**武汉大学**

**计算机学院实训**

**基于AI Agent的智能学伴系统设计说明书**

**项目成员： 李超然 丁欣雨 刘子俊**

**撰 写 人：李超然 丁欣雨 刘子俊** 2025年7月6日

**评 审 人：**  年 月 日

武汉凡诺软件技术有限公司

2025年

**修订记录**

| 日期 | 修订版本 | 修改章节 | 修改描述 | 修订人 |
| --- | --- | --- | --- | --- |
| 2025.7.6 | v1 | 全文 | 初始文档 | 李超然 丁欣雨 刘子俊 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目 录

[1. 简介 1](#_Toc6698)

[1.1 目的 1](#_Toc4522)

[1.2 范围 1](#_Toc26915)

[一、软件名称 1](#_Toc21689)

[二、软件功能 1](#_Toc30755)

[三、软件应用 1](#_Toc26321)

[2. 第0层设计描述 1](#_Toc11701)

[2.1 软件系统上下文定义 1](#_Toc1841)

[2.2 设计思路 1](#_Toc31603)

[一、设计可选方案 1](#_Toc15102)

[二、 设计约束 2](#_Toc3097)

[三、遵循标准 2](#_Toc31621)

[四、硬件限制 2](#_Toc3899)

[五、技术限制 2](#_Toc20378)

[六、其他 2](#_Toc30628)

[3. 第一层设计描述 2](#_Toc28930)

[3.1 系统结构 2](#_Toc27043)

[一、系统结构描述 2](#_Toc26113)

[二、业务流程说明 2](#_Toc16596)

[3.2 分解描述 2](#_Toc30853)

[一、模块1/子系统1描述 3](#_Toc4393)

[二、模块2/子系统2描述 3](#_Toc32647)

[3.3 依赖性描述 3](#_Toc25918)

[3.4 接口描述 3](#_Toc21656)

[一、模块1/子系统1的接口描述 3](#_Toc19513)

[二、模块2/子系统2的接口描述 3](#_Toc24249)

[4. 第二层设计描述 4](#_Toc7111)

[4.1 模块1名称 4](#_Toc9439)

[一、模块设计描述 4](#_Toc12029)

[1、类名 4](#_Toc11749)

[2、类名 4](#_Toc28951)

[二、功能实现说明 5](#_Toc24755)

[4.2 模块2名称 5](#_Toc15222)

[一、模块设计描述 5](#_Toc28380)

[1、类名 5](#_Toc2123)

[类名 5](#_Toc25868)

[二、功能实现说明 5](#_Toc18380)

[5. 数据库设计 5](#_Toc6134)

[5.1 实体定义 5](#_Toc21482)

[一、分解描述 5](#_Toc6564)

[二、内部依赖性描述 5](#_Toc7906)

[5.2 行为定义 6](#_Toc599)

[一、分解描述 6](#_Toc23323)

[二、外部依赖性描述 6](#_Toc29017)

[三、内部依赖性描述 6](#_Toc3886)

[6. 原型或界面设计 6](#_Toc3441)

关键词：AI Agent,知识图谱,ReAct,CDM

摘 要：EduAssistant 是一款基于AI Agent的智能教育辅助系统，通过大语言模型、智能体和知识图谱技术，为学生提供个性化学习体验，为教师提升教学效率。系统核心功能包括智能问答、个性化学习路径规划、知识掌握度评估和学习内容管理，采用分层架构设计，以ReAct Agent为核心，结合知识图谱构建、掌握度评估、学习路径生成等关键算法，实现了教育资源的智能推荐和学习过程的精准指导。EduAssistant 致力于通过技术创新推动教育个性化和智能化发展，为学生提供全方位的学习支持，为实现教育公平和提升教育质量贡献力量。

缩略语：无

# 简介

## 目的

本文档旨在清晰、全面地定义和描述AI Agent智能学伴系统的整体架构、关键组件及其交互方式。其主要目的是：

1. **提供设计蓝图：** 作为系统实现的权威技术参考，详细阐述系统的设计决策、技术选型、模块划分、数据流、接口规范以及非功能性需求的设计方案。
2. **促进沟通与共识：** 确保项目团队（包括开发、测试、运维等）对系统设计有统一、深入的理解，为后续的开发、测试、部署和维护工作奠定坚实基础，减少歧义和返工。
3. **支持评审与决策：** 为评审老师提供评估设计方案可行性、一致性、满足需求程度以及潜在风险的依据，辅助关键决策。
4. **指导实现与验证：** 为开发团队提供明确的编码指导，并为测试团队制定测试策略和用例提供设计依据。

本文档主要面向以下人员：

* **系统架构师与设计师：** 细化设计或评审设计。
* **软件开发工程师：** 理解整体架构和模块职责以进行具体实现。
* **质量保证/测试工程师：** 依据设计制定测试计划和验证系统行为。
* **运维工程师：** 了解系统架构以规划部署、监控和运维方案。
* **技术负责人：** 把控技术方向、进度和风险。
* **评审人：** 需要理解系统技术方案的技术型干系人。

## 范围

一、软件名称

基于AI Agent的智能学伴系统

二、软件功能

本系统采用分层架构设计，核心功能分为**学生端**与**教师端**两大模块，具体功能如下：

#### (1) 学生端功能

1. **学情自检与监控**
   1. **知识点掌握度分析**：基于CDM算法生成个人学习报告，可视化展示知识点掌握程度。
   2. **学情历史追溯**：支持查看历史作业成绩、学习进度趋势及薄弱知识点预警。
2. **智能知识检索**
   1. **语义化搜索**：通过自然语言处理技术，支持对知识库进行智能语义检索，精准定位学习资源。
   2. **关联知识推荐**：根据检索内容自动推送相关知识点、习题及课程资源。
3. **课程交互管理**
   1. **加入课程**：通过课程代码或邀请链接加入教师创建的课程。
   2. **作业与资源**：在线提交作业、访问教学资源（课件/视频/习题库），接收AI批改反馈。

#### (2) 教师端功能

1. **学情全景监控**
   1. **实时进度追踪**：动态监控班级整体学习进度、作业完成率及成绩分布。
   2. **知识点掌握分析**：基于CDM算法可视化全班知识点掌握热力图，定位共性薄弱环节。
2. **智能知识库建设**
   1. **知识图谱管理**：上传/编辑课程知识图谱，构建多层级知识结构。
   2. **CRUD操作**：支持知识条目的增删改查，关联习题、案例等教学资源。
   3. **语义化检索引擎**：提供知识库的智能语义搜索与内容管理功能。
3. **课程全周期管理**
   1. **课程创建**：设计课程结构，关联知识图谱，上传教学资源。
   2. **作业智能管理**：AI智能出题、AI批改客观题，手动批阅主观题。
   3. **资源统筹**：统一管理课件、视频、题库等资源，设置学生访问权限。

三、软件应用

本系统聚焦教育数字化赋能场景，主要服务于以下三大领域：

#### (1) 高等教育机构

* **精准化教学管理**

适用于高校专业课教学，教师通过知识图谱构建系统化课程框架，利用AI智能出题/批改实现大规模作业高效处理。

* **学情深度诊断**

基于CDM算法的学生知识点掌握度分析，为分层教学、补强辅导提供数据支撑，尤其适用于数学、工程等逻辑链紧密的学科。

#### (2) 职业培训与继续教育

* **技能掌握度量化评估**

在IT技能培训、资格考试辅导等领域，通过学情图直观展示学员技能薄弱点，定向推送学习资源。

* **弹性知识库建设**

支持企业内训师快速构建行业知识体系，结合语义检索实现知识高效复用。

#### (3) K12智能化教学辅助

* **个性化学习路径**

中学教师通过班级知识点掌握图定位教学盲区，动态调整授课重点。

* **减负增效实践**

# AI批改自动处理客观题与标准化作业（如单词听写、公式推导），释放教师精力至创造性教学活动。

# 第0层设计描述

## 2.1 软件系统上下文定义

EduAssistant 是一个基于 AI Agent 的智能教育辅助系统，旨在为教师和学生提供个性化学习支持。系统与以下外部实体交互：

用户（教师、学生）：通过Web界面访问系统。

第三方AI模型（DeepSeek、Google Gemini、GPT）：用于智能问答、作业批改、题目生成等。

向量数据库（Chroma）：存储知识库的语义向量，支持语义搜索。

图数据库（Neo4j Aura）：存储知识图谱，支持知识关联查询。

文件存储（阿里云OSS）：存储课程资源（PDF、PPT、视频等）。

搜索引擎（Google Search API）：用于增强AI Agent的外部知识检索能力。

## 2.2 设计思路

一、设计可选方案

|  |  |  |  |
| --- | --- | --- | --- |
| **方案** | **描述** | **优缺点** | **选择依据** |
| 单体架构 | 所有功能集中在一个服务 | 开发简单，但扩展性差 | 不适用，系统需模块化 |
| 微服务架构 | 按功能拆分为独立服务（如用户服务、课程服务、AI服务） | 扩展性强，但运维复杂 | 适用于未来扩展 |
| 分层架构（当前选择） | 前后端分离，后端按模块分层（Models-Services-Views） | 平衡开发效率和可维护性 | 适合中小规模系统 |

