目生成与审查 .....	34
四、 实现与验证 36	
4.1 应用效果实现 .....	36
4.1.1 首页 .....	36
4.1.2 登录页面 .....	36
4.1.3 生成题目页面 .....	37
4.1.4 试题库页面 .....	38
4.1.5 组卷页面 .....	39
4.1.6 出题历史页面 .....	40

4.2 数据验证 .....	41
五、 创新特色 .....	44
5.1 多智能体协同生成试题 .....	44
5.2 布鲁姆分类法定义试题难度 .....	44
5.3 贪心与遗传算法最优化组卷 .....	44
5.4 试题库管理与可视化 .....	45
5.5 行级别隔离的多租户模式 .....	45
六、 总结与展望 .....	46
6.1 工作总结 .....	46
6.2 未来展望 .....	47
参考文献 .....	48

# 一、目标问题与意义价值

## 1.1 背景分析

### 1.1.1 人工出题现状

考试是评估学生学业成就、教师教学能力、个体才能以及实现教育分流的重要工具，在教育教学体系中具有关键地位。尽管传统的人工出题方式凭借教师的丰富经验和对知识点的精准把握，能够根据学生的具体情况灵活调整，但由于个人精力和时间的限制，难以确保每份试卷都经过深度优化。此外，人工出题过程中不可避免的主观因素可能导致试题难度、区分度以及试卷的信度和效度出现偏差。在大型考试的筹备中，通常需要大量人力进行试卷的反复测试与优化，这不仅效率低下，还可能增加试卷保密性的风险。随着考试形式的日益丰富和标准的提高，教师面临的工作负担愈加沉重，出题工作变得繁琐且易出错，亟需寻找更高效、精准的出题方式。

### 1.1.2 人工智能的发展

根据新华社研究院中国企业发展研究中心发布的《人工智能大模型体验报告》，人工智能（AI）正引领新一轮科技革命和产业变革，势头强劲且持续。AI 技术在移动互联网、大数据、超级计算、传感网及脑科学等新理论和技术的推动下，展现出深度学习、跨界融合、人机协同、群智开放和自主操控等特征。这些特征不仅为经济发展注入新动力，也推动社会进步和全球治理方式的变革。人工智能技术已成为全球最重要的技术领域之一，掌握并发展 AI 技术是实现科技自立自强的关键。在中国，AI 技术被视为推动数字经济发展的核心技术，预计在未来将发挥关键作用。自 2023 年以来，大模型技术在 AI 领域受到广泛关注，越来越多的中国科技企业投入研发并推出自有大模型产品，这不仅体现了我国 AI 技术的快速发展，也预示着 AI 技术将在更多领域得到广泛应用，推动社会的持续进步。

### 1.1.3 数字化教育改革

信息技术的迅猛发展推动了教育领域的数字化转型，成为全球教育改革的重要方向。数字化教育拓宽了学习机会，丰富了教育资源，为学生量身定制个性化和高效的学习路径。在复杂多变的社会经济环境中，数字化教育成为提升教育质量和促进教育公平的关键。数字化手段优化了资源配置，提升了教学效率与质量，打破了传统教育的时空限制，实现学习的自主与灵活。结合数据分析与人工智能等先进技术，教育过程变得更加智能化，能够精准评估学生学习状况，提供个性化反馈与指导，助力全面发展。

数字化教育改革的核心在于信息技术与传统教学的融合，构建便捷、丰富的学习平台，使教师能够高效组织教学内容，实施个性化教学，学生则根据个人兴趣与进度，自主选择资源与学习路径，迈向自主学习与终身学习的时代。

人工智能的广泛应用使教育更加精准、科学和公平，推动教育创新发展。传统的人工出题模式存在耗时费力、主观性强、质量难以保障等弊端，而数字化教育改革提供了解决这些问题的新思路。通过开发多智能体的试卷生成系统，可以实现试卷生成的自动化和智能化，提高试卷生成效率和质量，减轻教师工作负担，推动教育教学的创新发展。

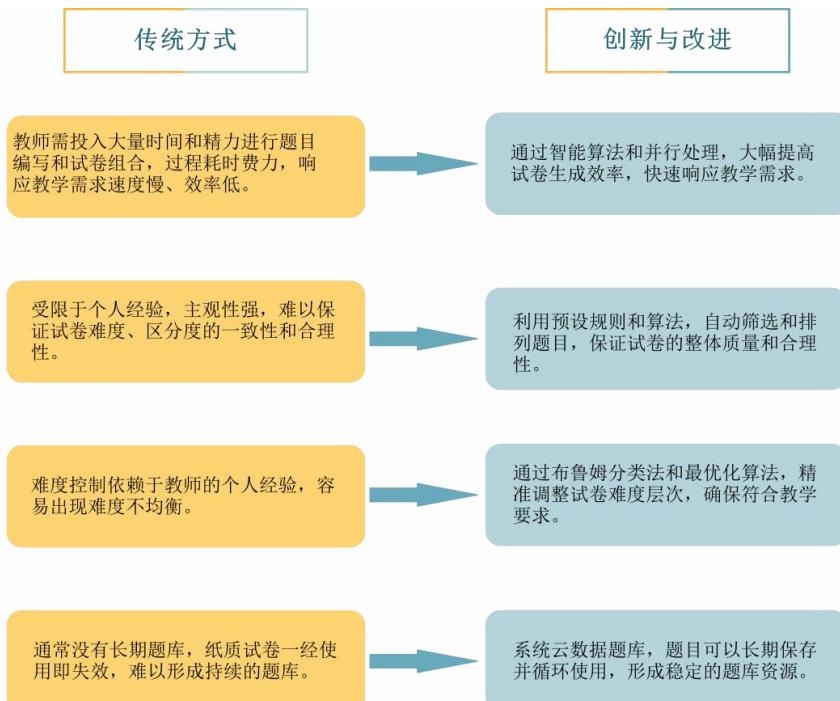


图 1-1 传统方式与创新改进

## 1.2 目标与基本功能

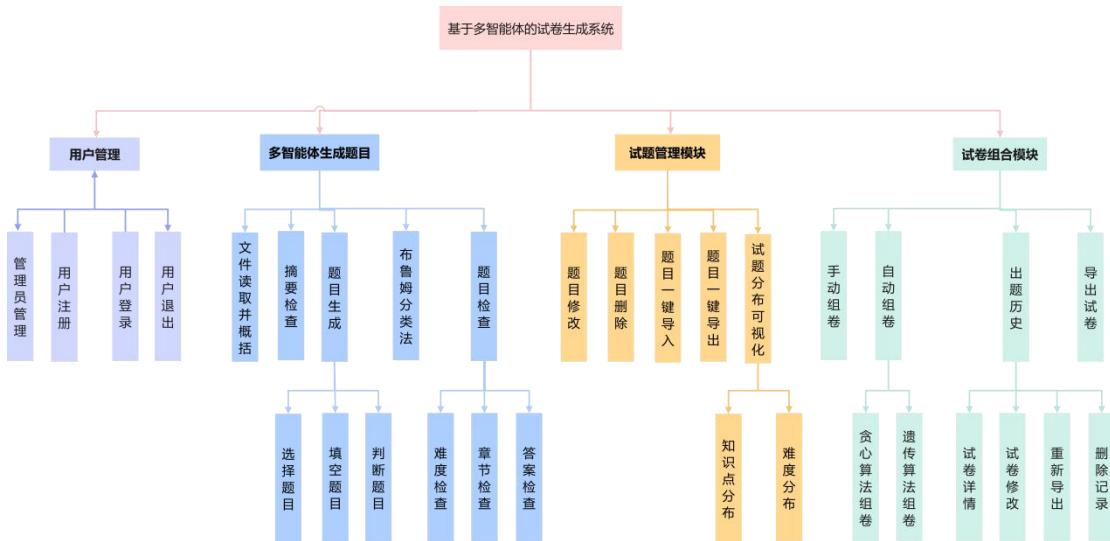
### 1.2.1 实现目标

本项目利用基于多智能体的试卷生成系统，旨在显著提高试卷生成效率，并根据教师需求和学生实际情况，快速生成符合要求的试卷，为教学评估提供支持。具体目标包括：

- **实现提高题目生成效率：**传统的题目生成过程通常依赖教师的手工操作，需要大量的时间和精力来选择、修改和排列题目。多智能体系统通过引入并行处理机制，使多个智能体能够同时工作，分担不同的任务，如题目选择、难度评估和题目排序等。各智能体通过协同合作，实现了比单一操作更高的效率。这种自动化处理方式不仅减少了教师的工作量，还能够在短时间内生成多套不同的试卷，满足不同教学场景的需求。
- **实现保障试卷生成质量：**手工生成试卷容易受到主观因素的影响，如教师个人偏好、时间压力等，这可能导致试卷难度分布不均、题目重复或偏离教学目标等问题。系统通过预设的规则和算法，自动筛选题目，并依据特定的教育目标对试卷进行优化。系统利用数据分析和题库中的题目元数据（如知识点覆盖、难度系数等），确保生成的试卷在内容覆盖、难度层次和题目多样性方面都具有较高的质量。
- **实现控制题目生成难度：**在传统试卷生成过程中，教师通常难以精准控制试卷的整体难度，容易导致试卷过难或过易。多智能体系统通过结合布鲁姆分类法，对题目进行分类，如记忆、理解、应用、分析等难度层级。系统根据学生的实际水平、教学进度和评估目标，通过智能算法对题目进行合理的选取和排列，调整试卷的难度分布，以确保试卷的整体难度与教学需求和学生能力相符。
- **实现自动组合试卷：**传统试卷的组合过程通常需要教师进行手动排版和调整，不仅耗时而且容易出错。系统通过自动化流程，从题库中筛选题目后，利用算法自动将题目按照预设的模板进行组合和排版。这一过程大大降低了手工操作的错误率，同时也减少了教师在试卷组合和排版方面的工作量。

## 1.2.2 基本功能

基于多智能体的试卷生成系统的基本设计思想是把整个系统按照实现模块进行分解，基本功能如图 1-2 所示：



### (1) 用户管理模块

提供用户注册、登录和注销功能，确保系统的安全性和用户信息的隐私保护。管理用户角色可以审核通过注册申请，并根据角色分配不同的权限。

### (2) 多智能体生成题目模块

- 1) 文件读取并概括：系统能够读取上传的文件（如 PDF、Word 等），并自动生成该文件的摘要，提取出关键的知识点和信息。
- 2) 摘要检查：自动检查生成的摘要是否准确、完整，确保信息提取的质量和可靠性。
- 3) 题目生成 利用多智能体系统，根据文件内容和教学目标自动生成试题，支持指定不同题型（如选择题、填空题、简答题等），并根据不同难度和不同章节来源生成合适的题目。
- 4) 难度控制：通过智能算法，根据预设的标准对生成的题目进行难度控制，确保题目适合不同层次的学生，并能涵盖不同层次的知识点。
- 5) 题目检查：提供自动化的题目检查功能，包括难度检查、章节检查、答案检查，确保生成的题目符合预期的标准和要求。

### (3) 试题管理模块

- 1) 题目修改与删除：对生成的题目进行编辑、修改或删除，确保试题的质量和适用性。
- 2) 题目一键导入、导出：支持将题目批量导入到试题库，或者从试题库中导出，方便管理和使用题目资源。
- 3) 试题分布可视化：提供题目的可视化管理工具，包括知识点分布和难度分布的图表，帮助教师直观地了解题库的整体情况，优化试题的选择和使用。

#### (4) 试卷组合模块

- 1) 自动组卷：贪心算法或遗传算法，根据设定的难度、题型比例和学科知识点自动组合生成试卷。
- 2) 手动组卷：允许教师根据需要手动选择试题，灵活地组合生成试卷。
- 3) 出题历史：系统保存每次出题的历史记录，教师可以查看和管理试卷历史记录，包括删除历史记录、查看试卷详情、修改试卷，并支持试卷的重新导出，方便教师进行试卷的管理和复用。
- 4) 试卷导出：支持将生成的试卷导出为 Word 文件，方便打印和分发。

### 1.3 意义与应用价值

AI 技术的引入推动了教育评估理论的变革，特别是在试卷出题的标准化与科学化方面。通过智能算法和数据驱动的方法，我们可以更精准地控制试卷的难度和质量，减少人为主观因素的干扰，这为教育评估理论的客观性和有效性提供了新的支持。其次，本项目应用的 MetaGPT 框架和多智能体系统，体现了智能体理论在教育评估中的实际应用。这不仅拓展了智能体理论的应用领域，也为教育技术的研究提供了新的方向。

通过自动化和智能化的试卷生成过程，系统能够大幅提升试卷生成的效率和质量。教师的工作负担因此得以显著减轻，使他们能够将更多精力投入到教学和学生发展上。此外，系统通过高效的算法分析和智能优化，确保试卷难度和内容的科学性与公平性，有助于提供一个更加公正和客观的考核环境。

系统的自动化特性有效解决了传统人工出卷方式中的公平性和保密性问题，提升了试卷生成和评估的整体水平。这不仅优化了教育资源的配置，也推动了教育公平和教学质量的提升，为教育工作者提供了一种高效、可靠的工具，具有广泛的实际应用前景。

## 二、计思路与方案

### 2.1 主要设计思路

首先基于用户需求和上传的文件，利用智能体实现 PDF 内容的完整读取和快速编译。其次建立一个多智能体问题生成并反复审查的题目生成流程，并基于布鲁姆分类法定义生成题目难度，利用云数据库存储题目及其属性信息。最终，基于贪心算法与遗传算法从题库抽取试题，最优化自动组合试卷，实现生成与组卷。因此，整个系统的设计可以分为四个部分：多智能体文件读取与处理、多智能体问题生成与审查、试题库管理与可视化、最优化试卷组合，如图 2-1 所示。

