

AI个性化学习伴侣

一个基于人工智能的个性化学习系统，能够动态评估学习者知识水平，自适应生成学习内容，并提供多模态解释方式。

核心功能

1. **动态知识评估**：通过交互式问答快速绘制学习者知识图谱
2. **自适应内容生成**：根据学习进度自动调整讲解深度和案例难度
3. **多模态解释**：对同一概念提供文字、图解、代码示例等多种解释方式
4. **用户认证系统**：支持用户注册、登录和会话管理
5. **学习进度跟踪**：记录学习历史，跟踪学习进度
6. **定时任务系统**：定期分析用户进度和发送提醒
7. **交互式AI学习助手**：在各个学习页面提供AI对话功能，帮助解答学习问题
8. **丰富的课程主题**：提供多种计算机科学主题的课程内容
9. **学习路径规划**：根据用户目标和知识水平生成个性化学习路径
10. **进度可视化**：以图表形式展示学习进度和知识掌握情况

技术实现

- 后端框架：Python Flask
- API设计：RESTful API
- 数据库：MongoDB
- 用户认证：基于JWT的会话管理
- 异步任务：Celery + Redis
- 前端：原生HTML/CSS/JavaScript
- 大语言模型：阿里云百炼API
- 数据可视化：Matplotlib






项目当前进度

已完成功能

- ☒ 用户注册、登录和认证系统 (JWT)
- ☒ 知识评估模块 (通过问答形式评估用户水平)
- ☒ 自适应内容生成 (支持阿里云百炼API和本地回退方案)
- ☒ 学习进度跟踪和历史记录
- ☒ 多模态内容展示 (文本、图表等)
- ☒ 个性化学习路径规划
- ☒ 学习进度可视化 (图表展示)
- ☒ 练习题生成和反馈处理
- ☒ 定时任务系统 (Celery)
- ☒ 前端界面 (HTML/CSS/JavaScript)
- ☒ 交互式AI学习助手
- ☒ 完整的RESTful API
- ☒ 日志系统和错误处理

- ☒ 安全措施（密码加密、输入验证等）

待开发功能

-  移动端适配
-  社交学习功能（学习小组、排行榜等）
-  离线学习支持
-  更丰富的课程主题
-  学习效果评估机制

项目结构

```
.
├── app.py                # 主应用文件
├── config.py             # 配置文件
├── database.py           # 数据库模块
├── auth.py               # 认证模块
├── progress_tracker.py   # 学习进度跟踪模块
├── tasks.py              # 定时任务模块
├── logging_config.py     # 日志配置模块
├── requirements.txt      # 项目依赖
├── README.md             # 项目说明文档
├── .env.example          # 环境变量示例文件
├── docs/                 # 文档目录
│   └── api.md            # API文档
├── models/               # 数据模型目录
│   ├── user.py           # 用户模型
│   └── lesson.py         # 课程模型
├── utils/                # 工具类目录
│   ├── knowledge_analyzer.py # 知识分析工具
│   ├── content_generator.py # 内容生成工具
│   ├── validators.py      # 输入验证工具
│   ├── response.py        # 统一响应工具
│   ├── paginator.py       # 分页工具
│   ├── security.py        # 安全工具
│   ├── learning_path_planner.py # 学习路径规划工具
│   ├── feedback_processor.py # 反馈处理工具
│   └── progress_visualizer.py # 进度可视化工具
├── templates/            # 前端模板目录
│   ├── index.html        # 主页面
│   ├── dashboard.html    # 仪表板页面
│   ├── lesson.html       # 课程页面
│   ├── exercise.html     # 练习页面
│   ├── login.html        # 登录页面
│   └── register.html     # 注册页面
└── static/               # 静态文件目录
    └── style.css          # 样式文件
```

安装与运行

1. 克隆项目到本地：

```
git clone <项目地址>
cd ai-learning-companion
```

2. 创建虚拟环境（推荐）：

```
python -m venv venv
source venv/bin/activate # Linux/Mac
venv\Scripts\activate    # Windows
```

3. 安装依赖：

```
pip install -r requirements.txt
```

4. 安装和配置数据库：

- 安装MongoDB并启动服务
- 安装Redis并启动服务（用于Celery）

5. 配置环境变量：

- 复制`.env.example`为`.env`
- 根据实际情况修改配置参数

6. 运行应用：

```
python app.py
```

7. （可选）运行Celery worker处理异步任务：

```
celery -A tasks.celery worker --loglevel=info
```