1. 设计约束

本系统的设计需满足以下约束条件：

功能性约束：本系统需实现教师与学生的核心功能差异化，包括课程管理、作业批改、智能问答等，并确保AI功能深度集成RAG技术，结合本地知识库与第三方大模型生成精准回答。同时，系统必须支持多格式课程资源（如PDF、视频）的上传、解析与存储，通过OCR和分段处理实现可检索的文本化存储，满足教育资源结构化管理的需求。

非功能性约束：系统需保证高性能（API响应≤200ms，数据库查询≤50ms）与高安全性（RBAC权限控制、数据加密），同时具备可扩展性以应对未来微服务化需求。此外，需优化资源占用（如Chroma向量数据库的内存管理），并适配低配置设备，确保流畅的用户体验。

合规性约束：系统需严格遵守《个人信息保护法》等数据隐私法规，确保用户教育数据的安全存储与传输。第三方AI服务（如Google Gemini）的调用需符合其API条款（如速率限制），并避免生成违规内容。长期需支持教育行业标准（如LTI、SCORM），以便与外部平台集成。

1. 遵循标准

本系统严格遵循主流软件开发标准以确保代码质量和可维护性。后端采用RESTful API设计规范，接口命名、状态码（如200成功、404资源不存在）和数据格式（JSON）均符合行业惯例，并基于OpenAPI 3.0生成标准化文档。数据库设计遵循第三范式（3NF），减少冗余并确保数据一致性，同时通过索引优化（如PostgreSQL的B-tree索引）提升查询性能。向量数据存储兼容Chroma DB的嵌入格式标准（如OpenAI的文本嵌入模型），确保语义搜索的准确性。

在教育领域，系统参考LTI（Learning Tools Interoperability）标准设计接口，为未来与Moodle等平台集成预留兼容性；学习行为数据记录遵循xAPI（Experience API）规范，支持细粒度的学习进度追踪。安全性方面，系统全面落实OWASP Top 10防护措施，包括输入验证（防SQL注入）、HTTPS全站加密传输，以及基于角色的访问控制（RBAC），确保用户数据隐私符合GDPR和《个人信息保护法》要求。

四、硬件限制

系统部署基于华为云ECS服务器，最低配置要求为4核CPU、8GB内存及100GB SSD存储，确保AI计算、PostgreSQL数据库和Chroma向量数据库的稳定运行；同时支持横向扩展以应对高并发场景，并通过连接池优化（如PgBouncer）提升数据库性能，适配从低配移动端到云端服务的全场景硬件环境。

五、技术限制

本系统在技术实现上面临以下主要限制：AI模型依赖第三方API（如DeepSeek、Gemini）存在响应延迟和调用频率约束；数据库层面，PostgreSQL全文搜索对中文分词支持有限，需集成Jieba等分词工具，而Chroma向量数据库在单节点部署时可能遇到性能瓶颈；此外，网络带宽限制要求大文件上传需采用分块传输策略。这些限制需要在架构设计和资源分配时予以针对性优化。

六、其他

在系统架构设计中，我们特别关注了成本优化与用户体验的平衡。通过智能调度策略（如优先调用本地知识库、按需使用大模型API）有效控制运营成本，同时对高频访问数据实施本地缓存机制以提升响应速度。在容灾方面，系统采用华为云RDS每日自动备份，并为关键服务（如AI问答模块）设计降级方案，确保在第三方服务不可用时仍能提供基础功能，保障系统的持续可用性。

为适应未来发展需求，系统架构预留了标准化接口，支持与Moodle等主流教育平台的快速集成。知识图谱数据采用RDF等通用格式存储，确保可迁移性。

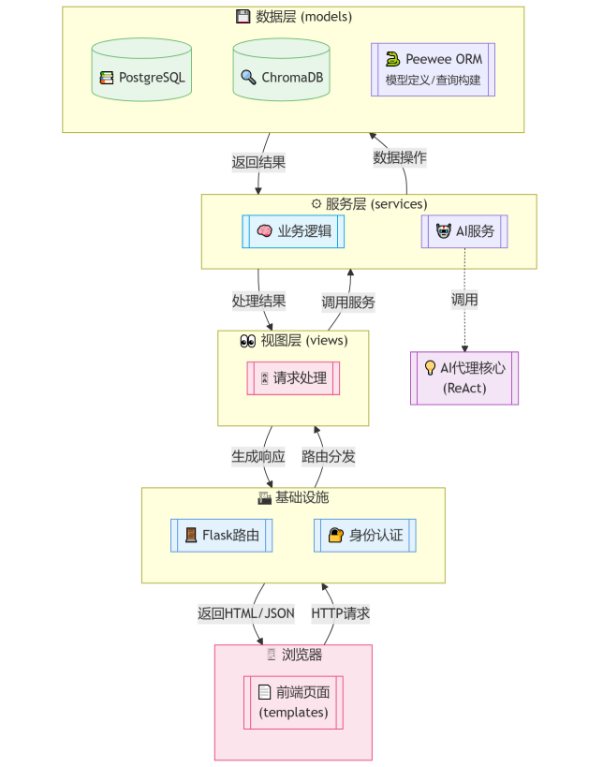
# 第一层设计描述

## 3.1 系统结构

一、系统结构描述

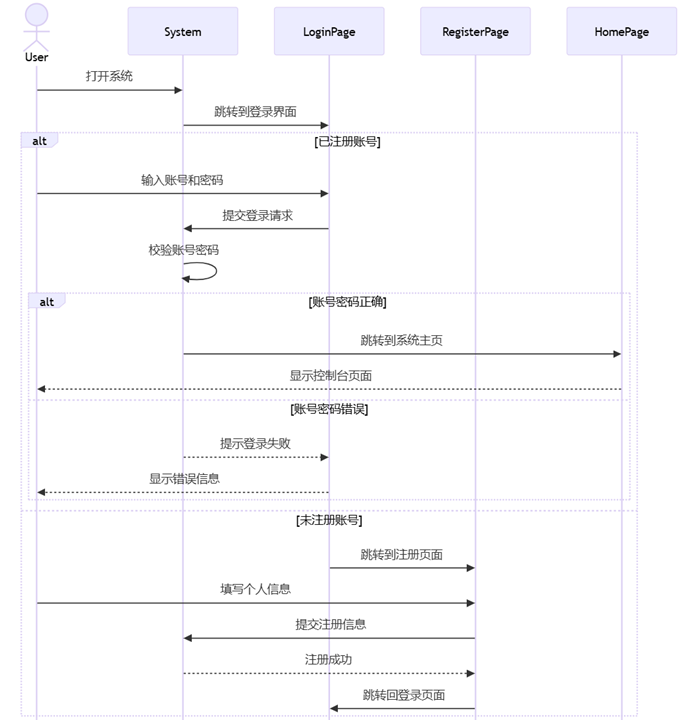
本系统采用分层架构设计，依托Python 3.10与轻量级框架Flask构建，以一体式架构简化开发流程，降低维护复杂度。

系统主要分为三层：数据层、服务层与视图层。数据层基于PostgreSQL存储用户、课程、作业与错题信息等核心业务数据，通过Peewee ORM实现模型定义与高效数据库交互，并集成ChromaDB增强检索能力，确保数据持久化与一致性。服务层封装业务逻辑与AI Agent模块。视图层由Flask框架支撑，负责HTTP请求路由分发与身份认证管理，结合Jinja2模板引擎动态生成HTML页面，实现前后端数据无缝衔接；AI Agent在此层响应前端交互，协调智能决策与结果反馈。