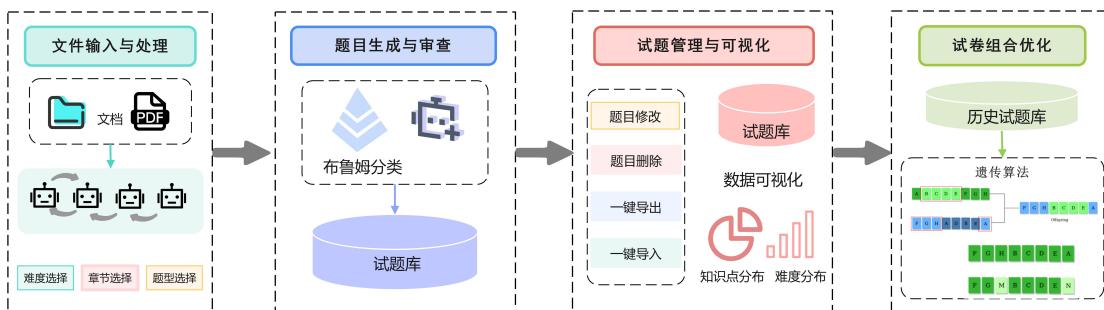


图 2-2 系统架构图

**多智能体文件读取与处理：**系统首先接收用户上传的文件，并获取用户试题难度、章节、题型选择等需求。智能体负责读取 PDF 文件的内容，并将其转化为系统可以处理的格式。另一个智能体则负责对内容进行编译和提取，以便后续生成题目。

**多智能体问题生成与审查：**基于布鲁姆分类法，将生成的题目按照不同的认知层次进行分类。智能体根据用户需求和内容类型自动生成问题。生成的题目通过多个智能体进行反复审查，确保题目的准确、难度适中。

**试题库管理与可视化：**使用云数据库存储生成的题目及其属性信息（如难度、分类、提供一个前端界面，允许用户查看、筛选、编辑题库中的题目。

**最优化试卷组合：**使用贪心算法与遗传算法从题库中抽取试题，根据用户定义的条件（如难度、覆盖知识点等）最优化自动组合试卷。

## 2.2 技术路线

本系统的技术路线如图 2-2 所示。

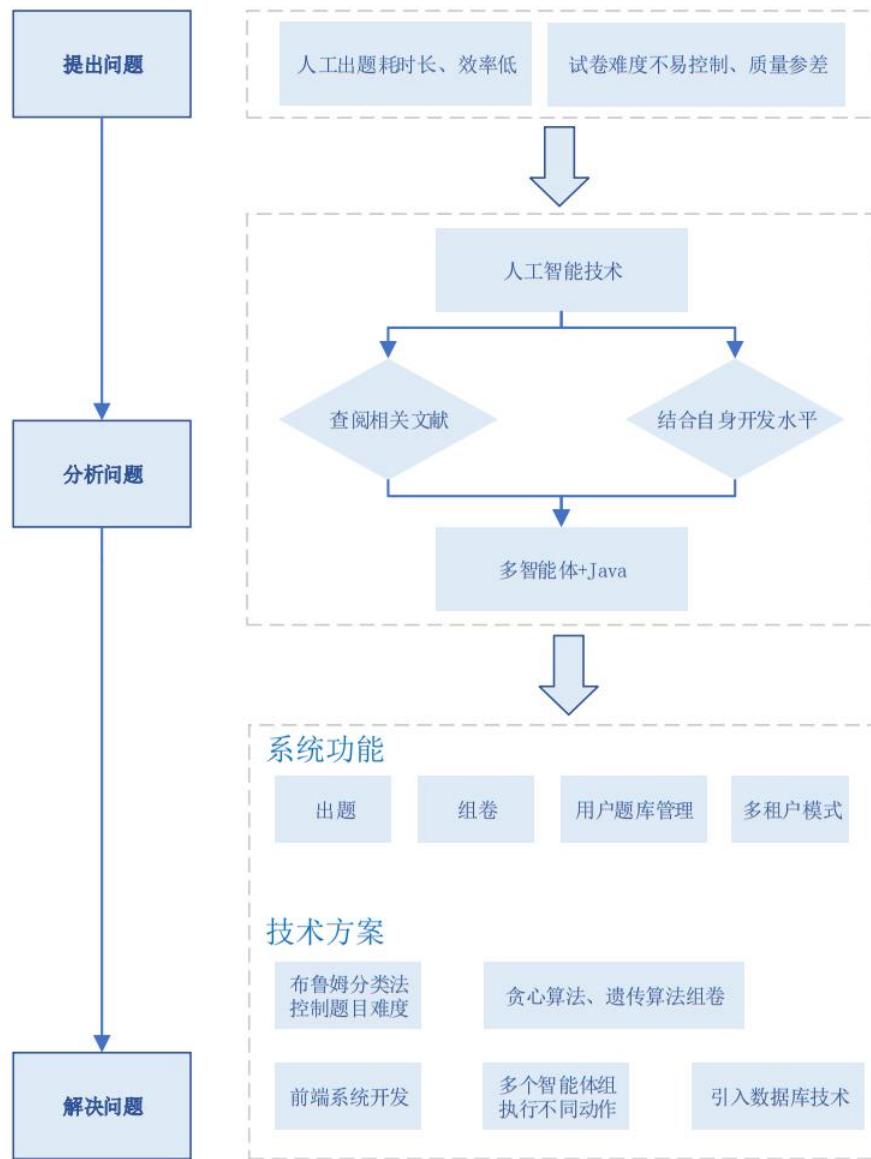


图 2-2 技术路线图

通过上述技术路线的设计，可以充分利用先进的自然语言处理技术、智能体协同策略、数据库技术、布鲁姆分类法以及优化算法，实现高效、科学的题库生成和试卷组卷系统。大语言模型的引入可以显著提高题目生成的质量和多样性，多智能体协同策略则能够确保各个环节的高效协作和灵活扩展。数据库技术提供了可靠的数据管理和检索功能，布鲁姆分类法则为题目难度的合理分配提供了科学依据。最后，贪心与遗传算法的应用在组卷过程中进一步保证了试卷的科学性和合理性。

## 2.3 设计方案

### 2.3.1 多智能体协同

在当代教育评估系统的设计中，高效且准确地处理大量复杂的文档信息，并自动化生成高质量的试题，始终是一个关键的挑战。为了有效应对这一挑战，本系统设计了一种多智能体协同工作的试题生成方案，充分利用了人工智能技术，以显著提升试题生成的效率和质量。多智能体系统通过模块化设计，使不同功能的智能体能够分工明确、协同作业，从而实现系统的高效运作。整体架构如图 2-3 所示，各模块之间的交互采用了串行、并行及循环反馈结构，确保整个系统的有序性和灵活性，整体架构如图 2-3 所示。

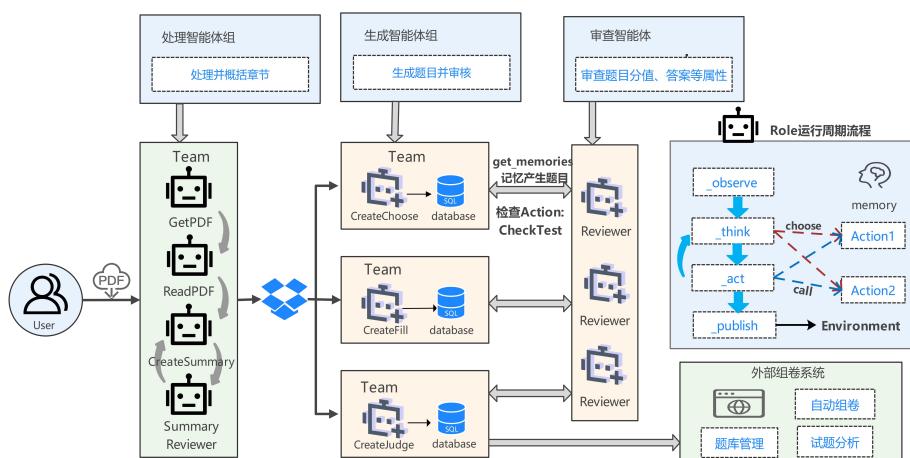


图 2-3 多智能体整体架构图

#### (1) 处理智能体组

处理智能体组负责用户上传文档的初步处理和信息提取工作。该模块接收由用户上传的包含相关知识的 PDF 文件或考试大纲，随后由一系列智能体协作解析文件内容并生成文件摘要。具体而言，GetPDF 智能体负责接收和存储用户上传的 PDF 文件，ReadPDF 智能体负责读取并解析 PDF 文件的内容，CreateSummary 智能体负责生成内容摘要，最后由 Summary Reviewer 智能体审查生成的摘要，以确保其准确性和完整性。

在处理智能体组内部，各智能体之间存在多种结构组合，包括串行、循环和判断结构。串行结构用于按顺序处理输入和输出，保证信息处理的流畅性。例如，从文档上传到内容解析，再到摘要生成的过程，均按此顺序进行。循环结构则用

于在生成的摘要不准确或不完整时，智能体反复执行摘要生成操作，直至符合要求。判断结构则允许系统根据不同条件进行分支处理，例如在处理文档内容时，系统根据文档类型选择不同的解析策略，从而提升处理效率和准确性。

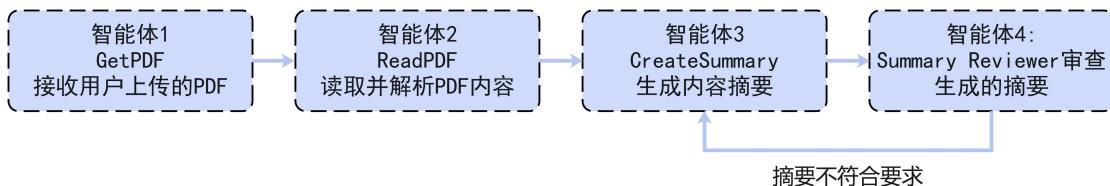


图 2-4 处理智能体组

## (2) 生成智能体组

生成智能体组依据处理智能体组生成的内容摘要及用户输入的题目要求，自动生成不同类型的题目并将其存储至题库中。具体地，CreateChoose 智能体生成选择题并将其存入数据库，CreateFill 智能体生成填空题并存储，CreateJudge 智能体生成判断题并同样将其保存至数据库。

系统在生成题目时，依据布鲁姆分类法来确保题目的难度和质量。在生成题目之后，Reviewer 智能体将对所有题目进行全面审查。若发现题目分值、答案不符合要求，或者难度不符合布鲁姆分类法的标准，系统将返回生成智能体组进行重新生成，以确保最终题目质量符合预期标准。

生成智能体组的各个智能体，包括选择题生成智能体、判断题生成智能体和填空题生成智能体，采用并行结构进行设计。这一设计允许多种题型的生成任务同步进行，大大提升了试题生成的速度与效率。每个智能体能够独立运行，利用处理智能体组生成的摘要内容，同时生成各自类型的题目，并且通过并行处理，系统可以在较短时间内完成多种题目的生成任务。

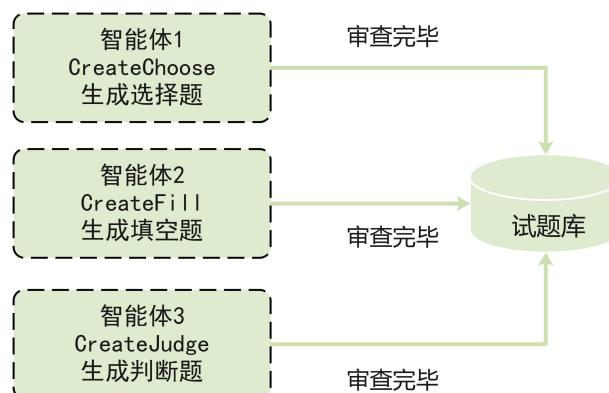


图 2-5 生成智能体组

### (3) 审查智能体组

Reviewer 智能体负责检查生成题目的各项属性，包括题目分值、答案的正确性、所属章节及其难度是否符合布鲁姆分类法的要求。审查智能体组与生成智能体组之间通过循环反馈结构相连。在审查过程中，若发现生成的题目在质量上存在问题，如分值设置不合理、答案不准确或难度偏离预定标准，审查智能体将启动反馈机制，将问题题目返回至生成智能体组进行重新生成。