8. （可选）运行Celery beat处理定时任务：

```
celery -A tasks.celery beat --loglevel=info
```

9. 访问应用： 打开浏览器访问 <http://localhost:5000>

本系统支持集成阿里云百炼API，以实现真实的自适应内容生成。

配置阿里云百炼API

1. 注册并登录[阿里云百炼平台](#)
2. 创建API密钥
3. 在[.env](file:///c:/Users/lenovo/Desktop/project/.env)文件中配置密钥：

```
DASHSCOPE_API_KEY=your-dashscope-api-key-here
```

使用说明

系统会自动检测配置的API密钥并使用阿里云百炼API。如果未配置，则回退到预定义的内容生成方式。

当使用阿里云百炼API时，系统会根据用户的学习目标、知识水平和进度动态生成个性化的解释内容和练习题，提供更真实和个性化的学习体验。

使用方法

1. 环境准备

安装依赖

首先，你需要安装项目所需的依赖包：

```
pip install -r requirements.txt
```

配置环境变量

复制 `.env.example` 文件为 `.env` 并根据实际情况修改配置：

```
cp .env.example .env
```

需要配置的关键参数包括：

- MongoDB连接地址
- Redis连接地址（用于Celery）
- Flask密钥
- JWT密钥
- 大语言模型API密钥（可选，但推荐配置）
- Celery配置

2. 启动服务

启动主应用

```
python app.py
```

启动Celery Worker (可选)

如果你需要处理异步任务，可以启动Celery worker：

```
celery -A tasks.celery worker --loglevel=info
```

启动Celery Beat (可选)

如果你需要执行定时任务，可以启动Celery beat：

```
celery -A tasks.celery beat --loglevel=info
```

3. 使用API接口

应用提供了一套完整的RESTful API接口，以下是具体的使用方法：

3.1 用户认证流程

注册新用户

```
curl -X POST http://localhost:5000/api/register \
-H "Content-Type: application/json" \
-d '{
  "username": "testuser",
  "email": "test@example.com",
  "password": "SecurePassword123"
}'
```

用户登录

```
curl -X POST http://localhost:5000/api/login \
-H "Content-Type: application/json" \
-d '{
  "username": "testuser",
  "password": "SecurePassword123"
}'
```

响应示例：

```
{
  "success": true,
  "message": "登录成功",
  "data": {
    "user_id": "用户ID",
    "username": "testuser",
    "token": "JWT访问令牌"
  }
}
```

3.2 核心功能使用

分析用户知识水平

```
curl -X POST http://localhost:5000/api/analyze-knowledge \
-H "Content-Type: application/json" \
-H "Authorization: Bearer 你的JWT令牌" \
-d '{
  "answers": [
    {
      "question_id": 1,
      "answer": "用户答案"
    }
  ]
}'
```

生成个性化课程

```
curl -X POST http://localhost:5000/api/generate-lesson \
-H "Content-Type: application/json" \
-H "Authorization: Bearer 你的JWT令牌" \
-d '{
  "learning_goal": "我想学习Python基础"
}'
```

生成个性化学习路径

```
curl -X POST http://localhost:5000/api/personalized-path \
-H "Content-Type: application/json" \
-H "Authorization: Bearer 你的JWT令牌" \
-d '{
  "learning_goal": "掌握数据结构"
}'
```

处理练习反馈

```
curl -X POST http://localhost:5000/api/exercise-feedback \
-H "Content-Type: application/json" \
-H "Authorization: Bearer 你的JWT令牌" \
-d '{
  "exercise_data": {
    "exercise_id": "练习ID",
    "type": "multiple_choice",
    "topic": "python_basics",
    "user_answer": "用户答案",
    "correct_answer": "正确答案"
  }
}'
```

获取学习进度摘要

```
curl -X GET http://localhost:5000/api/progress-summary \
-H "Authorization: Bearer 你的JWT令牌"
```

获取学习报告

```
curl -X GET http://localhost:5000/api/learning-report \
-H "Authorization: Bearer 你的JWT令牌"
```