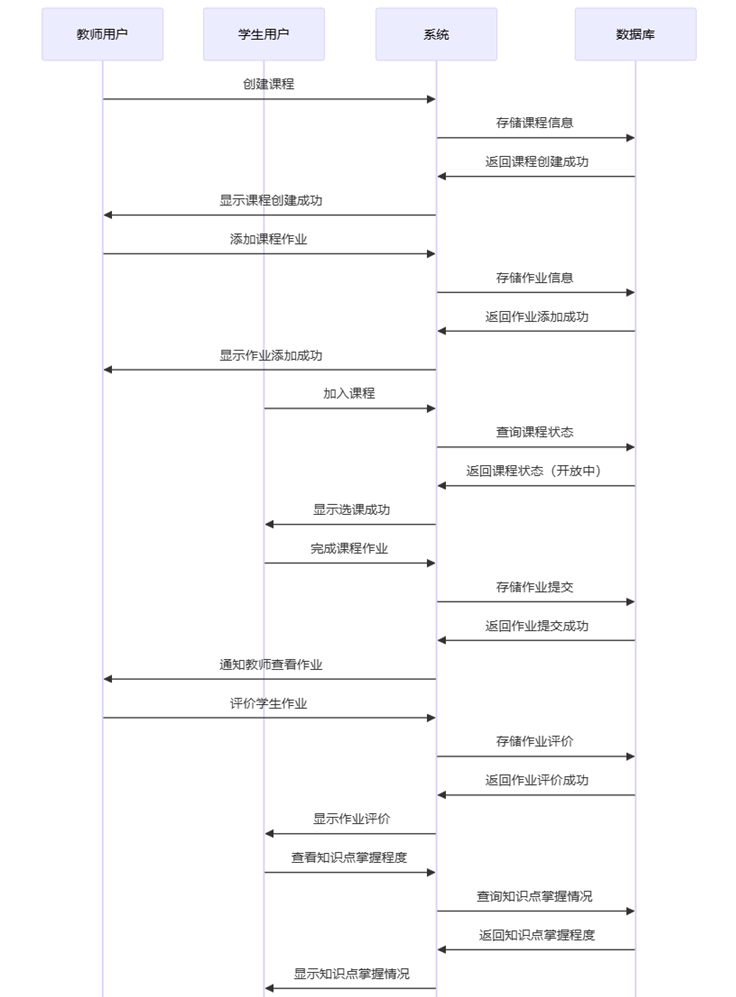


1. 业务流程说明

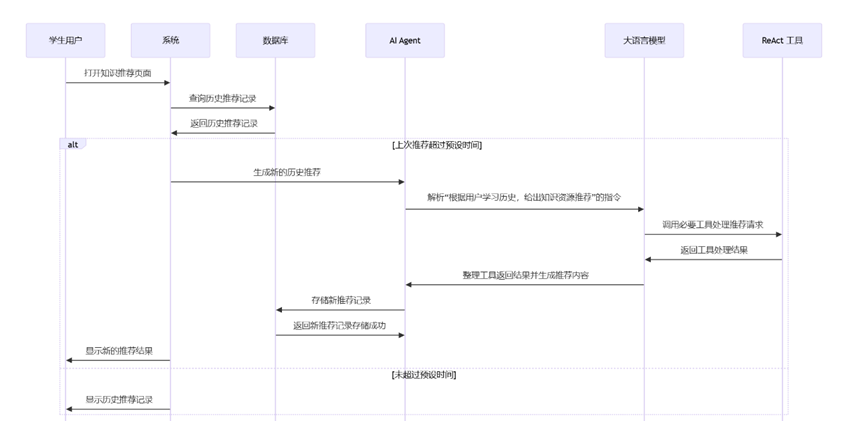
登陆注册界面流程如下：



课程学习模块流程如下：



知识推荐模块流程如下：



## 3.2 分解描述

一、登陆注册模块描述

1. Overview简介

用户登录注册功能模块的需求主要分为用户注册和用户登录两部分：用户注册功能允许新用户在系统中创建账户，通过填写必要信息（如用户名、密码、联系方式等）并进行合法性验证后，系统将其信息录入数据库，完成注册流程。用户登录功能则面向已注册用户，系统需验证其凭证信息（如用户名和密码）是否匹配，验证成功后允许其访问系统内部功能模块。

1. Functions功能列表

**表3.1 登录注册模块的具体需求**

|  |  |  |
| --- | --- | --- |
| **序号** | **功能名称** | **功能描述** |
| 1 | 用户注册 | 新用户填写基本信息完成注册，包括用户名、密码、联系方式等，系统需进行格式校验及唯一性验证 |
| 2 | 用户登录 | 用户通过已注册的账号信息（用户名+密码）登录系统，系统需进行身份验证 |
| 3 | 信息校验 | 在用户提交注册或登录信息时，系统需对字段格式（如邮箱格式）与账号有效性进行校验 |
| 4 | 密码加密存储 | 用户密码应在后台加密存储，防止明文泄露，提升数据安全性 |
| 5 | 忘记密码找回 | 用户可通过邮箱验证码等方式进行身份验证后找回密码 |
| 6 | 身份鉴权与会话管理 | 系统需生成唯一登录会话标识（token），用于后续功能模块的访问权限控制 |

1. 课程学习模块描述
2. Overview简介

课程学习模块是本系统的核心功能模块之一，旨在为学生提供结构化的学习路径和个性化的学习体验。用户可在系统中选择感兴趣或符合当前学习目标的课程，在AI学伴的陪伴与辅助下，系统引导用户依次完成课程内容、课后作业与互动练习。AI Agent 将持续记录用户在每个学习环节中的表现，根据用户的学习进度和任务完成情况，评估其掌握程度，并动态调整推荐的学习内容或提示建议，从而实现个性化学习支持。该模块不仅承担着教学内容的呈现与互动，还通过作业评价与学习数据分析，形成闭环式反馈机制，帮助用户明确自身的学习薄弱环节并及时进行查漏补缺。

1. Functions功能列表

**表3.2 课程学习模块的具体需求**

|  |  |  |
| --- | --- | --- |
| **序号** | **功能名称** | **功能描述** |
| 1 | 课程选择 | 用户可浏览平台上提供的课程列表，自主选择一门或多门课程进行学习 |
| 2 | 内容学习 | 用户可按章节逐步阅读学习资料（视频、文本、图文等），系统记录学习进度 |
| 3 | 作业提交 | 每个课程包含课后作业，用户可在线提交作业，支持文本输入或文件上传 |
| 4 | AI自动评估 | AI根据用户的作业完成情况、准确率、学习时间等维度进行评估并生成掌握程度报告 |
| 5 | 学习反馈与建议 | 根据评估结果，系统给出个性化学习建议或推荐额外练习内容 |
| 6 | 进度管理与提醒 | 系统记录课程学习进度，支持阶段提醒、任务截止提醒等功能 |
| 7 | 学习历史与统计查看 | 用户可查看个人历史学习记录，包括已完成课程、作业成绩、掌握情况图表等 |

1. 知识推荐模块描述
2. Overview简介

知识推荐模块作为系统实现“个性化学习支持”的重要支撑，主要依托大语言模型和用户画像数据，为不同阶段、不同学习目标的用户动态推送与其当前学习内容相关的知识点、扩展阅读、典型习题或外部资源。通过智能推荐算法，系统能够实现从“被动接受”到“主动引导”的转变，帮助用户更深入地理解所学内容、拓展知识广度，并提升学习的连贯性与效率。

该模块将充分考虑用户的学习行为数据（如课程学习进度、作业完成情况、AI掌握程度评估结果等），构建用户的知识掌握图谱，进而精准匹配知识资源。此外，系统还应支持用户主动查询与收藏推荐内容，提升学习的互动性和可控性。

1. Functions功能列表

**表3.3 知识推荐模块的具体需求**

|  |  |  |
| --- | --- | --- |
| **序号** | **功能名称** | **功能描述** |
| 1 | 学习行为数据分析 | 系统从数据库中读取用户的课程学习记录、作业完成情况和掌握程度等数据，进行行为分析 |
| 2 | 智能推荐触发机制 | 在用户学习后或主动请求时，触发推荐流程，自动调用知识搜索服务引擎 |
| 3 | 联网搜索推荐工具 | 系统通过接入外部知识搜索工具（如搜索引擎API、大模型等）获取相关知识内容 |
| 4 | 知识内容筛选与处理 | 对搜索结果进行筛选、摘要、去重与结构化处理，确保推荐内容的质量与可读性 |
| 5 | 推荐内容入库 | 生成的推荐内容将自动写入数据库，按用户ID、时间戳、推荐标签等信息记录 |
| 6 | 用户查看推荐历史 | 用户可在系统中随时查看历史推荐内容，支持按课程、标签或时间筛选 |

## 3.3 依赖性描述

**数据流依赖**

* 用户请求流程: 浏览器 → Flask路由 → 视图层 → 服务层 → 数据层 → 返回响应
* AI代理流程: 用户输入 → ReAct代理 → 工具选择 → 服务调用 → 结果整合 → 响应输出
* 数据同步流程: 关系数据库 ↔ 向量数据库 ↔ 图数据库