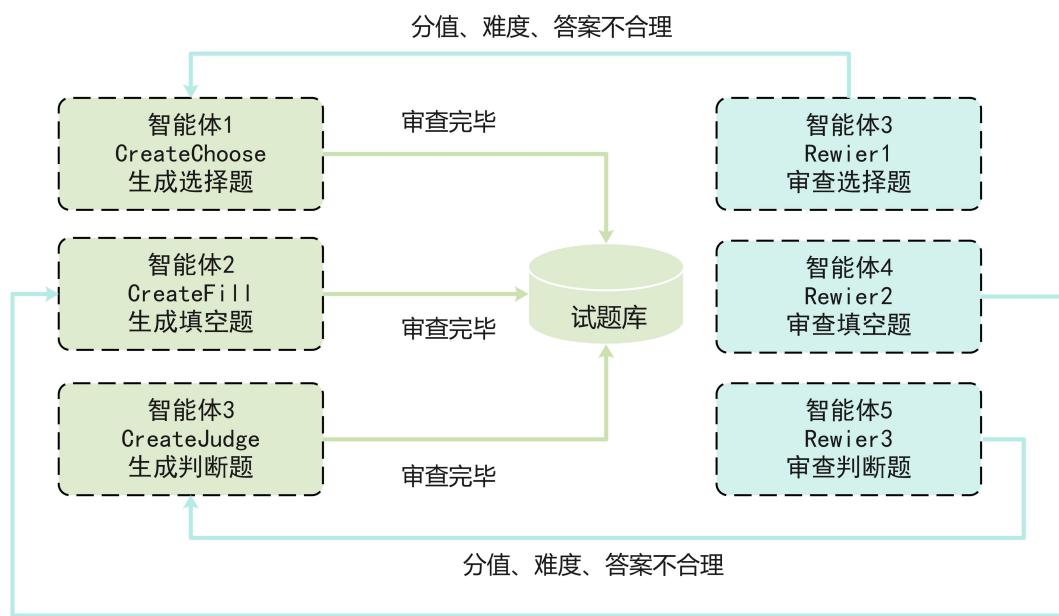


图 2-6 审查智能体组

### 2.3.2 试题难度定义

**记忆（L1）：** 学生能够回忆或识别已经学习过的信息和知识。此层次的问题通常涉及简单的记忆任务，如记住公式、定义、日期、事件和具体事实等。

**理解（L2）：** 学生能够解释或阐述所学内容，展示对信息的理解。这包括解释概念的意义、总结段落、阐述因果关系等。

**应用（L3）：** 学生能够在新的情境中使用所学知识和技能。这些问题通常要求学生应用所学公式解决问题、运用理论解释现象或进行实际操作。

**分析（L4）：** 学生能够分解信息和知识结构，理解其组成部分及其相互关系。例如，分析文章的结构、区分事实与观点、辨识关系模式等。

**评价（L5）：**学生能够根据标准和准则，做出判断并解释其理由。这包括评估方案的优缺点、判断实验结果的可靠性、进行批判性分析等。

**创新（L6）：**学生能够结合所学知识创造出新的思想、产品或方法。典型的例子包括设计实验、提出新理论、创作艺术作品、开发新应用等。

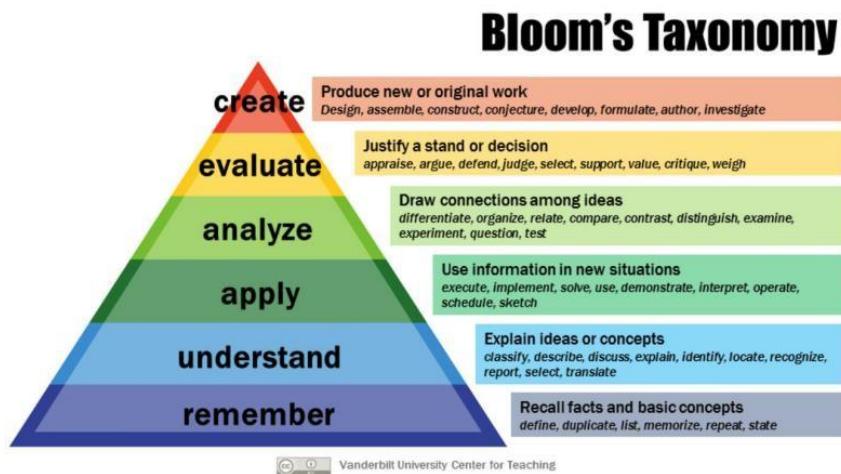


图 2-7 布鲁姆分类法层次结构

本系统结合布鲁姆分类法，并利用大语言模型（Large Language Model, LLM）对生成的每道题目进行分类打标签。在题库生成过程中，系统严格遵循布鲁姆分类法的指导原则，确保题库中涵盖不同认知层次的题目，达到从基础的记忆（L1）到复杂的创新（L6）的全面覆盖。这些标签不仅有助于区分题目的难度和类型，还能确保题库的多样性和合理性。

每道题目在生成时，系统都会根据其涉及的认知层次自动打上布鲁姆分类法的标签。例如，涉及记忆类任务的题目标记为 L1，涉及理解类任务的题目标记为 L2，以此类推直到创新类任务的题目标记为 L6。通过对题目难度的精确分类，系统能够在试卷生成时提供难度合理分布的题目，帮助教师根据学生的掌握情况调整教学策略。

布鲁姆分类法不仅为每道题目提供了科学的难度分级标准，还为试卷的整体难度控制提供了依据。通过合理分配各层次的题目，系统确保了生成试卷的难度分布合理，有助于设计全面的测试题目，覆盖知识、理解、应用、分析、评价、创新等各个认知层次。这种难度分类方法能够帮助教师更好地理解学生的学习状况，并提供针对性的辅导和练习，从而提高教学效果。

### 2.3.3 题库管理

题库管理模块是系统的核心组成部分之一，负责题目数据的管理、检索、更新和维护，确保题库中的题目始终满足系统需求，并为组卷模块提供准确的题目数据支持。该模块主要由以下六个表组成：**QuestionBank**、**QuestionMaterial**、**TestPaperGenHistory**、**User**、**QuestionLabels**、**Questiongenhistory**。这些表结构支持模块的主要功能，如题目的存储、标签化、用户行为记录以及试卷生成历史的追踪。各数据表介绍及功能如下：

#### QuestionBank（题库表）

描述：该表存储所有生成的题目，包含题目的基本信息，如题目内容、类型、难度、分值、章节和标签等。它是题库管理模块的核心数据表。

功能：

存储题目内容及其属性（类型、难度、分值、章节等）。

关联题目素材表，实现题目内容的细化管理。

提供基础数据供组卷模块使用。

字段说明：

**id**: 题目唯一标识符（主键）。

**topic**: 题目内容，以长文本形式存储。

**topic\_material\_id**: 关联 QuestionMaterial 表的素材 ID。

**answer**: 题目答案。

**topic\_type**: 题目类型（如填空题、选择题等）。

**score**: 题目分值。

**difficulty**: 题目难度级别。

**chapter\_1**、**chapter\_2**: 题目所属的章节信息，用于知识点分类。

**label\_1**、**label\_2**: 题目的标签信息，用于题目分类及检索。

**update\_time**: 题目的更新时间。

#### QuestionMaterial（题目素材表）

描述：该表存储与题目相关的素材信息，如题目选项、详细内容等，用于辅助题目的生成与存储。

功能：

保存题目相关的详细素材，供题目生成时调用。

通过与 QuestionBank 表的 topic\_material\_id 字段关联，实现题目和素材的关联管理。

字段说明：

id: 素材唯一标识符（主键）。

material: 素材内容，以长文本形式存储。

update\_time: 素材的更新时间。

TestPaperGenHistory（试卷生成历史表）

描述：该表记录所有试卷生成的历史信息，包含生成的试卷名称、题目数量、难度分布等数据，用于追踪历史试卷的生成记录。

功能：

存储试卷生成的基本信息，包括试卷名称、生成时间、题目数量、平均难度等。

用于后续查询和重新生成试卷。

支持生成日志的记录和统计分析。

字段说明：

id: 记录唯一标识符（主键）。

test\_paper\_uid: 试卷唯一标识符。

test\_paper\_name: 试卷名称。

question\_count: 试卷中的题目数量。

average\_difficulty: 试卷的平均难度。

update\_time: 试卷生成的时间。

username: 生成试卷的用户名。

User（用户表）

描述：该表存储系统中的用户信息，包含用户的基本资料及其权限配置。

功能：

存储用户的基本信息（如用户名、密码、角色等），管理用户权限。

记录用户的登录信息及活动状态。

字段说明：

**id:** 用户唯一标识符（主键）。

**username:** 用户名。

**password:** 用户密码。

**user\_role:** 用户角色（如教师、管理员等）。

**last\_login:** 用户上次登录时间。

**enable:** 用户启用状态。

**QuestionLabels**（题目标签表）

**描述:** 该表存储每道题目的标签信息，如题目的难度、知识点分类、题型等。

用于对题目进行分类检索。

**功能:**

对题目进行多维度的标签化管理，支持复杂条件下的检索功能。

提供灵活的标签化分类，帮助用户快速找到所需的题目。

**字段说明:**

**id:** 标签唯一标识符（主键）。

**question\_id:** 关联 QuestionBank 中的题目 ID。

**label\_type:** 标签类型（如难度、知识点、题型等）。

**label\_value:** 标签的具体值。

**Questiongenhistory**（题目生成历史表）

**描述:** 记录系统中每道题目的生成历史信息，追踪题目的生成时间及来源，便于题目的审查和追溯。

**功能:**

记录题目生成的来源、时间等详细信息。

支持题目历史的追踪和管理。

**字段说明:**

**id:** 记录唯一标识符（主键）。

**username:** 生成题目的用户名称。

**generated\_time:** 题目生成的时间戳。

**question\_id:** 关联 QuestionBank 中的题目 ID。

**source:** 题目生成来源信息（自动生成或手动创建）。

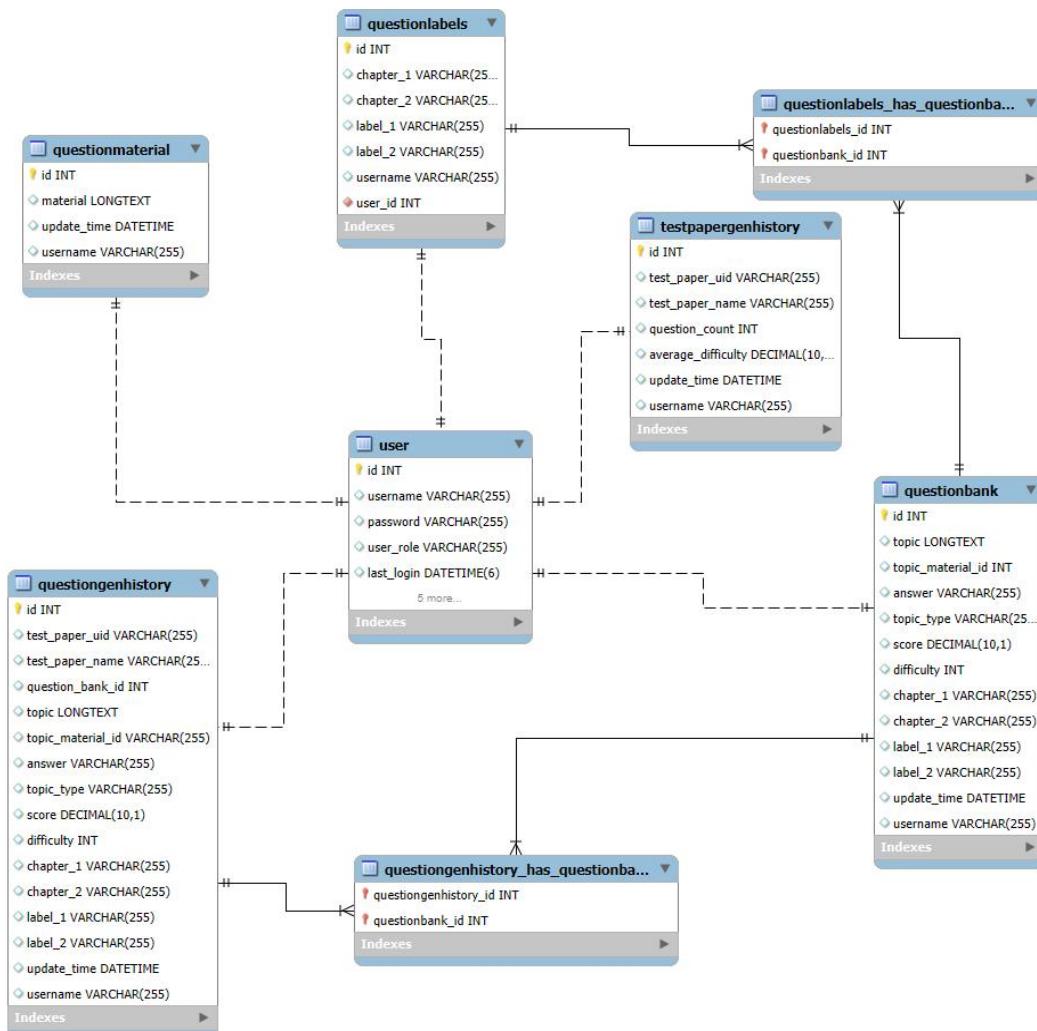


图 2-8 数据库关系结构图

题库管理模块通过以上六个表的相互配合，实现了题目的存储、更新、检索等核心功能。该模块为系统提供了强大的数据支持，并通过优化的数据库设计与表结构，确保了数据的高效存储与灵活检索。同时，借助题目生成历史与试卷生成历史记录，系统能够实现更透明的数据管理，方便后续的维护和审查。

### 2.3.4 算法组合试卷

#### (1) 贪心算法

贪心算法是一种优化算法，它在每一步决策中都选择当前状态下最优的选项，以求达到全局的最佳结果。这种方法通过局部最优解的连续选择，试图产生全局最优解，但不保证每次都能达到全局最优。贪心算法特别适用于问题可以分解为

能够通过一系列局部最优决策来解决的子问题。在实际应用中，贪心算法简单、直观且易于实现，虽然它不总是能解决所有问题，但在许多特定情况下提供了足够近似的优秀解决方案。先通过例题“零钱兑换”了解贪心算法的工作原理：给定目标金额，我们贪心地选择不大于且最接近它的硬币，不断循环该步骤，直至凑出目标金额为止。