获取特定主题的学习进度

```
curl -X GET http://localhost:5000/api/topic-progress/python_basics \
-H "Authorization: Bearer 你的JWT令牌"
```

3.3 其他实用接口

获取所有可用学习主题

```
curl -X GET http://localhost:5000/api/topics \
-H "Authorization: Bearer 你的JWT令牌"
```

获取学习主题推荐

```
curl -X GET http://localhost:5000/api/recommendations \
-H "Authorization: Bearer 你的JWT令牌"
```

获取学习进度

```
curl -X GET http://localhost:5000/api/progress \
-H "Authorization: Bearer 你的JWT令牌"
```

获取学习历史 (支持分页)

```
curl -X GET "http://localhost:5000/api/learning-history?page=1&per_page=10" \
-H "Authorization: Bearer 你的JWT令牌"
```

与AI交互式学习助手对话

```
curl -X POST http://localhost:5000/api/interactive-chat \
-H "Content-Type: application/json" \
-H "Authorization: Bearer 你的JWT令牌" \
-d '{
  "message": "我想学习Python变量",
  "context": {},
  "topic": "python_basics"
}'
```

4. 使用前端界面

应用也提供了Web界面，你可以通过浏览器访问以下页面：

- 主页: <http://localhost:5000/>
- 仪表板: <http://localhost:5000/dashboard>
- 登录页面: <http://localhost:5000/login>
- 注册页面: <http://localhost:5000/register>
- 课程页面: <http://localhost:5000/lesson>
- 练习页面: <http://localhost:5000/exercise>
- API文档: <http://localhost:5000/docs>

5. 完整使用流程示例

以下是一个完整的学习流程示例：

第一步：注册并登录

1. 访问 <http://localhost:5000/register> 注册账户
2. 访问 <http://localhost:5000/login> 登录账户

第二步：评估知识水平

1. 访问 <http://localhost:5000/dashboard>
2. 点击"开始知识评估"按钮
3. 回答系统给出的问题

第三步：生成学习路径

1. 在仪表板页面输入学习目标
2. 系统将根据你的知识水平生成个性化学习路径

第四步：学习课程内容

1. 访问 <http://localhost:5000/lesson> 查看课程
2. 阅读课程内容，系统会根据你的水平调整内容难度

第五步：完成练习

1. 访问 <http://localhost:5000/exercise> 完成相关练习
2. 系统会根据你的答案提供反馈

第六步：与AI助手交互

1. 在任何页面使用AI助手功能
2. 向AI助手提问学习相关问题

第七步：查看学习进度

1. 返回仪表板查看学习进度图表
2. 查看知识掌握情况和学习报告

课程主题

系统目前支持以下课程主题：

1. **Python基础** - 变量、数据类型、控制结构
2. **机器学习基础** - 监督学习、无监督学习、模型评估
3. **Web开发基础** - HTML基础、CSS样式、JavaScript交互
4. **数据结构与算法** - 数组、链表、树、图
5. **数据库基础** - SQL、关系模型、查询优化
6. **计算机网络基础** - TCP/IP、HTTP协议、网络安全
7. **操作系统基础** - 进程管理、内存管理、文件系统
8. **软件工程基础** - 软件开发生命周期、设计模式、测试方法
9. **网络安全基础** - 常见攻击类型、防护措施、加密技术

注意事项

1. 确保MongoDB和Redis服务正在运行
2. 请求体必须使用JSON格式，并设置正确的Content-Type头
3. 对于需要认证的接口，必须在请求头中包含有效的JWT令牌
4. 响应数据中的图表以base64编码的PNG图片形式提供
5. 分页接口支持page和per_page参数控制分页
6. 为了保证服务稳定性，API可能实施请求限流

展示亮点

- 现场演示系统如何对编程新手和进阶者解释同一概念的不同方式
- 展示知识图谱随时间进化的可视化效果
- 演示用户学习进度跟踪和个人化推荐
- 展示交互式AI学习助手在各个学习场景中的应用

后续优化方向

1. 开发完整的前端界面（React/Vue等）
2. 添加学习进度可视化功能
3. 实现社交学习功能（学习小组、排行榜等）
4. 添加移动端应用支持
5. 增强推荐算法，提供更精准的学习路径规划
6. 实现学习效果评估和反馈机制
7. 添加离线学习支持