**模块间依赖**

* 服务层依赖: 所有服务类都依赖数据模型层，部分服务间存在交叉依赖
* 视图层依赖: 视图函数依赖对应的服务层和模板系统
* AI工具依赖: 工具函数依赖服务层提供的业务功能
* 数据库依赖: 三种数据库通过服务层进行数据同步和一致性维护

**外部系统依赖**

* 大语言模型: 支持多种LLM接口（DeepSeek、Gemini、LM Studio等）
* 第三方API: Google搜索API、Wiki API等
* 文件系统: 本地文件存储和OCR处理
* 环境配置: 通过环境变量管理敏感配置信息

## 3.4 接口描述

基于对EduAssistant系统的分析，我为您撰写详细的接口描述：

**一、用户管理模块接口描述**

### 1. 用户认证接口

**名称：** authenticate\_user

**说明：** 验证用户登录凭证，返回用户对象或None

**定义：**

def authenticate\_user(username: str, password: str) -> User or None

**参数：**

* username: 用户名
* password: 密码

**返回值：** 验证成功返回User对象，失败返回None

### 2. 用户创建接口

**名称：** create\_user

**说明：** 创建新用户账户并分配角色

**定义：**

def create\_user(username: str, email: str, password: str, name: str, role\_names: List[str] = None) -> User

**参数：**

* username: 用户名（唯一）
* email: 邮箱地址（唯一）
* password: 密码
* name: 真实姓名
* role\_names: 角色名称列表

**返回值：** 创建的用户对象

### 3. 角色检查接口

**名称：** has\_role

**说明：** 检查用户是否拥有指定角色

**定义：**

def has\_role(user: User, role\_name: str) -> bool

**参数：**

* user: 用户对象
* role\_name: 角色名称

**返回值：** 布尔值，表示是否拥有该角色

**二、课程管理模块接口描述**

### 1. 课程创建接口

**名称：** create\_course

**说明：** 创建新课程并关联教师

**定义：**

def create\_course(name: str, code: str, description: str, teacher\_id: int) -> Course

**参数：**

* name: 课程名称
* code: 课程代码
* description: 课程描述
* teacher\_id: 教师用户ID

**返回值：** 创建的课程对象

### 2. 学生选课接口

**名称：** enroll\_student

**说明：** 将学生添加到指定课程

**定义：**

def enroll\_student(student\_id: int, course\_id: int) -> StudentCourse

**参数：**

* student\_id: 学生用户ID
* course\_id: 课程ID

**返回值：** 选课关系对象

### 3. 获取教师课程接口

**名称：** get\_courses\_by\_teacher

**说明：** 获取指定教师的所有课程

**定义：**

def get\_courses\_by\_teacher(teacher\_id: int) -> List[Course]

**参数：**

* teacher\_id: 教师用户ID

**返回值：** 课程对象列表

**三、作业管理模块接口描述**

### 1. 作业创建接口

**名称：** create\_assignment

**说明：** 创建新作业并关联课程

**定义：**

def create\_assignment(title: str, description: str, course\_id: int, due\_date: datetime, total\_points: float = 100.0) -> Assignment

**参数：**

* title: 作业标题
* description: 作业描述
* course\_id: 课程ID
* due\_date: 截止日期
* total\_points: 总分

**返回值：** 创建的作业对象

### 2. 作业提交接口

**名称：** submit\_assignment

**说明：** 学生提交作业答案

**定义：**

def submit\_assignment(student\_id: int, assignment\_id: int, answers: List[Dict], work\_time: datetime = None) -> StudentAssignment

**参数：**

* student\_id: 学生ID
* assignment\_id: 作业ID
* answers: 答案列表
* work\_time: 提交时间

**返回值：** 学生作业对象

### 3. 作业批改接口

**名称：** grade\_assignment

**说明：** 教师批改学生作业

**定义：**

def grade\_assignment(student\_assignment\_id: int, scores: List[float], feedback: str = None) -> StudentAssignment

**参数：**

* student\_assignment\_id: 学生作业ID
* scores: 各题得分列表
* feedback: 评语

**返回值：** 更新后的学生作业对象

**四、知识库管理模块接口描述**

### 1. 知识条目添加接口

**名称：** add\_knowledge\_entry

**说明：** 添加新的知识条目到知识库

**定义：**

def add\_knowledge\_entry(title: str, content: str, category: str, tags: List[str] = None) -> KnowledgeEntry

**参数：**

* title: 知识条目标题
* content: 知识内容
* category: 分类
* tags: 标签列表

**返回值：** 创建的知识条目对象

### 2. 语义搜索接口

**名称：** semantic\_search

**说明：** 基于语义相似度搜索知识库

**定义：**

def semantic\_search(query: str, top\_k: int = 5) -> List[Dict]

**参数：**

* query: 搜索查询
* top\_k: 返回结果数量

**返回值：** 搜索结果列表，包含相似度和内容

### 3. 知识点导入接口

**名称：** import\_excel\_to\_knowledge\_points

**说明：** 从Excel文件批量导入知识点到图数据库

**定义：**

def import\_excel\_to\_knowledge\_points(file\_path: str, course\_id: int) -> Dict

**参数：**

* file\_path: Excel文件路径
* course\_id: 课程ID

**返回值：** 导入结果字典

**五、AI代理模块接口描述**

### 1. AI对话接口

**名称：** chat\_with\_agent

**说明：** 与AI代理进行对话交互

**定义：**

def chat\_with\_agent(user\_id: int, message: str, role: str = 'student') -> str

**参数：**

* user\_id: 用户ID
* message: 用户消息
* role: 用户角色

**返回值：** AI代理的回复消息

### 2. 工具注册接口

**名称：** register\_as\_tool

**说明：** 将函数注册为AI代理可使用的工具

**定义：**

def register\_as\_tool(roles: List[str] = None, description: str = None)

**参数：**

* roles: 允许使用该工具的角色列表
* description: 工具描述

**返回值：** 装饰器函数

### 3. 题目生成接口

**名称：** generate\_questions

**说明：** 基于知识点自动生成题目

**定义：**

def generate\_questions(knowledge\_points: List[str], question\_type: str, count: int = 5) -> List[Dict]

**参数：**

* knowledge\_points: 知识点列表
* question\_type: 题目类型
* count: 生成数量

**返回值：** 生成的题目列表

**六、学习分析模块接口描述**

### 1. 掌握度预测接口

**名称：** predict\_student\_mastery

**说明：** 使用NeuralCDM模型预测学生知识点掌握度

**定义：**

def predict\_student\_mastery(student\_id: int, course\_id: int) -> Dict

**参数：**

* student\_id: 学生ID
* course\_id: 课程ID

**返回值：** 掌握度预测结果字典

### 2. 学习行为分析接口

**名称：** analyze\_learning\_behavior

**说明：** 分析学生学习行为和模式

**定义：**

def analyze\_learning\_behavior(student\_id: int, course\_id: int, time\_range: str = 'month') -> Dict

**参数：**

* student\_id: 学生ID
* course\_id: 课程ID
* time\_range: 时间范围

**返回值：** 学习行为分析结果

### 3. 推荐生成接口

**名称：** get\_recommendations\_by\_history

**说明：** 基于学习历史生成个性化推荐

**定义：**

def get\_recommendations\_by\_history(user\_id: int) -> List[str]

**参数：**

* user\_id: 用户ID

**返回值：** 推荐内容列表

**七、错题本模块接口描述**

### 1. 错题添加接口

**名称：** add\_to\_wrongbook

**说明：** 将错题添加到学生的错题本

**定义：**

def add\_to\_wrongbook(student\_id: int, question\_id: int, course\_id: int, student\_answer\_id: int) -> QuestionWrongBook

**参数：**

* student\_id: 学生ID
* question\_id: 题目ID
* course\_id: 课程ID
* student\_answer\_id: 学生答题记录ID

**返回值：** 错题本条目对象

### 2. 错题练习接口

**名称：** practice\_wrong\_questions

**说明：** 从错题本生成练习题目

**定义：**

def practice\_wrong\_questions(student\_id: int, course\_id: int, count: int = 10) -> List[Dict]

**参数：**

* student\_id: 学生ID
* course\_id: 课程ID
* count: 练习题目数量

**返回值：** 练习题目列表

**八、外部系统接口描述**

### 1. Google搜索接口

**名称：** google\_search

**说明：** 调用Google搜索API获取网络信息

**定义：**

def google\_search(query: str, max\_results: int = 5) -> List[Dict]