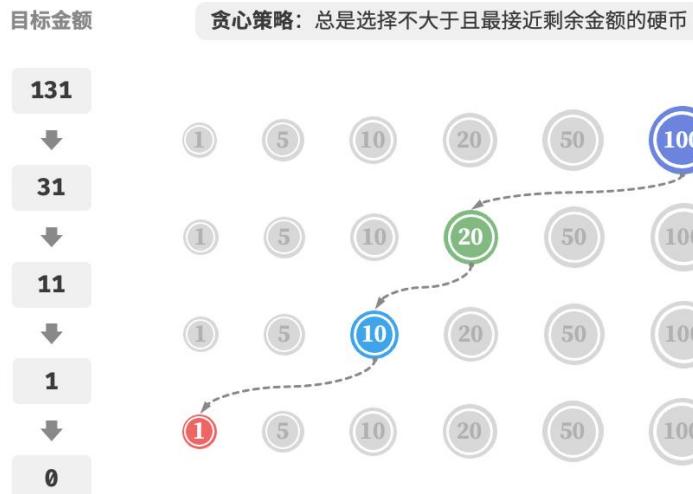


图 2-9 贪心策略图

贪心算法是一种优化算法，通过在每一步选择当前状态下的局部最优解，来期望得到全局最优解。在零钱兑换问题中，贪心算法的策略是每次选择面额最大的硬币，以尽快凑出目标金额。以下是贪心算法的公式推导过程：公式化表示如下：

初始化：设硬币数量  $count = 0$ 。

对所有  $c \in C$  (硬币集合) 按降序排列。

循环直到  $N = 0$ :

选择最大的  $c \leq N$ 。

更新  $N := N - c$ 。

增加硬币数量  $count := count + 1$ 。

每一步的贪心选择由以下表达式控制：

$$c = \max \{c_i \mid c_i \leq N\}$$

其中， $c_i$  表示当前可用的硬币面额， $N$  表示剩余的目标金额。

通过上述过程，贪心算法每次都选择当前面额最大的硬币，以减少总硬币数量，最终达到找零的目标。这种方法确保了在每一步中都做出局部最优的选择，从而在许多实际情况下能够有效地解决问题。



图 2-10 贪心局部最优解

在试卷组卷过程中，贪心算法被用于高效地选择最合适的话题，以确保试卷的结构合理性和科学性。

**高效的题目选择：**贪心算法通过逐步选取当前最佳题目，快速构建符合预设要求的试卷。这种方法在选择每一道题目时，都会优先考虑能最佳匹配当前试卷结构和难度要求的题目，从而大大提高组卷的效率。

**简单且实用：**由于贪心算法的简洁性，它在实际操作中具有很高的可实施性。通过逐步优化选择题目，可以在较短时间内生成一个符合要求的试卷，适用于要求快速组卷的场景。

**局部最优策略：**贪心算法采用的是局部最优的策略，即每一步选择当前最优解，不保证全局最优。然而，在组卷过程中，它能够快速地逼近全局最优，使试卷在内容覆盖和难度分布上较为合理。

## (2) 遗传算法

遗传算法的基本原理源于达尔文的自然选择和遗传学的基本概念。在遗传算法中，解决方案的每个实例被视为一个“个体”，整个解决方案空间形成一个“种群”。每个个体通过一串“基因”来表示，这些基因编码了解决方案的具体参数。遗传算法通过迭代过程，不断改进种群的质量，逼近最优解。其核心步骤包括选择（Selection）、交叉（Crossover）和突变（Mutation）。

考虑一个简化的遗传算法模型，其适应度函数  $f(x)$  用于评估每个个体的性能，其中  $c$  是一个编码了个体特征的向量。算法的目标是最大化适应度函数。遗传算法的一次迭代可以表示为以下步骤：

①选择：个体被选择用于繁殖的概率与其适应度成正比。如果我们设  $P_i$  是第  $i$  个体被选择的概率，则：

$$P_i = \frac{f(x_i)}{\sum_{j=1}^N f(x_j)}$$

其中， $f(x_i)$  为第  $i$  个个体的适应度函数值， $N$  为种群中个体的总数。

②交叉：选择的个体通过交叉操作生成新的后代。如果交叉点为  $k$ ，考虑两个个体  $x_i$  和  $x_j$ ，后代  $x_{new}$  可以表示为：

$$x_{new} = (x_{i1}, x_{i2}, \dots, x_{ik}, x_{j(k+1)}, \dots, x_{jm})$$

其中， $x_{ik}$  表示第  $i$  个个体的第  $k$  个基因， $x_{j(k+1)}$  表示第  $j$  个个体从第  $k+1$  基因开始的部分。

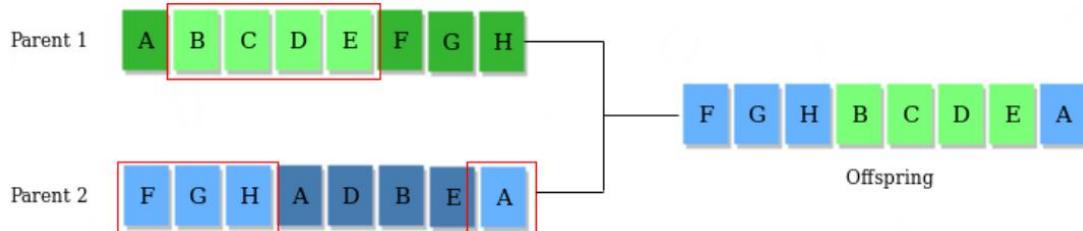


图 2-11 交叉算子

③突变：突变操作以小的概率  $\mu$  修改新生个体的某些基因，以增加种群的多样性。对于基因  $x_{nk}$ ，突变操作可以表示为：

$$x_{nk} = x_{nk} + \delta$$

其中， $\delta$  随机的小变化量， $\mu$  是突变率。

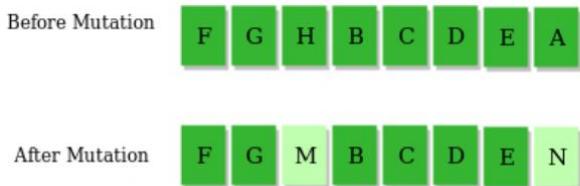


图 2-12 变异算子

④收敛分析：遗传算法的收敛性通常通过适应度函数在迭代过程中的变化来分析，收敛图展示了在每次迭代中种群中最优个体的适应度值。

以下是遗传算法收敛的示意图 2-13，图中显示了算法在多次迭代过程中的适应度值变化。随着迭代次数的增加，种群中最优个体的适应度值逐步提高，最终趋于稳定。这表明遗传算法在不断改进种群质量，逐步逼近最优解。

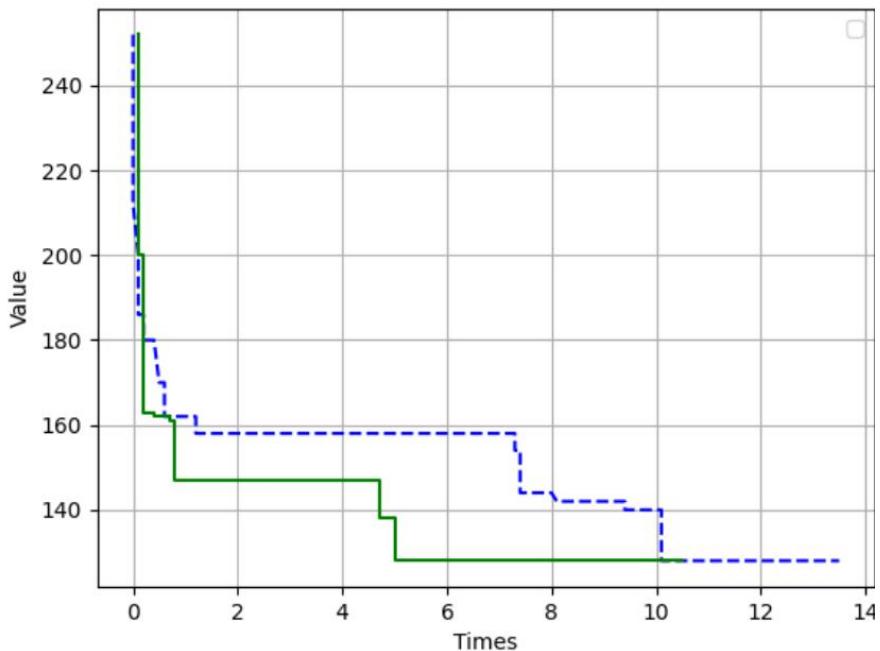


图 2-13 遗传算法收敛图

通过收敛图，我们可以观察到遗传算法在优化过程中是否稳定，并评估其收敛速度。收敛图有助于验证算法的有效性以及参数设置的合理性。

遗传算法是一种全局优化算法，用于在组卷过程中优化题目选择，确保试卷在各方面的全面性和合理性。

**全局优化：**与贪心算法的局部最优不同，遗传算法通过模拟自然选择过程，逐步优化题目选择，最终达到全局最优解。这使得试卷在题目质量、难度分布和

内容覆盖上更加科学合理。

**多样性和适应性：**遗传算法通过交叉和变异操作，生成多种试卷方案，并通过适者生存的原则选择最优方案。这种方法能够自适应地调整试卷结构，满足各种复杂的出题需求。

**平衡与优化：**在组卷过程中，遗传算法能够有效平衡试卷的难度、题型分布和知识点覆盖，确保试卷在各方面的均衡性。这种全局优化的能力使得遗传算法特别适用于生成复杂的试卷。

### 2.3.5 多租户模式

采用行级别隔离的多租户模式以实现数据的有效隔离和管理。这种查询方式确保了用户只能访问和操作其所属租户的数据，维护了数据的安全性和一致性。具体实现方案如下：

系统在单一数据库实例中部署多个 schema，每个租户被分配一个独立的 schema。确保了不同租户之间的数据在逻辑上完全隔离，降低了跨租户数据泄露的风险，同时简化了数据库管理。

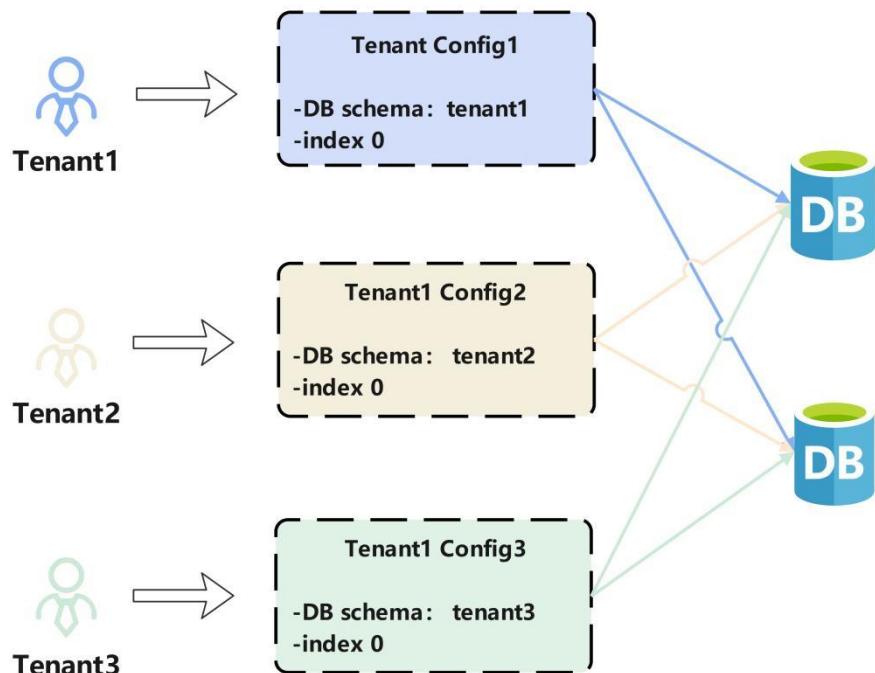


图 2-14 多租户模式架构图

在数据库表设计中引入了租户标识字段 `username`，用于标识每条数据记录的归属租户。在所有数据操作（查询、插入、更新、删除）中，通过附加租户标识来确保数据访问的范围被严格限制在当前租户的数据范围内。

在执行 SQL 查询时，系统通过在 WHERE 子句中加入租户标识条件来实现数据隔离。租户访问前端应用时，附带租户标识被系统读取，系统获取租户信息查询符合要求的后端服务和数据库数据，上传并显示在前端，过程如下图 3-6 所示：

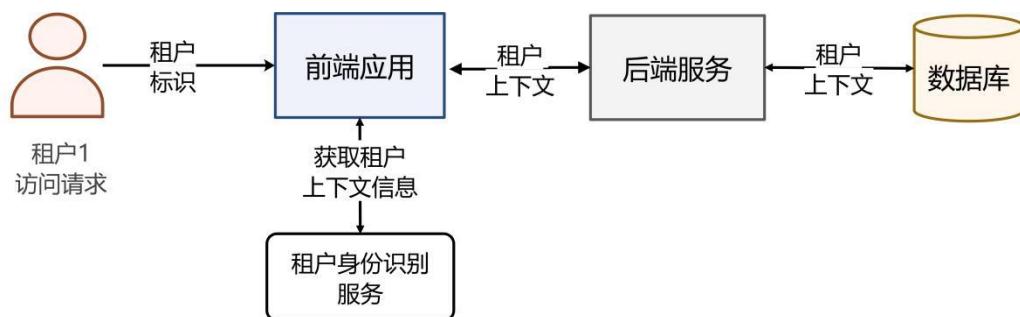


图 2-15 租户访问流程图

### 三、方案实现

#### 3.1 网页技术架构

##### 3.1.1 总体技术架构

本系统网站基于 Java 语言进行开发，采用 SpringBoot+React 的前后端分离开发模式。前端使用 React 框架，同时采用 Ant Design 组件库进行界面设计，进行响应式布局，适应不同大小的屏幕。后端基于 Spring Boot 框架开发，前端则使用 JavaScript 和 LESS 技术，以确保界面与功能的统一性。数据库采用 MySQL 关系型数据库，确保数据的稳定性和可扩展性。



图 3-1 网页技术架构

通讯层通过负载均衡服务器（Server Load Balancer, SLB）和内容分发网络（Content Delivery Network, CDN）实现请求的分发与优化，以确保系统的高可用性和低延迟。NGINX 作为反向代理服务器，进一步管理和分发来自用户的请求，提升系统的性能和安全性。