**参数：**

* query: 搜索查询
* max\_results: 最大结果数

**返回值：** 搜索结果列表

### 2. OCR识别接口

**名称：** ocr\_recognize

**说明：** 识别图片中的文字内容

**定义：**

def ocr\_recognize(image\_path: str) -> str

**参数：**

* image\_path: 图片文件路径

**返回值：** 识别的文字内容

### 3. 大语言模型接口

**名称：** call\_llm

**说明：** 调用大语言模型进行文本生成

**定义：**

def call\_llm(prompt: str, model: str = 'deepseek', temperature: float = 0.7) -> str

**参数：**

* prompt: 输入提示
* model: 模型名称
* temperature: 生成温度

**返回值：** 生成的文本内容

# 第二层设计描述

采用分层架构设计，包含以下子模块：

1. 基础模型子模块：提供所有模型的公共基类

2. 教学管理子模块：处理课程、作业等教学核心数据

3. 学习分析子模块：记录和统计学习行为数据

4. 智能交互子模块：管理聊天会话和AI生成内容

5. 知识库子模块：构建知识体系和资源管理

## 4.1 基础模型模块

一、模块设计描述

作为整个系统的数据建模基石，该子模块通过BaseModel基类为所有数据模型提供统一的基础架构。采用Active Record模式封装了时间戳管理（created\_at/updated\_at）、数据库连接配置等核心功能，重写save()方法实现审计字段的自动维护。所有子模型通过继承该基类，自动获得标准的CRUD操作能力和数据持久化支持，确保系统数据层的一致性和可维护性。Meta类集中管理数据库连接策略，避免各模型分散配置。

**1、BaseModel类**

（1）**标识**

EduAssistant\_core\_models\_BaseModel

（2）**简介**

BaseModel 作为系统所有数据模型的基类，通过继承 Peewee 的 Model 类并扩展功能，为所有子模型提供统一的基础能力：自动维护 created\_at 和 updated\_at 时间戳字段，在每次调用 save() 方法时自动更新最后修改时间；通过内部 Meta 类集中管理数据库连接配置，确保所有模型使用统一的数据库实例；同时扩展基础 CRUD 操作，封装了包含时间戳自动更新等通用逻辑的 save() 方法，简化子模型的持久化操作，实现审计字段的自动化管理。

**2、Course类**

（1）**标识**

EduAssistant\_models\_Course

（2）**简介**

课程核心数据模型，存储课程基本信息包括名称、唯一课程代码、描述等，通过teacher外键关联User模型建立教师与课程的对应关系（一位教师可教授多门课程）。包含is\_active字段用于标记课程状态，提供\_\_repr\_\_方法便于调试时直观显示课程关键信息，并强制code字段唯一性确保课程标识不重复

**3、StudentCourse类**

（1）**标识**

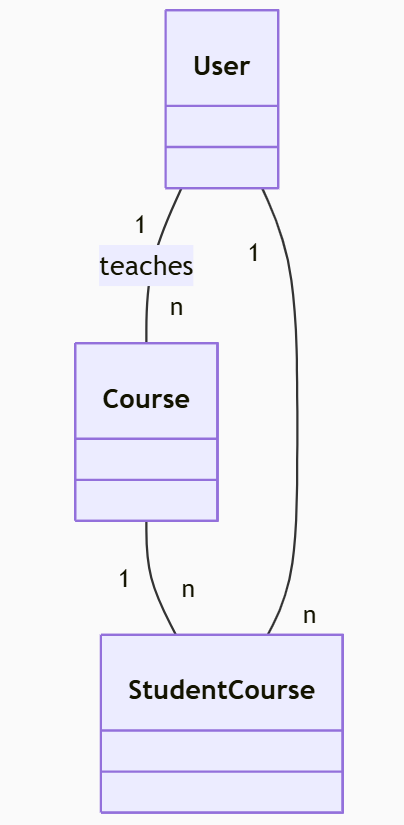
EduAssistant\_models\_StudentCourse

（2）**简介**

学生选课关联模型，建立学生与课程的多对多关系映射，通过复合索引确保每个学生只能选同一门课程一次。包含is\_active字段管理选课状态，通过外键分别关联User和Course模型，支持双向查询（如查询学生所选课程或课程下的学生列表），是教学管理系统中的核心关系型数据模型。

二、功能实现说明

课程管理协作图：



## 4.2 作业与评估模块

一、模块设计描述

该模块构建了教学业务的核心数据体系，包含Course（课程）、Assignment（作业）、StudentCourse（选课）等关键模型。通过外键关联实现教师-课程-学生的完整教学关系网络，其中Assignment模型支持作业全生命周期管理，StudentCourse模型使用复合索引确保选课唯一性。特别设计了冗余字段（如StudentAssignment中的course字段）优化高频查询场景，\_\_repr\_\_方法增强调试可视化能力，形成支撑教学业务流程的完整数据模型。

**1、Assignment类**

（1）**标识**

EduAssistant\_models\_Assignment

（2）**简介**

作业核心模型，记录作业基础信息（标题、描述、截止日期等）并与课程建立多对一关联，通过 course 外键关联 Course 模型，支持通过 backref 反向查询课程下的所有作业。提供 \_\_repr\_\_ 方法用于调试时直观显示作业与课程的关联关系，包含总分数字段（total\_points）用于成绩计算。

**2、StudentAssignment类**

（1）**标识**

EduAssistant\_models\_StudentAssignment

（2）**简介**

学生作业提交记录模型，使用复合主键（学生ID + 作业ID）确保唯一性，跟踪作业状态（未提交/已提交/已批改）、得分情况（总分与最终得分）及提交时间。通过外键同时关联 Assignment 和 User 模型，支持双向查询（如查询某学生的所有作业或某作业的所有学生提交记录），course 字段冗余存储课程信息以优化查询效率。

**3、Question类**

（1）**标识**

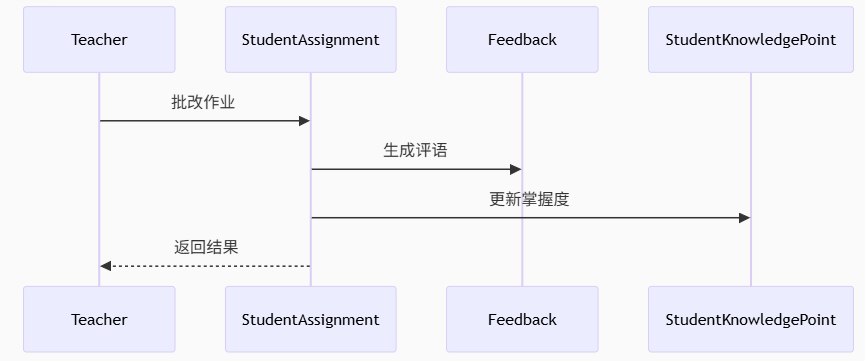
EduAssistant\_models\_Question

（2）**简介**

试题基础模型，存储各类题目的完整信息（题干、答案、解析等），通过status字段区分题型（选择/判断/简答），关联作业和课程，是智能题库系统的核心数据单元。

二、功能实现说明

作业批改序列图：



## 4.3 学习分析模块

一、模块设计描述

专注于学习过程数据的采集与分析，通过LearningActivity模型详细记录视频观看、作业完成等学习行为，结合StudentKnowledgePoint模型量化知识点掌握度。模型设计上采用JSONField存储行为元数据（如视频进度、交互热区），timestamp字段建立时间序列索引，支持对学习路径、知识漏洞的多维度分析。WrongBook及其关联模型构建错题管理体系，为个性化学习推荐提供数据基础。

**1、LearningActivity类**

（1）**标识**

EduAssistant\_models\_LearningActivity

（2）**简介**

学习行为记录模型，详细追踪学生在课程中的各类学习活动(视频观看、作业等)，记录持续时间、时间戳及元数据，为学习分析提供原始数据支撑。

**2、StudentKnowledgePoint类**

（1）**标识**

EduAssistant\_models\_StudentKnowledgePoint

（2）**简介**

学生知识点掌握度跟踪模型，记录学生对每个知识点的掌握程度(0-1范围)和最后学习时间，通过复合索引确保学生-知识点记录唯一性，支撑个性化学习分析。

**3、WrongBook类**

（1）**标识**

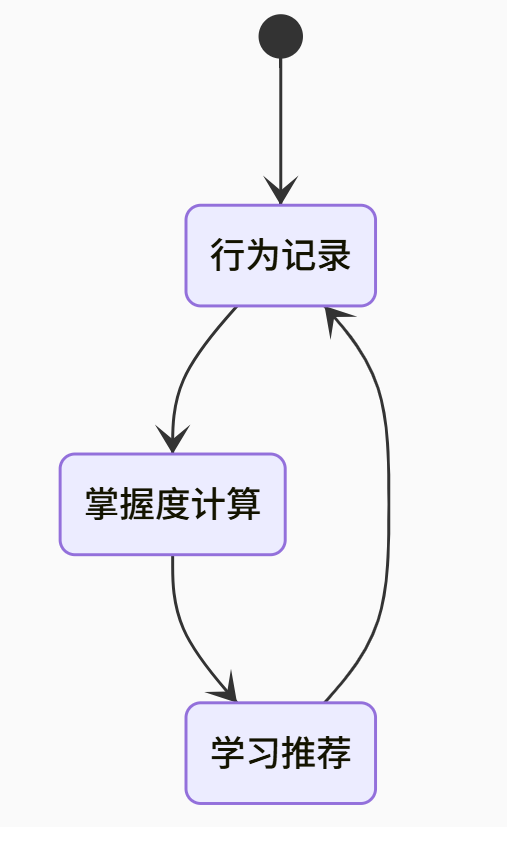
EduAssistant\_models\_WrongBook

（2）**简介**

错题本容器模型，管理学生按课程分类的错题集合，记录错题本名称和创建时间，是错题复习功能的基础数据结构。

二、功能实现说明

学习分析状态图：



## 4.4 智能交互模块

一、模块设计描述

管理系统的AI交互能力，Chat与ChatMessage模型构成完整的对话上下文体系，role字段明确区分用户/AI/系统消息来源，is\_active字段实现会话状态管理。AIQuestion模型扩展标准题目属性，通过is\_approved字段实现AI生成内容的审核工作流。技术实现上采用(chat, timestamp)联合索引优化消息排序查询，backref机制支持双向关系导航，满足智能教学场景的交互需求。

**1、Chat类**

（1）**标识**

EduAssistant\_models\_Chat

（2）**简介**

聊天会话主模型，记录用户与AI助手的完整对话会话，包含会话标题、激活状态等元信息，通过外键关联User模型建立用户与会话的一对多关系（一个用户可创建多个会话）。模型默认表名为"chats"，并提供is\_active字段用于标记会话是否有效，支持通过backref反向查询用户的所有聊天会话。

**2、ChatMessage类**

（1）**标识**

EduAssistant\_models\_ChatMessage

（2）**简介**

聊天消息明细模型，详细记录每个会话中的单条消息内容，通过role字段区分消息发送者身份（用户/AI助手/系统），包含消息内容、时间戳等核心字段。模型通过外键关联Chat主表建立一对多关系，并在(chat, timestamp)字段上创建联合索引以优化按会话和时间排序的查询性能，默认表名为"chat\_messages"。

**3、AIQuestion类**

（1）**标识**

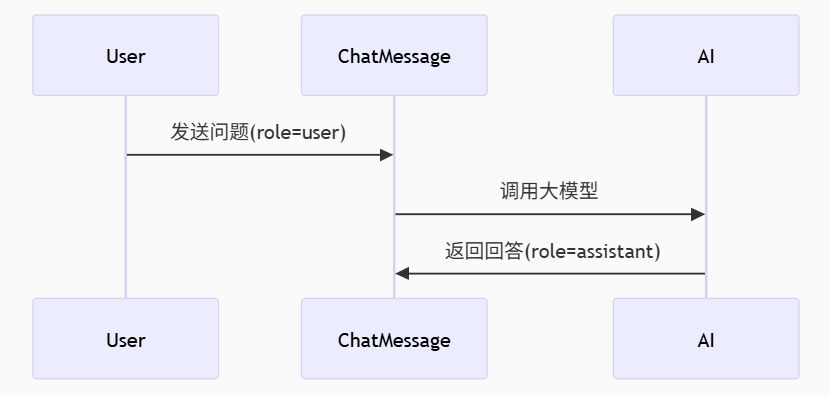
EduAssistant\_models\_AIQuestion

（2）**简介**

AI生成题目专用模型，继承基础题目属性并扩展审核状态字段（is\_approved），通过original\_question关联母题，实现AI题目生成、审核到入库的全流程管理。

二、功能实现说明

消息处理时序图：



## 4.5 知识库模块

一、模块设计描述

构建结构化知识体系的核心模块，KnowledgeBase模型支持多格式教学资源（文本/PDF/PPT）存储，通过vector\_id对接Chroma实现语义搜索。KnowledgePoint模型采用树形结构（parent自关联）组织知识点体系，AssignmentKnowledgePoint模型建立作业与知识点的权重映射。设计上强调知识关联性，通过多对多关系模型（如KnowledgeBaseKnowledgePoint）实现资源与知识体系的智能连接，为自适应学习提供知识图谱支撑。

**1、KnowledgeBase类**

（1）**标识**

EduAssistant\_models\_KnowledgeBase

（2）**简介**

知识库资源核心模型，用于存储和管理各类教学资源，包括纯文本、PDF、PPT等多种格式内容。模型通过type字段区分资源类型，content字段存储核心内容，并支持通过course外键与课程建立关联关系。特别设计vector\_id字段用于关联Chroma向量数据库实现语义搜索功能，同时提供tags字段以JSON格式存储资源标签，支持灵活的资源分类和检索，是智能教育系统中的重要知识存储载体。

**2、KnowledgePoint类**

（1）**标识**

EduAssistant\_models\_KnowledgePoint

（2）**简介**

学科知识点核心模型，通过树形结构(parent自关联)构建知识体系，关联特定课程并支持知识点层级关系管理，是教学知识图谱的基础构建单元。

**3、AssignmentKnowledgePoint类**

（1）**标识**

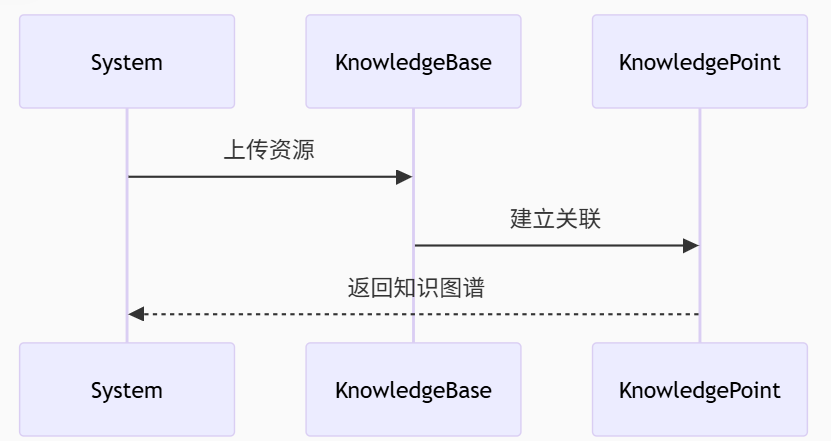
EduAssistant\_models\_AssignmentKnowledgePoint

（2）**简介**

作业-知识点关联模型，建立作业与知识点的多对多映射关系，通过weight字段表示知识点在作业中的权重，支持基于知识点的作业分析。

二、功能实现说明

知识关联序列图：



# 数据库设计

## 5.1 实体定义

一、分解描述

设计思路及约束规则：

* 采用关系型数据库（PostgreSQL）存储结构化数据，辅以向量数据库（Chroma）处理语义检索。
* 核心约束：
  + 用户权限隔离（学生/教师）
  + 知识图谱层级结构（父子关系依赖）
  + 学情数据动态更新（CDM算法驱动）
* 静态数据初始化：Role表预置基础角色（学生/教师/管理员）。

数据表定义:

1. User表:用于存储系统中所有用户的基本信息。



1. Role表:定义系统中的用户角色类型。



1. UserRole表:用户与角色的多对多关联表。



1. Course 表:存储教学课程的基本信息。



1. StudentCourse 表:学生与课程的多对多关联表，记录选课信息。



1. Assignment表:存储课程作业的基本信息。



1. StudentAssignment表:学生与作业的关联表，记录学生作业完成情况。



1. KnowledgePoint表:存储课程中的知识点信息。



1. StudentKnowledgePoint表(如表3.9):知识掌握度表记录学生对知识点的掌握程度。



1. LearningActivity表:学习活动表记录学生的学习行为和活动。



1. KnowledgeBase表:知识库表存储教学资源和知识条目。



1. Chat表:聊天表存储用户与AI助手的聊天会话。



1. ChatMessage表:聊天消息表存储聊天会话中的具体消息。



二、内部依赖性描述



## 5.2 行为定义

1. 分解描述

存储过程/触发器分类及功能:

|  |  |
| --- | --- |
| **类别** | **主要功能** |
| **学情分析类** | 动态计算知识点掌握度（CDM算法）、生成学习热力图 |
| **资源推荐类** | 基于学习路径算法生成个性化资源列表 |
| **知识图谱类** | 维护知识点的父子/依赖关系，确保图谱一致性 |
| **事务处理类** | 处理选课/退课、作业提交等原子操作 |

关键存储过程定义:

-- 存储过程: update\_knowledge\_mastery

-- 功能: 更新学生知识点掌握度（CDM算法）

-- 输入: student\_id, knowledge\_point\_id, new\_evidence (JSON)

-- 输出: 更新后的mastery\_level (FLOAT)

-- 依赖表: StudentKnowledgePoint, KnowledgePoint

-- 事务: REQUIRED (需原子性更新主知识点及关联点)

CREATE PROCEDURE update\_knowledge\_mastery(...)