其次，表现层采用了 React 和 UmiJS 前端框架，用于构建用户界面和处理用户交互。通过 WebSocket 技术，表现层能够与服务层保持实时通讯，实现数据的实时展示与更新。这种设计不仅提升了用户体验，还确保了数据交互的高效性和及时性。

服务层是系统的核心部分，负责处理具体的业务逻辑。其功能模块包括用户管理、多智能体协同生成试题、题库管理和自动组卷。多智能体协同生成试题模块利用先进的人工智能技术，实现了高效、智能的题目生成，极大地提高了试卷制作的效率和质量。

数据层由腾讯云和 MySQL 数据库组成，提供了可靠的数据存储与管理服务。服务层与数据层之间通过数据库查询和存储操作进行交互，确保系统中的数据能够被高效地存取和处理。

### 3.1.2 前端与后端技术

将前端和后端的开发分离开来，使前端和后端可以独立开发、测试和部署。前端负责发送请求、处理数据、展示数据和用户交互，后端负责处理业务逻辑和数据存储。

在 UMI 项目中，前端使用 Mock 数据，使其可以独立于后端进行开发。UMI 基于路由，支持配置式和约定式路由，确保路由功能的完备性和可扩展性。前端使用 Ant Design 设计体系的 React UI 组件库 Antd，并通过 dva 处理数据流与交互操作。dva 内置了 react-router 和 fetch，进一步简化了开发流程。

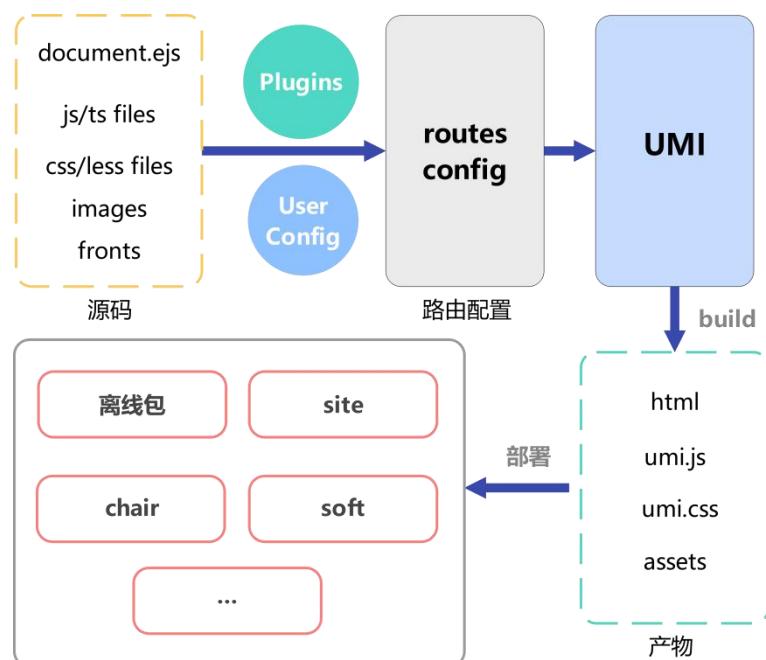


图 3-2 UMI 前端上线流程图

Umi 从源码到上线的流程如上图所示，umi 首先会加载用户的配置和插件，然后基于配置或者目录，生成一份路由配置，再基于此路由配置，把 JS/CSS 源码和 HTML 完整地串联起来。

后端采用 Spring Boot 框架，利用其自动配置和快速启动特性，极大地简化了 Spring 应用的开发过程。Thymeleaf 被选作模板引擎，因其高度集成与易部署特性，使得服务端渲染更为高效。

如图所示，整个系统的前后端交互流程是通过 Controller 层处理的。客户端发起的 HTTP 请求首先会被控制器捕获，该控制器负责将请求路由到适当的 Service 层（服务层）。在服务层，业务逻辑被处理，并且可能需要访问数据库中的数据，这时服务层会通过 Model 层 和 Repository 层（数据访问层）与数据库进行交互。处理完成后，结果数据会通过服务层返回给控制器，最后由控制器将响应发送回客户端。

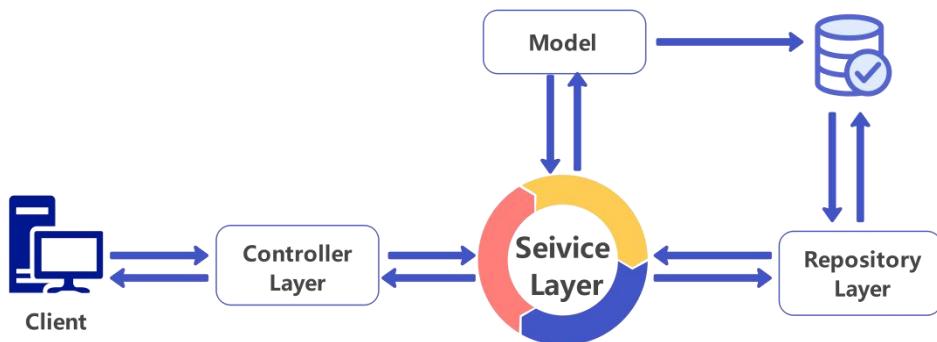


图 3-3 后端架构示意图

### 3.1.4 前后端交互

前端负责处理、展示数据和用户交互，后端负责处理业务逻辑和数据存储。前端使用 HTML、CSS、JavaScript 等技术实现用户界面和交互逻辑，并通过 WebSocket 等技术与后端通信，获取数据并将数据展示给用户。后端开发编写业务逻辑和数据存储的代码，提供 API 接口供前端调用。前端、后端及数据库各模块通过 API 接口通信，确保数据流动与功能协调，前后端交互图如下图所示。

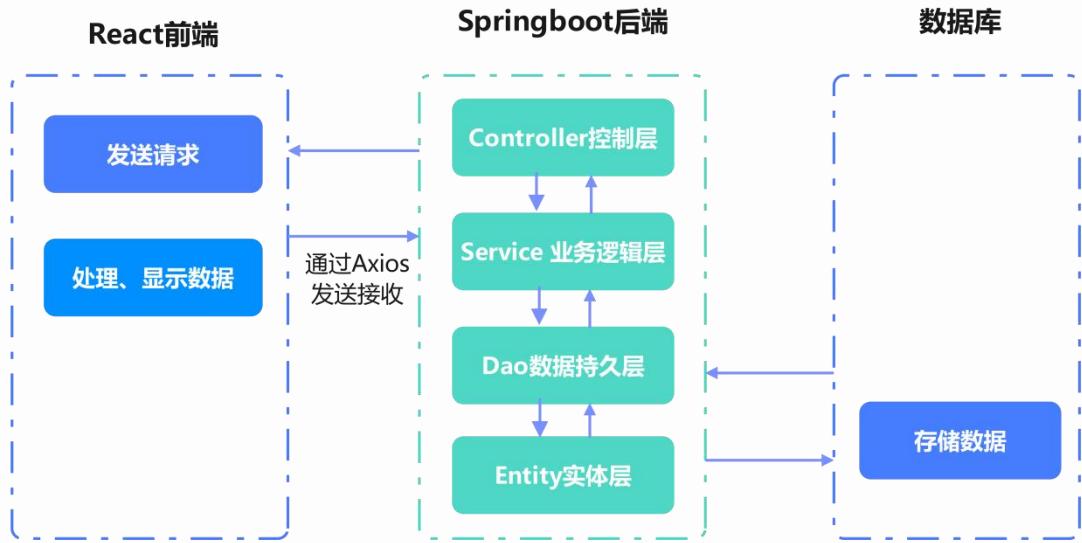


图 3-4 前后端交互图

### 3.1.5 环境部署

在本项目中，我们选择了阿里云的 ECS 作为主要计算资源，并使用宝塔面板进行部署与管理，以确保系统的稳定性和可扩展性。详细的部署方案如下：

我们采用了阿里云的 ECS 实例，并通过宝塔面板对实例进行管理。ECS 实例的高性能计算能力保证了应用在高并发场景下的稳定性。使用宝塔面板管理服务器资源，调整实例配置以应对业务需求的变化，确保系统的高效运行。

应用程序在本地完成构建和测试后，通过 Maven 打包为 JAR 文件。部署过程中，我们借助宝塔面板的文件管理功能，将 JAR 文件上传至服务器，并通过面板的命令行工具进行运行和管理。宝塔面板还支持计划任务的设置，用于定时执行脚本，如自动重启服务和清理日志。

在阿里云控制台中，我们配置了虚拟私有云（VPC），将服务器部署在同一网络内，通过 VPC 实现应用服务器与数据库服务器的安全隔离。为了确保用户数据传输安全，通过宝塔面板的防火墙功能配置安全组规则，限制对服务器的访问权限，并通过 SSL 证书管理工具，启用 HTTPS 加密通信。

为了实现高可用性，我们使用阿里云的负载均衡服务 SLB 进行流量分配，同时宝塔面板的监控工具用于实时监控服务器状态，及时发现并响应潜在的性能问题。

### 3.1.6 云数据库

本系统采用腾讯云的 MySQL 数据库作为后端数据存储解决方案，设计了合理的数据模型以满足各功能模块的需求。通过对系统需求的详细分析，数据库设计遵循了高内聚低耦合的原则，确保数据存储的完整性、一致性和高效性。利用腾讯云数据库提供的自动备份与手动备份功能，定期进行数据备份，并支持快速恢复，以防止数据丢失。

## 3.3 多智能体实现

多智能体采用 Python 语言进行开发，基于 MetaGpt 框架构建。MetaGpt 是一个利用 GPT 模型进行智能体开发的框架，支持 Windows 系统并兼容 Python 3.10 版本。该框架提供了强大的自然语言生成能力，特别适用于系统中的题目生成任务。通过 MetaGpt 框架，各个智能体可以根据预设的生成规则和算法独立生成题目，并通过框架内置的通信机制和任务调度功能，实现智能体之间的高效协作与任务执行。

### 3.3.1 提示词工程

在多智能体系统中，提示词工程（Prompt Engineering）是确保智能体能够高效、准确执行任务的关键技术手段。本系统使用智谱 GLM-4 大模型来配置提示词，并构建了适用于不同任务的提示词模板。这些提示词模板是智能体理解任务和生成输出的基础，决定了智能体如何接收输入、处理信息并生成符合预期的结果。提示词模板的设计包含多个关键要素，确保智能体的输出具有一致性、准确性和完整性。模板内容一般包括以下部分：

#### 提示词模板

#角色：定义智能体的身份和职责，使其能够根据设定的角色扮演相应的任务。

#语言：指定智能体生成内容的语言。

#描述：对任务进行详细说明，帮助智能体理解任务的具体要求和背景信息。

#目标：明确任务的最终目标，指导智能体朝着正确的方向生成内容。

##目标 1

##目标 2

# 约束：设置任务的特定限制条件，确保生成的内容符合预设标准。

**## 格式：**规定输出内容的结构和格式，确保文本具有统一的样式和便于理解的结构。

**## 示例：**提供正向示例，帮助智能体参考并生成符合预期的输出。

### 3.3.2 动作实现

在 MetaGPT 中，类 Action 是动作的逻辑抽象。用户可以通过简单地调用 self.\_aask 函数令 LLM 赋予这个动作能力，即这个函数将在底层调用 LLM api。

---

**代码：**实现智能体动作

---

```
from metagpt.actions import Action
1: class Judge(Action):
2:     PROMPT_TEMPLATE: str = """提示词"""
3:     name: str = "动作名字"
4:     async def run(self, instruction: str):
5:         prompt = self.PROMPT_TEMPLATE.format(instruction=instruction)
6:         rsp = await self._aask(prompt)
7:         code_text = Judge.parse_code(rsp)
8:         return code_text
```

---

(1) 类定义与继承

Judge 继承自 Action 类，这是动作的基类，提供了基础的功能和接口。

(2) 提示词模板

PROMPT\_TEMPLATE 字符串模板，用于生成发送给 LLM 的提示词。这个模板包含了指令的占位符 {instruction}，在运行时将被实际的指令替换。

(3) 动作的名字

name 定义了这个动作的名字，便于识别和调度。

(4) 异步运行逻辑

run 方法是动作的核心逻辑，使用 async 定义，支持异步调用。该方法接收一个指令 instruction，并基于提示词模板生成相应的提示词。

(5) 调用 LLM

prompt 是经过格式化后的提示词，传递给 \_aask 函数，后者负责与 LLM 交互并获取生成的响应 rsp。

(6) 解析响应

Judge.parse\_code(rsp) 是一个静态方法或类方法，用于从 LLM 的响应中提取和解析生成的代码文本。这个方法可以根据实际需要进行定制，确保提取到的代码符合预期。