-- 触发器: trg\_after\_assignment\_submit

-- 功能: 作业提交后自动触发知识点评估

-- 事件: AFTER INSERT ON StudentAssignment

-- 依赖: 调用update\_knowledge\_mastery存储过程

1. 外部依赖性描述

* **模块依赖**：
  + 学情分析模块依赖StudentKnowledgePoint数据更新。
  + AI问答模块依赖KnowledgeBase表的向量化存储。
* **API交互**：知识图谱构建算法需调用KnowledgePoint表父子关系数据。

1. 内部依赖性描述

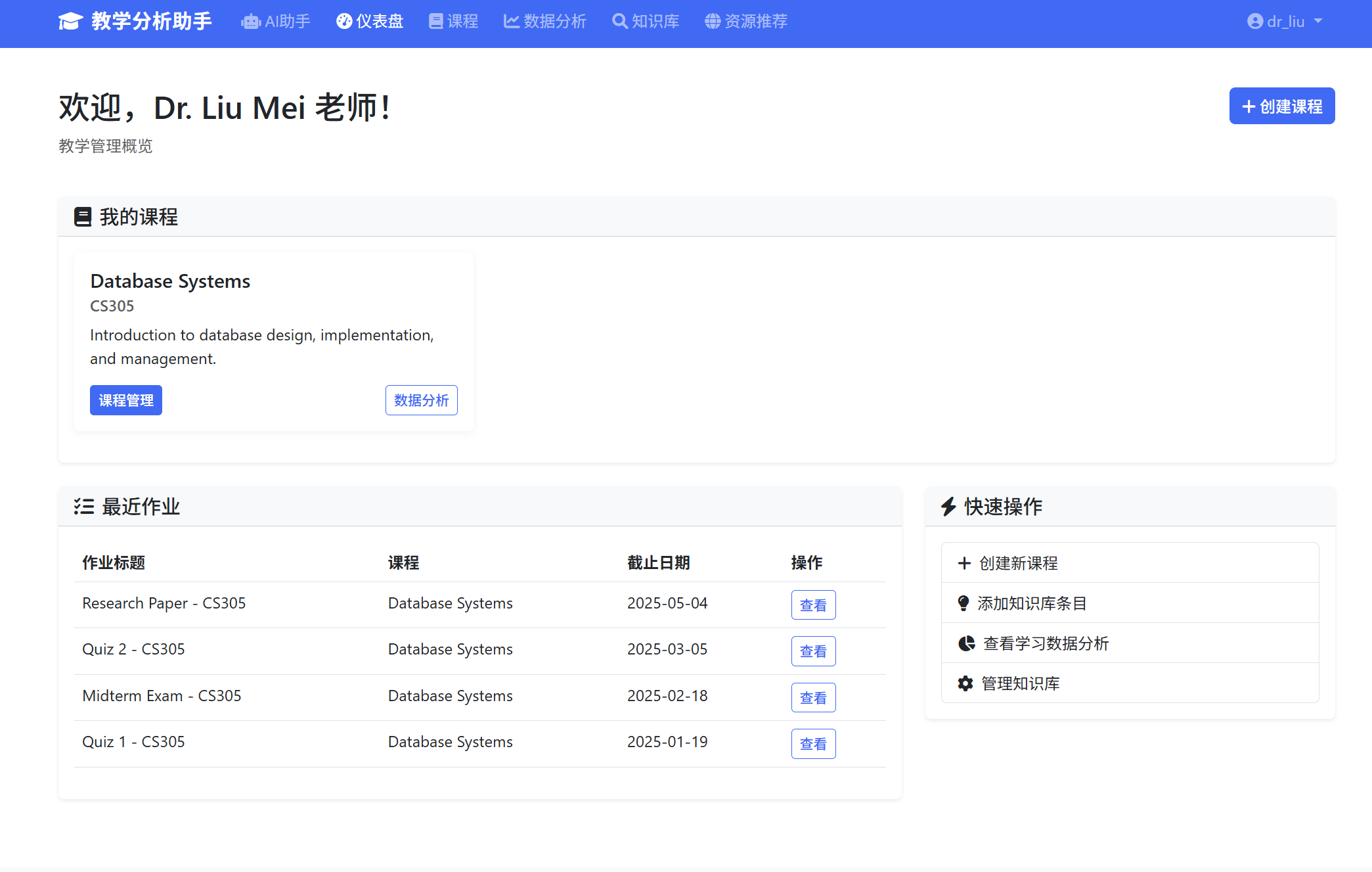
* **存储过程间依赖**：
  + generate\_learning\_path → get\_knowledge\_mastery
  + answer\_question → semantic\_search
* **数据表依赖**：
  + StudentKnowledgePoint表更新会触发LearningActivity日志记录。
  + 删除Course记录需级联清理Assignment和KnowledgePoint。