### (7) 返回结果

生成的代码文本 `code_text` 会作为 `run` 方法的返回值, 被调用者接收和使用。

### 3.3.3 角色实现

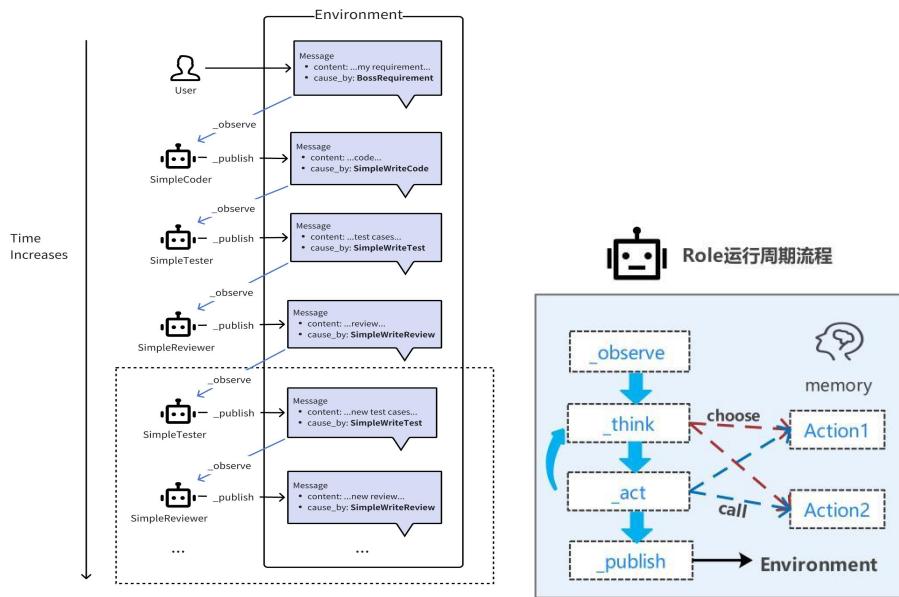


图 3-5 角色运行周期流程图

Role 类是智能体的逻辑抽象。一个 Role 能执行特定的 Action, 拥有记忆、思考并采用各种策略。Role 将从 Environment 中 observe Message。如果有一个 Role \_watch 的特定 Action 引起的 Message, 那么这是一个有效的观察, 触发 Role 的后续思考和操作。在 \_think 中, Role 将选择其能力范围内的 Action 并将其设置为要做的事情。在 \_act 中, Role 执行要做的事情, 即运行 Action 并获取输出。将输出封装在 Message 中, 最终 publish\_message 到 Environment, 完成了一个完整的智能体运行。

#### 代码：实现智能体角色

```
from metagpt.roles import Role
1: class SimpleCoder(Role):
2:     name: str = "Alice"
3:     profile: str = "SimpleCoder"
4:     def __init__(self, **kwargs):
5:         super().__init__(**kwargs)
```

---

```

6:         self.set_actions([SimpleWriteCode])
7:     async def _act(self) -> Message:
8:         logger.info(f'{self._setting}: to do {self.rc.todo}({{self.rc.todo.name}})')
9:         todo = self.rc.todo
10:        msg = self.get_memories(k=1)[0]
11:        code_text = await todo.run(msg.content)
12:        msg = Message(content=code_text, role=self.profile, cause_by=type(todo))
13:        return msg

```

---

### (1) 类定义与继承

SimpleCoder 继承自 Role 类，这是智能体角色的基类，提供了角色相关的基本功能。

### (2) 角色名字

name 和 profile 分别定义了这个角色的名字和个人简介，用于标识角色的身份和功能。

### (3) 初始化方法

`_init` 方法用于初始化角色实例。`super().__init__(**kwargs)` 调用基类的初始化方法。通过 `self.set_actions([SimpleWriteCode])`，为该角色设置了可执行的动作 SimpleWriteCode。

### (4) 角色执行逻辑的核心方法

- 1) `_act` 方法是角色的核心执行逻辑，它是异步方法，支持并发调用。
- 2) `logger.info(f'{self._setting}: to do {self.rc.todo}({{self.rc.todo.name}})')` 用于记录角色的执行情况。
- 3) `todo = self.rc.todo` 获取当前需要执行的动作。
- 4) `msg = self.get_memories(k=1)[0]` 从角色记忆中获取最近的一条消息。
- 5) `code_text = await todo.run(msg.content)` 调用动作的 `run` 方法，并传递消息的内容来生成代码文本。

## 3.3.4 记忆实现

Memory 类是智能体的记忆的抽象。当初始化时，Role 初始化一个 Memory 对象作为 `self.rc.memory` 属性，它将在之后的 `_observe` 中存储每个 Message，以便后续的检索。Role 的记忆是一个含有 Message 的列表。当需要获取记忆时（获取 LLM 输入的上下文）使用 `self.get_memories`。函数定义如下：

### 代码 :获取记忆函数

---

```
1: def get_memories(self, k=0) -> list[Message]:  
2:     """A wrapper to return the most recent k memories of this role, return all when k=0""""  
3:     return self.rc.memory.get(k=k)
```

---

使用 `self.rc.memory.add(msg)`添加记忆， `msg` 是 `Message` 的实例。在定义 `_act` 逻辑时将 `Message` 的动作输出添加到 `Role` 的记忆中。 `Role` 能记住它先前说过或做过什么，以便采取下一步的行动。

## 3.4 题目生成实现

### 3.4.1 文档读取与处理

本文采用 python 的 `pdfplumber` 库进行 PDF 文档的文本提取。该方法提供了一种高效的方式来处理不同格式的 PDF 文件，并从中提取文本内容。以下伪代码描述了如何读取 PDF 文件的所有页面并提取文本：

---

#### 算法 : 读取 PDF 所有页面的文本

---

**Input:** `pdf_path`  
**Output:** `text`

---

```
1: text ← ""  
2: try  
3: Open pdf_path with pdfplumber as pdf  
4: for each page in pdf.pages do  
5: page_text ← Extract text from page  
6: if page_text is not empty then  
7: text ← text + page_text.strip() + "\n"  
8: end if
```

---

### 3.4.2 摘要生成

在本系统中，摘要生成任务由一个专门配置的智能体角色 `CreateSummaryRole` 负责，该角色通过调用 `CreateSummary` 动作来实现对文本内容的知识点提炼和摘要生成。这一过程的实现涉及多个关键步骤，确保生成的摘要既全面又精准。以下是该过程的详细描述：

**(1) 角色与动作配置：**在系统的设计中，`CreateSummaryRole` 角色的核心任务是生成知识点摘要。该角色通过初始化时配置的 `CreateSummary` 动作来实

现其功能。`CreateSummary` 动作负责从用户提供的文本中提炼出关键信息，并生成符合要求的摘要。角色还监视与摘要生成相关的其他动作，如 `CheckSummary` 和 `Read`，以确保生成过程的连贯性和准确性。

**(2) 任务执行流程：**摘要生成的任务执行流程由 `CreateSummaryRole` 的 `_act` 方法控制。该方法首先记录当前待执行的任务，并从智能体的记忆中检索与任务相关的检查信息(`check`)。随后，从记忆中提取用户上传的文本内容(`msg`)，以便进行后续的知识点提炼。通过这种方式，系统确保了摘要生成过程能够根据最新的上下文信息进行优化和调整，从而提升了生成结果的相关性和质量。

**(3) 调用摘要生成动作：**通过其 `run` 方法，接收用户提供的文本内容和检查信息作为输入。`CreateSummary` 动作采用了预定义的提示词工程模板和规则，结合自然语言处理技术，对文本进行深入分析。具体而言，该动作通过文本分析和知识点提炼，将用户提供的内容转换为结构化的知识点摘要。

**(4) 生成和返回摘要：**摘要生成后，`CreateSummary` 动作将生成的内容封装到 `Message` 对象中，并返回给系统。这一过程不仅保证了摘要的完整性，还确保了信息的准确传达。通过将生成结果封装为 `Message` 对象，系统能够有效地处理和传递摘要内容，为后续的题目生成和知识点分析提供了可靠的基础。最终生成的摘要将被应用于后续的任务中，如题目生成和知识点评估，从而实现系统的整体功能目标。

### 3.3.3 题目生成与审查

**(1) 角色与动作配置：**`Teacher2` 角色负责生成题目和处理审查反馈。它通过初始化时配置的 `Judge` 动作来生成题目，同时监视 `Get` 和 `Checktest` 动作。`Judge` 动作负责根据摘要内容生成判断题，包括题目、选项、答案、解释等。

**(2) 任务执行流程：**`Teacher2` 角色的 `_act` 方法控制任务执行。该方法首先记录当前待执行的任务，然后从记忆中提取与任务相关的信息，包括检查结果、内容摘要、难度、题目数量及标签信息。

通过正则表达式从 `init` 中提取 `summary_content`（摘要内容）、`difficulty_content`（难度）、`num_content`（题目数量）、`label_1_content`（主章节标签）和 `label_2_content`（次章节标签）等信息。提取后的信息用于生成题目，

Judge 动作通过 run 方法生成题目，并将结果返回给系统。

**(3) 调用题目生成动作：**Judge 动作使用预定义的提示模板 (PROMPT\_TEMPLATE)，结合提取的内容生成判断题。提示模板中包含题目生成的详细要求和约束，如题目格式、难度定义、分数定义等。Judge 动作的 run 方法将提取的信息格式化到模板中，然后调用 \_aask 方法生成题目。

**(4) 生成和返回摘要：**生成的题目由 Judge 动作返回，并被封装到 Message 对象中。Teacher2 角色将 Message 对象返回给系统，确保题目内容的完整性和准确传达。生成的题目包括题目本身、答案、答案解释、题目类型、分数和难度等部分，并按照指定格式进行分隔和标记。

## 四、实现与验证

### 4.1 应用效果实现

#### 4.1.1 首页



图 4-1 首页

#### 4.1.2 登录页面

登录页面提供用户登录功能，确保只有授权用户可以访问系统。用户需输入用户名和密码进行身份验证，系统会根据用户角色分配相应的权限。登录页面支持注册功能，并可跳转至注册页面以新增用户。

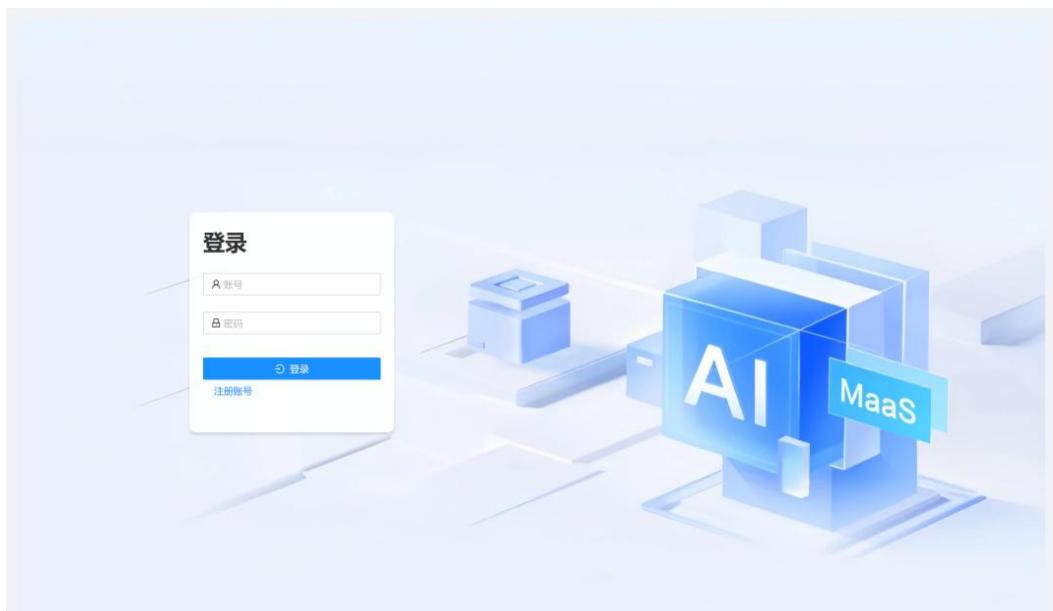


图 4-2 登录页面

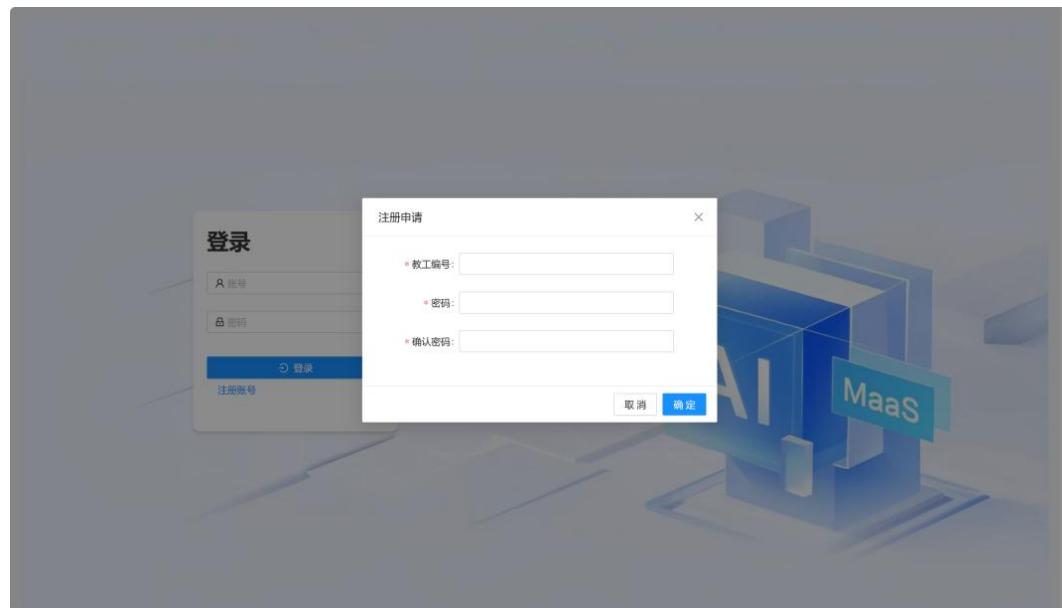


图 4-3 注册页面

### 4.1.3 生成题目页面

生成题目页面为教师提供了基于多智能体的自动化题目生成工具。用户可以上传教材或资料，系统会自动生成多种题型并按难度和章节分类。页面提供题目生成参数设置，包括题型选择、难度控制等，确保生成题目符合教学需求。

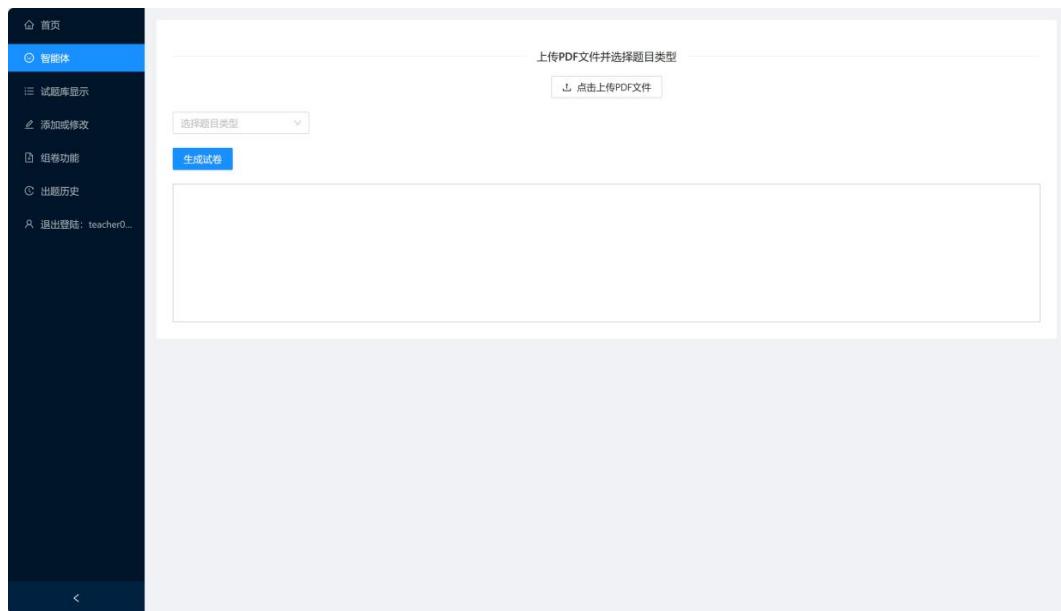


图 4-4 生成题目页面

#### 4.1.4 试题库页面

试题库页面展示和管理系统内所有的试题，教师可以在此查看、编辑和删除题目。页面支持批量导入、导出试题，并提供试题分布的可视化图表，帮助教师了解题目的知识点覆盖和难度分布，为试题选择提供数据支持。

#	题目	题目类型	分值	答案	大知识点	小知识点	大章节	小章节
0	若机器字长为16位，可表示的无符号数范围为_____。	填空题	1分	AAAAAwww	绪论	微型计算机发展概况	1	1.1
1	已知机器字长为16位，数x的补码为FFFEH，其真值是_____。	填空题	1分	AAAA	绪论	微型计算机系统概论	1	1.1
2	已知机器字长为16位，数N~1025，则其原码为_____，其补码为_____，其反码为_____。	填空题	3分	AAAA	绪论	微型计算机发展概况	1	1.1
3	微机硬件系统的基本结构由CPU、存储器、_____、外部设备以及系统总线等组成。	填空题	1分	AAAA	绪论	计算机中数和字符的表示	1	1.2
4	微机软件系统由两部分组成：_____，系统软件和应用软件。计算机软件可分为_____和_____。	填空题	3分	AAAA	绪论	微型计算机系统概论	1	1.1
5	空格符的ASCII码为_____，换行符的ASCII码为_____，ASCII码中，大写字母和小写字母的差值是_____。	填空题	3分	AAAA	绪论	微型计算机发展概况	1	1.1
6	把十进制数 65533 转换成二进制、八进制和十六进制。(1) 二进制：_____。(2) 八进制：_____。(3) 十六进制：_____。	填空题	3分	AAAA	绪论	计算机中数和字符的表示	1	1.2
7	已知机器字长为16位，最高位为符号位，分别用原码、反码、补码表示，求其真值。(1) [X]原=111110000B，其真值为_____。(2) [X]反=111110000B，其真值为_____。(3) [X]补=111110000B，其真值为_____。	填空题	3分	AAAA	绪论	微型计算机系统概论	1	1.1
8	计算机软件分为系统软件和应用软件，其中调试程序属于_____。	填空题	1分	AAAA	绪论	微型计算机发展概况	1	1.1
9	已知存储器容量为4GB，若用2 <sup>n</sup> 的方式表示大小，其值为_____B。	填空题	1分	AAAA	绪论	计算机中数和字符的表示	1	1.2

图 4-5 试题库页面

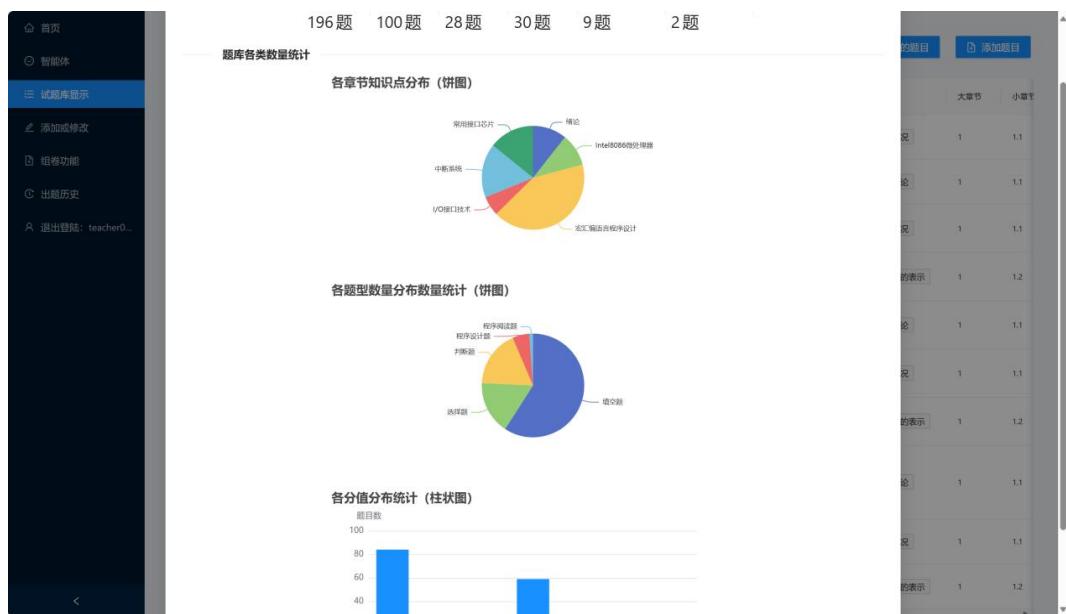


图 4-6 数据可视化页面

### 4.1.5 组卷页面

组卷页面为用户提供自动和手动组卷功能。通过选择题目难度、题型比例和知识点，用户可以自动生成试卷或手动选择试题进行组合。页面还支持试卷预览和导出，确保生成的试卷符合预期要求并易于分发。

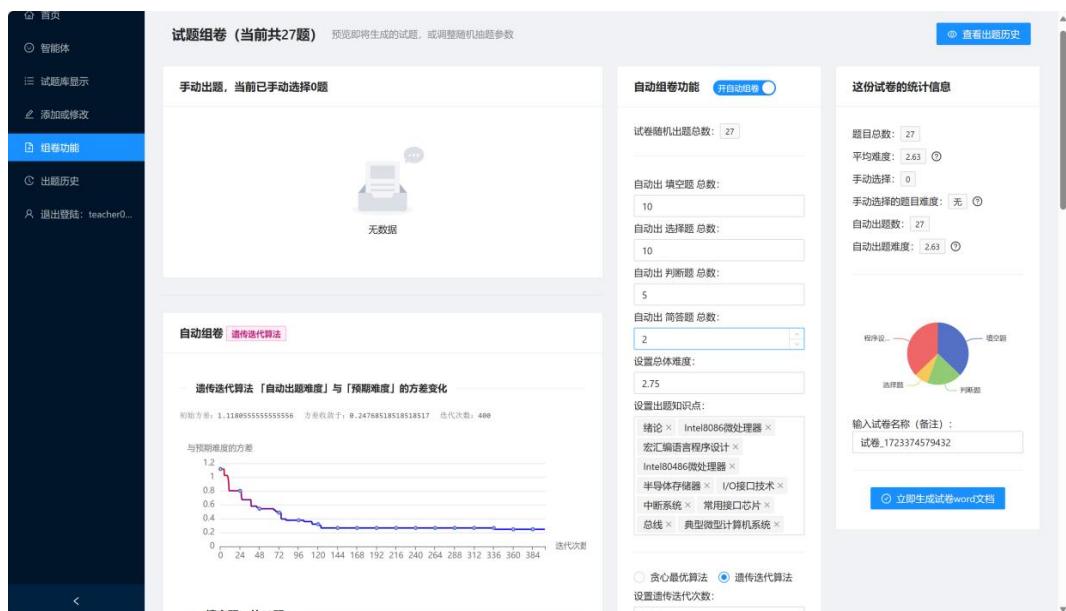


图 4-7 组合试卷页面

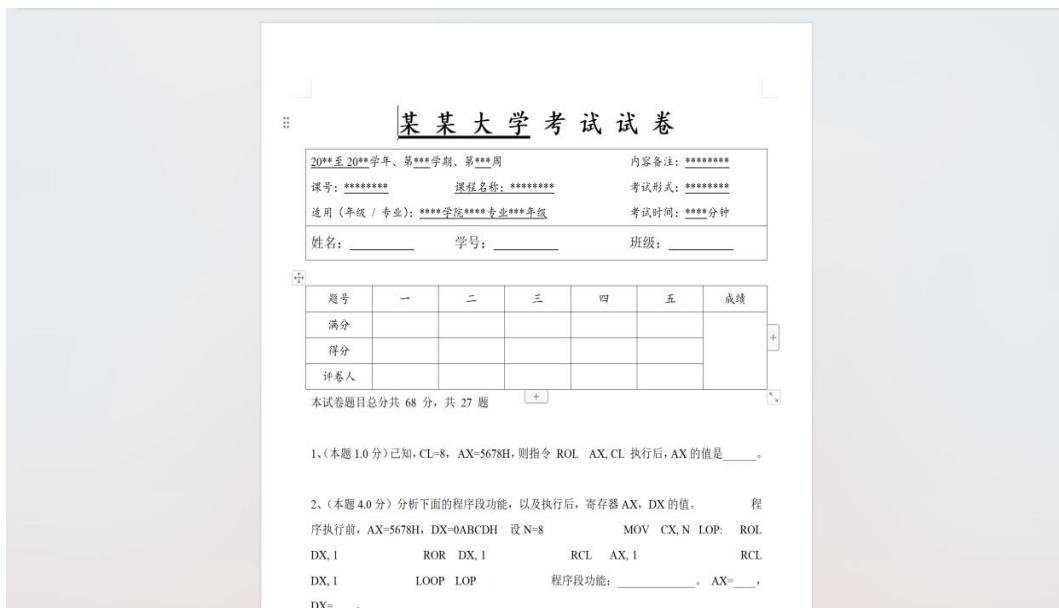


图 4-8 导出试卷格式

## 4.1.6 出题历史页面

出题历史页面记录了用户之前生成的试卷信息，提供试卷详情查看、修改和重新导出的功能。用户可以查看历史生成的试卷，并根据需要对试卷进行调整或复用，方便教师管理和复用高质量的试卷资源。

#	试卷名称	出题人	题目数
1	1	teacher001	27
2	试卷_1652792378265	teacher001	36
3	试卷_1652791845069	teacher001	30
4	试卷_1652791437359	teacher001	14
5	试卷_165279133674	teacher001	28

**《1》试卷出题历史记录**

**总体难度2.7**

**本试卷共27题**

**第1题 (1分)** 已知, CL=8, AX=5678H, 则指令 ROL AX, CL 执行后, AX 的值是\_\_\_\_\_。

**第2题 (4分)** 分析下面的程序段功能, 以及执行后, 寄存器 AX, DX 的值。程序执行前, AX=5678H, DX=0ABCDH, 设 N=8  
ROL DX, I ROR DX, I RCL AX, I RCL  
DX, I LOOP LOP 程序段功能: \_\_\_\_\_, AX=\_\_\_\_\_, DX=\_\_\_\_\_.  
DX=\_\_\_\_\_.