# 原型或界面设计

**登录页面**



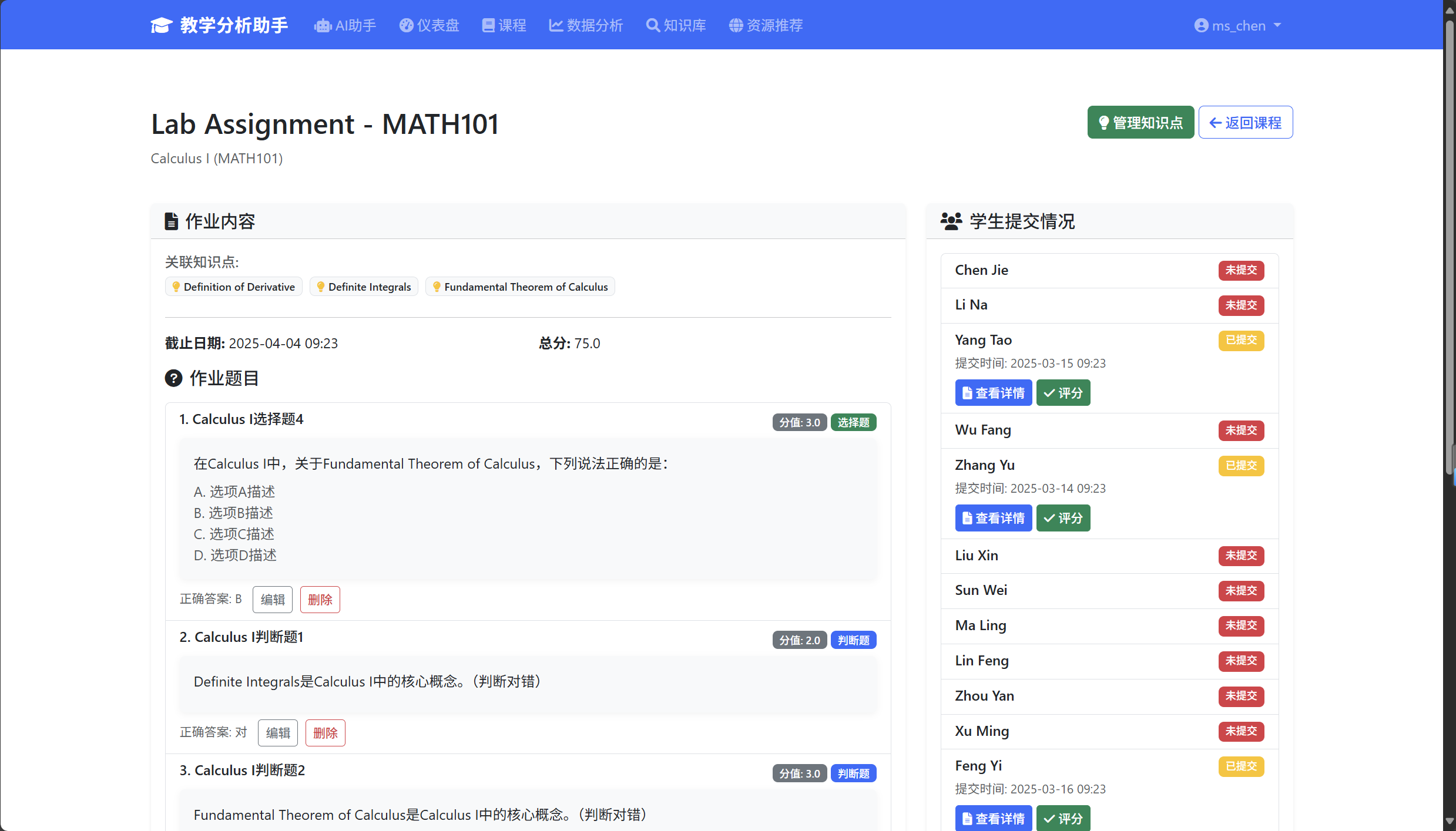
**主页面**



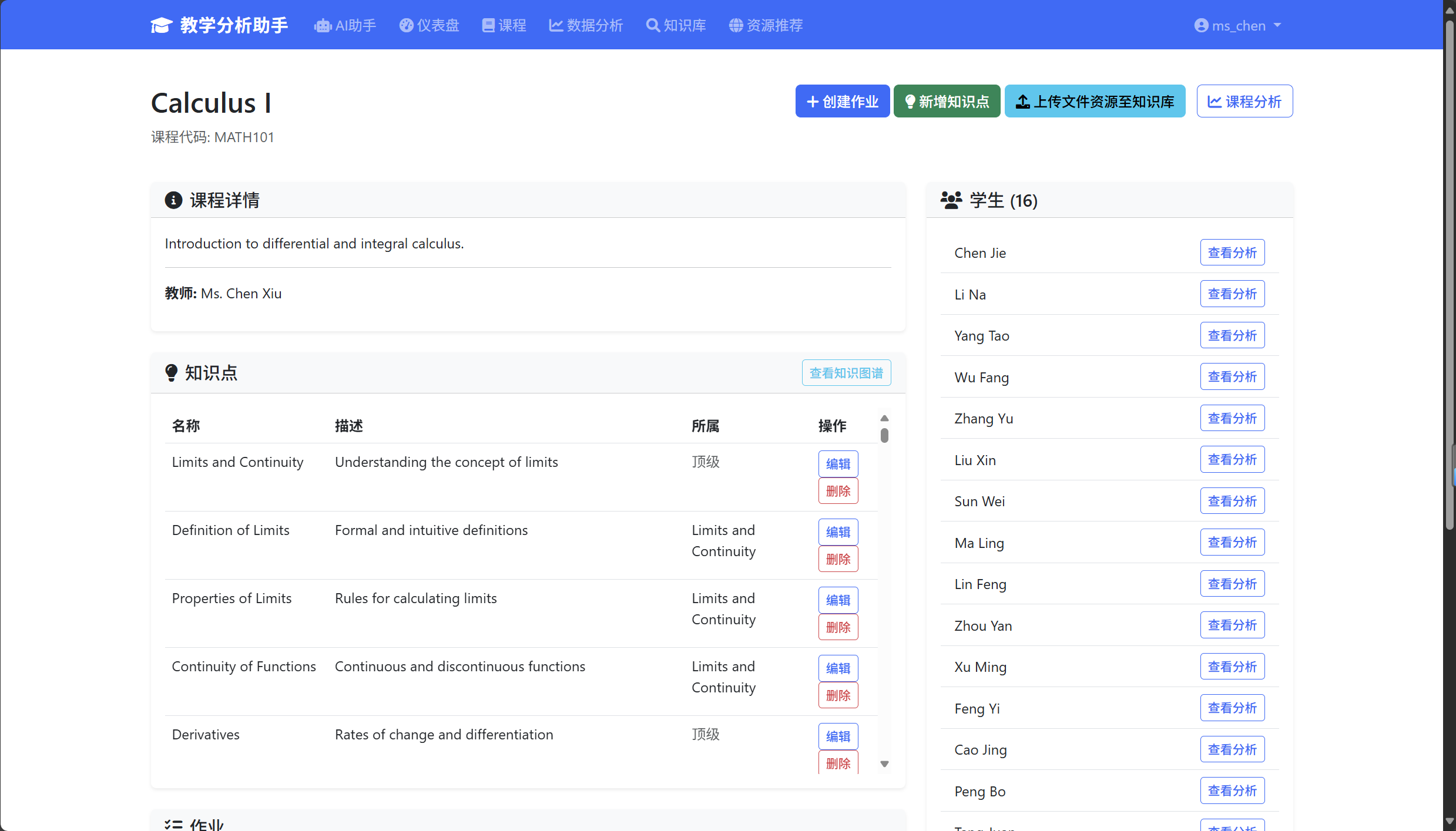
**AI交互界面**



**作业详情页**



**课程详情页**



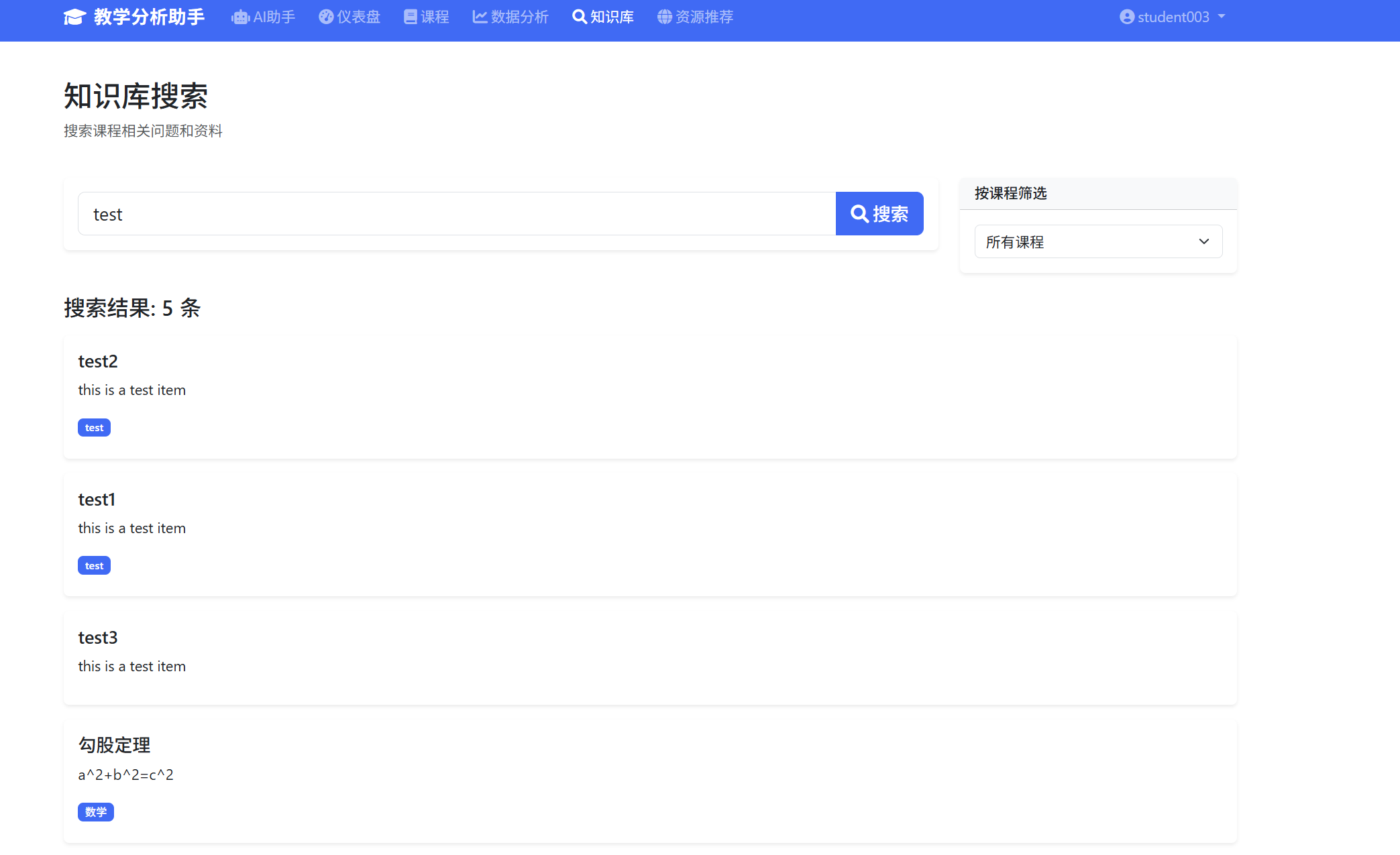
**创建作业与添加题目界面**



**数据分析页面**



**知识库页面**



**资源推荐页面**