**第3题 (3分)** 把十进制数 65533 转换为二进制、八进制和十六进制。(1) 二进制: \_\_\_\_\_, (2) 八进制: \_\_\_\_\_, (3) 十六进制: \_\_\_\_\_.

**第4题 (6分)** 已知字符串数组 ARRAY, 从字符串首地址进入 MAX 字节单元中。DATA SEGMENT ARRAY DB 10E, 115, 210, COUNT EQU \$ - ARRAY DATA ENDS CODE SEGMENT ASSUME CS:CODE, DS:DATA START: MOV AX, DATA MOV DS, AX, 0 MOV AL, [BX] DEC CX AGAIN: CMP AL, [BX] JNE AGAIN MOV AH, 4CH INT 21H CODE ENDS END START

**第5题 (1分)** 宏指令的定义要用伪指令\_\_\_\_\_实现。

**第6题 (1分)** 8259P计数器, 其CLK2~2MHz, 若要实现OUT2输出信号频率为110Hz, 则计数初值应设为\_\_\_\_\_。

**第7题 (6分)** 在数据BUF中, 存放着20个字, 把其中的偶数累加求和, 放放到变量SUM中(不考虑溢出)。DATA SEGMENT BUF DW 690,121,488,...,711 SUM DW ? DATA ENDS CODE SEGMENT ASSUME CS:CODE, DS:DATA START: MOV AX, DATA MOV DS, AX, 0 MOV AH, 4CH INT 21H CODE ENDS END START

**第8题 (1分)** 假设DISP=50H, 指令存放在代码段地址为200H的两个字节单元中, 则指令IMPSHORT DISP的有效物理地址为: \_\_\_\_\_。

**第9题 (2分)** 在中断向量表中, 从偏移地址为100H开始的8个字节单元, 其存放的数据依次为00H, 34H, 20H, 50H, 60H, 00H, 30H, 若执行指令INT 41H, 则对应的中断服务程序入口地址为(CS:IP): \_\_\_\_\_, 或其物理地址为: \_\_\_\_\_。

**第10题 (3分)** 已知机器字长为16位, 基址位为位寻址, 分别用原码、反码、补码表示, 求其真值。(1) [X原]=11111000B, 其真值为\_\_\_\_\_。(2) [X反]=11111000B, 其真值为\_\_\_\_\_。(3) [X补]=11111000B, 其真值为\_\_\_\_\_。

**第11题 (2分)** 已知主片ICW3=14H, 则该中断系统最多可接的请求数目是\_\_\_\_\_. (1) 14H; (2) 22; (3) 16; (4) 36。

图 4-9 历史试卷页面

## 4.2 数据验证

为了验证多智能体系统在生成题目、优化试卷组合和提高题目质量方面的效果，我们进行了多次实验，并记录了不同实验条件下的性能数据。实验主要包括以下几方面：多智能体生成题目轮数、智能体数量对题目生成的影响、题目章节分布、布鲁姆分类法难度分布以及进化算法的优化结果。

在不同轮数的实验中，我们评估了多智能体系统在题目生成速度、题目质量和综合评价三个方面的表现。表 4-1 显示了各轮次实验的具体数据：

表 4-1 多智能体不同生成轮数表现

多智能体生成题目轮数	布鲁姆分类法难度分布	题目总数	平均生成时间(秒)	题目质量评分(1-10)	综合评价
1	7:23:35:17:11:4	97	34	5.3	12.69
2	9:21:31:26:14:3	104	57	8.7	14.38
3	3:19:20:24:6:1	73	82	9.6	14.26

结果表明，随着生成轮数的增加，虽然题目的质量评分和综合评价有所提升，但同时也带来了生成时间的显著增加。这表明在实际应用中，可能需要在题目质量和生成时间之间做出权衡。

为了评估智能体数量对题目生成效果的影响，我们进行了单智能体和多智能体（3 个和 8 个智能体）实验。结果如下表 4-2 所示：

表 4-2 不同数量多智能体表现

生成题目智能体的个数	布鲁姆分类法难度分布	题目总数	平均生成时间(秒)	题目质量评分(1-10)	综合评价
单智能体	5:23:45:10:3:1	87	29	3.1	12.5
多智能体(3 个)	3:26:29:17:5:5	85	56	8.3	16.91
多智能体(8 个)	9:34:26:19:4:2	94	90	9.7	17.83

实验数据表明，多智能体系统显著提高了题目质量评分和综合评价分数，尤其是在使用 8 个智能体时，生成的题目质量评分达到了 9.7 的高分，综合评价为 17.83。这说明多智能体系统在复杂题目生成中的优势明显，但也需要更多的时间成本。

为验证遗传算法自动组卷的可行性和有效性，以《操作系统》课程为例，进行组卷实验。本次实验的试题库中共有 207 道题，试题库中各章节的题量分布情况见表 4-3，难度分布情况见表 4-4。

表 4-3 题量分布情况

章节	第一章	第二章	第三章	第四章	第五章
题目	34	57	67	30	19

表 4-4 难度分布情况

布鲁姆分类法难度分布	知道	领会	应用	分析	综合	评价
题目	57	47	55	19	23	6

实验要求为组一份试卷，利用遗传算法进行组卷，进行了 100 代的进化运算运算，各进化代数下的传统遗传算法种群中最优适应度如表 4-7 所示。其中，各题型题目数分数要求见表 4-5，各章节分数要求见表 4-6，试卷的总分为 100 分，难度系数为 0.6，实验算法采用 C 语言环境编程实现，效果如表 4-7 所示。

表 4-5 各题型题目数分数

题型	单选题	填空题	判断题
题量	20	20	20
每小题分值	2	2	1

表 4-6 各章节分数

章节名称	第一章	第二章	第三章	第四章	第五章
分数	17	28	32	13	10

表 4-7 遗传算法种群中最优适应度

进化代数	传统遗传算法种群中最优适应度
第 0 代	0.07903
第 10 代	0.07901
第 20 代	0.08656
第 30 代	0.09071
第 40 代	0.09874
第 50 代	0.10153
第 60 代	0.10153

实验结果表明，遗传算法在优化题目选择过程中表现出良好的效果。在相对较少的代数内，算法能够逐步提升种群的适应度，并最终趋于稳定。与组卷相比，遗传算法在较早的进化代数内就达到了更高的适应度，证明了其在题目优化和试卷生成中的优越性。

## 五、创新特色

### 5.1 多智能体协同生成试题

引入多智能体协同工作的创新机制，这些智能体各自职责明确，高效协作。它们不仅深度参与题目的生成和筛选流程，还共同调控题目的难易程度，确保试卷的整体质量和评估效果。在题目生成阶段，处理智能体组读取传入的 PDF 内容，处理并概括大小章节知识点后传递给出题智能体组。出题智能体凭借其专业知识，创作出符合教学要求的题目并传递给审查智能体组，审查智能体分别从答案合理性、难度合理性、知识点来源等方面进行多轮循环审查，对题目进行优化调整后存入数据库，这些题目覆盖了广泛的题型和不同的难度层次，知识点来源于不同章节。

### 5.2 布鲁姆分类法定义试题难度

在传统的试题生成系统中，题目难度通常通过简单的指标（如题目正确率或专家评分）来定义。然而，这种方法往往缺乏对学生认知水平的精确测量，无法全面评估学生在不同认知层次上的表现。为了克服这一局限，我们在多智能体系统中创新性地采用布鲁姆分类法，将每个题目对应的认知层级作为难度判定的核心标准。每个智能体在生成题目时，会根据题目的内容和要求，自动判断其所属的布鲁姆层级。系统会根据这些分类，生成符合指定认知层级的题目，确保试题难度的科学性和合理性。

### 5.3 贪心与遗传算法最优化组卷

在传统的组卷系统中，题目的选择通常依赖于固定的规则或人工选择，这导致效率低下和结果质量的不可控性。本项目创新性地引入了贪心算法与遗传算法相结合的智能组卷方法，极大地提高了组卷过程的效率与质量。

贪心算法特别适用于处理大规模题库时的初步筛选，能够迅速为题目集合找到一个较优解。为了进一步优化试卷的质量，并考虑更复杂的目标（如题目难度分布、知识点覆盖率等），本项目在贪心算法的基础上，引入了遗传算法。遗传

算法是一种基于自然选择和遗传机制的优化算法，它通过模拟自然进化的过程，对初始题目集合进行全局优化。

贪心算法与遗传算法的结合创新，充分利用了两者的优势，使得系统既能够快速生成一个较优解，又能够通过进一步优化提升试卷的整体质量。最终生成的试卷不仅在题目质量和难度平衡上表现优异，还能够更好地满足考试的特定需求，如知识点覆盖、题型分布和考核目标的全面性。

## 5.4 试题库管理与可视化

试题管理与可视化不仅实现了试题的集中存储与高效管理，还通过可视化界面为用户提供了直观、便捷的操作体验。题库管理系统支持从试题的题型、难度、知识点、所属章节多维度检索，使得出题人员能够迅速定位所需试题，提升工作效率。同时，系统提供了试题统计与分析功能，包括试题总数、不同题型试题数量占比等情况反馈，为题库的优化与调整提供了数据支持。在可视化方面，系统通过饼图、柱状图等形式直观展示题库的状态，如试题数量、类型分布、难度分布等，帮助用户快速了解题库的整体情况。此外，组卷过程中的题目选取、不同题型数量占比、历史试卷数据详情等，也均支持可视化展示，使得整个组卷过程更加透明、可控。

## 5.5 行级别隔离的多租户模式

多租户安全隔离是云计算环境下的一种重要安全策略，它旨在确保在共享资源的同时，不同租户之间的数据和操作保持独立且互不干扰。在智能试卷生成系统中，它关乎到题目的保密性和隐私保护。本系统采用单数据库、行级别隔离的多租户方案。由于所有租户的数据共享一个数据库实例，系统可以更好地利用数据库的计算和存储资源，避免了多实例部署时可能出现的资源浪费，减少运维成本和复杂度。每个用户的数据严格受到限制，只有经过验证的用户才能访问与其关联的数据。这种机制有效防止了数据泄露和越权访问，确保了每个租户数据的安全和隐私。

## 六、总结与展望

### 6.1 工作总结

随着人工智能和教育技术的迅速发展，教育行业面临着越来越多的新挑战和机遇。智能题库系统、自动化组卷工具以及基于人工智能的教育应用，正在逐步改变传统的教育模式。这些新技术不仅提高了教育的效率，也使得个性化学习和智能化教学成为可能。然而，在这些技术的应用过程中，如何优化智能题库的生成、确保系统的高效性和稳定性，以及如何在多用户环境中提供高质量的服务，成为亟待解决的问题。

为了应对这些挑战，我们设计并实现了一个基于多智能体的在线组卷系统，并开发了一系列关键模块和功能。项目完成的主要工作成果总结如下：

一、开发了基于 Python 语言的多智能体模块，使用 MetaGPT 框架来实现题目生成任务。每个智能体专注于特定类型题目的生成与审查，确保生成的题目种类丰富、难度适中，满足了不同用户的需求。通过前端界面，用户可以便捷地运行多智能体团队，实现个性化题目生成。

二、采用多租户方案（单数据库，行级别隔离）来管理用户数据。通过在数据库表中添加用户列，确保了用户数据的隔离性和安全性。这种方法不仅简化了数据库的管理和维护，还能有效应对大量用户的高并发访问需求。

三、利用阿里云 ECS 与腾讯云 MySQL 数据库相结合，构建了高性能的云端部署环境。通过 JAR 文件打包和 DevOps 工具的持续集成与部署（CI/CD）服务，确保了系统的稳定性和可扩展性。

四、通过前后端分离的开发模式，使用 SpringBoot 框架实现后端逻辑，React 框架实现前端界面设计。结合 Ant Design 组件库，我们打造了一个简洁、响应式的用户界面，提供了良好的用户体验。

我们相信，通过这一系列的工作成果，我们不仅在教育领域实现了智能化和自动化的创新，还为未来的教育技术发展奠定了坚实的基础。在未来的发展中，我们将继续探索和优化多智能体技术，推动教育行业的智能化变革。

## 6.2 未来展望

一、我们计划对现有的智能体模块进行全面的优化和升级，以提升题目生成的质量和多样性。具体而言，我们将研究更加先进的题目生成算法，确保智能体能够生成更加符合教育标准、难度适中且多样化的题目。此外，还将引入上下文理解能力，使得智能体能够在生成题目时更好地理解并保留题目的语境，从而提高题目与实际课程内容的相关性。未来的优化还将包括对智能体生成速度的提升，减少用户等待时间，并且支持更大规模的数据处理与题目生成。

二、在多智能体系统中，智能体之间的协作与信息交互是生成高质量题目的关键。为此，我们计划引入多智能体内部交互的可视化工具，帮助开发者和用户直观地了解智能体之间的沟通与合作过程。通过可视化界面，用户可以观察到智能体如何分配任务、交换信息、协作生成题目。此外，用户还可以通过该工具实时监控智能体的状态与工作流程，确保每个智能体都在有效地执行任务。

三、随着自然语言处理技术的进步，预训练大模型在各类生成任务中表现出色。我们计划引入私有大模型，进一步增强系统的题目生成能力。通过定制和优化大模型，使其能够更好地理解教育领域的特殊需求，并在此基础上生成更加符合要求的题目。此外，私有大模型的引入还将支持个性化题目生成，即根据不同的用户的学习进度、学习风格和需求，生成专门适配的题目，从而提供更加个性化的学习体验。通过定期更新和训练私有大模型，我们将确保其持续提供高质量的题目生成服务。

通过以上几个方面的详细规划和实施，我们期待在未来的版本中，为用户提供一个更加智能、灵活且高效的在线组卷系统。这个系统不仅能够适应不断变化的教育需求，还能通过先进的技术手段，不断为用户带来全新的题目生成体验。我们相信，通过持续的创新与优化，系统将在教育领域中发挥更大的作用，助力用户实现更高效的教学与学习目标。

## 参考文献

- [1]张杰.基于 React+Spring 的教学系统设计与实现[D].山东师范大学,2019.
- [2]席素梅,赵红艳,张成强,等.基于布鲁姆目标分类法的自适应学习系统研究与应用[J].长江信息通信,2023,36(11):44-47.
- [3]张亚妮.对传统几种组卷算法改进与比较分析[J].计算机与数字工程,2017,45(12):2364-2367+2397.
- [4]杜纪龙,李新峰,何岩峰.基于 SpringBoot+React 的智慧农业系统设计与实现[J].智慧农业导刊,2024,4(14):17-20.DOI:10.20028/j.zhnydk.2024.14.005.
- [5]党小娟,郝建军,张瑜,等.试卷生成系统在教学中的应用[J].企业科技与发展,2018(09):285-286.
- [6]陈慧.基于多模态融合的智能组卷算法研究与实现[D].西安电子科技大学,2023.DOI:10.27389/d.cnki.gxadu.2023.002508.
- [7]陈烨烨,李捍东.基于贪心混合定位算法三阶段排样问题研究[J].机械与电子,2024,42(03):12-16+25.
- [8]周波,杨智昉,汪慧菁.基于布鲁姆教学目标分类整合课程教学模式的探索与实践[J].中国继续医学教育,2024,16(04):11-15.
- [9]翟雪松,季爽,焦丽珍,等.基于多智能体的人机协同解决复杂学习问题实证研究[J].开放教育研究,2024,30(03):63-73.DOI:10.13966/j.cnki.kfjyyj.2024.03.007.