**一、项目背景与意义**

1. **背景**  
   随着教育信息化的不断深入，利用信息技术手段辅助教学、提高教学质量与效率成为重要趋势。传统的题库管理与试卷生成方式往往需要教师手动搜集、筛选并编排题目，费时费力，且容易出现难度把控不准确、题目重复利用率不足等问题。另一方面，学生也需要大量的练习题来巩固所学知识，但市面上的练习资料数量庞大且质量参差不齐，难以满足个性化需求。  
   近年来，人工智能特别是自然语言处理技术（NLP）的快速发展，为智能题目生成提供了新的思路。通过训练或调用大型语言模型（如 GPT 系列、ChatGLM 等），可根据年级、难度、题型等条件快速生成相对高质量的题目，极大地减少教师在命题过程中的重复劳动，提高题目多样性和创新性，同时也为学生提供多元化的练习资源。
2. **意义**
   * 对教师：减少命题和批量生成练习题的负担，提高教学效率；实现对题库的灵活管理，支持按需定制试卷或练习题集。
   * 对学生：提供多样且个性化的练习题目，帮助学生针对性地复习与练习；结合题目难度与题型，满足不同阶段学生的需求。
   * 对学校或培训机构：可作为在线教学平台的辅助模块，增强教学信息化水平；有助于数据收集和教学分析，通过学生完成题目的情况进行教学质量评估与改进。

**二、项目目标与功能需求**

1. **基本目标**
   * **AI 题目生成**：根据用户（教师/学生）选择的学科、年级、难度、题型（如选择题、填空题、解答题等），快速生成相应题目。
   * **题库管理**：实现对已生成或编辑过的题目的存储、管理、检索与修改。
   * **用户管理**：至少支持两类角色（如教师、学生），教师可管理题库、生成试卷，学生可在前端平台自由选择难度或直接浏览题库进行练习。
2. **扩展目标**（为满足毕业设计对功能深度和广度的要求，可添加以下功能）：
   * **个性化学习推荐**：根据学生的做题数据，分析其薄弱知识点，智能推荐或再生成相应类型的练习题。
   * **题目解析与答案生成**：除了生成题目本身，还可以自动生成答案解析，为学生或教师提供参考。
   * **在线测评与记录**：学生可以在线提交答案，系统自动判断正确性（对客观题），并记录学生的错误率、正确率、平均用时等。教师可查看统计结果，并对学生进行针对性辅导。
   * **试卷一键生成**：教师选定一定数量或范围的知识点，系统自动生成综合性试卷，包含答案与难度评估。
   * **多语言支持**：对于某些语言类科目（如英语），可提供英译中、中译英或阅读理解类题目的生成。
   * **题目信息统计分析**：如同一类型题目的出现频率、难度分布、学生平均正确率等，为后续迭代和改进教学提供数据支持。

**三、可行性分析**

1. **技术可行性**
   * **AI 题目生成**：利用现有的大语言模型（如 OpenAI 的 GPT、国内的 ChatGLM 等）或其他 NLP 模型，结合提示工程（Prompt Engineering），即可生成较高质量的题目与答案。
   * **后端技术栈**：Python + FastAPI，可以快速搭建 RESTful API，结合 Pydantic 进行参数校验，确保请求数据的正确性与安全性。
   * **前端技术栈**：React + Tailwind CSS + shadcn/ui，能够以简洁、高扩展性的方式实现响应式界面以及组件化开发，进一步提高前端的开发效率和可维护性。
   * **数据库与存储**：可使用关系型数据库（MySQL、PostgreSQL）或轻量级的 SQLite 存储题目信息、用户信息、测评记录等数据，满足毕业设计规模需求。
2. **市场需求与应用价值**
   * 在教育领域，针对不同年级、学科、题型的练习题生成需求非常旺盛。若系统能够保证一定的准确度与智能化水平，将具备相当的实际应用价值。
   * 多数现有在线教育平台仅提供静态题库，尚缺乏灵活和高质量的 AI 题目生成功能。通过本系统，可以丰富平台的功能并提升用户体验。
3. **风险及挑战**
   * 生成题目的准确性与质量：需对 AI 输出的题目进行一定的过滤和审校，比如语法错误、知识点偏差等。
   * 题库规模与训练数据限制：若使用自研模型则需考虑训练语料与算力，若使用第三方接口则要兼顾接口稳定性和付费限制。
   * 数据隐私与安全：学生作答数据和个人信息的存储安全，以及接口调用的稳定性。

**四、系统整体设计**

1. **系统架构**
   * **前端**：React + Tailwind CSS + shadcn/ui
     + 界面层：用户登录注册、角色管理、题目生成条件设置、题库浏览与检索、做题与测评模块等。
   * **后端**：FastAPI + Pydantic
     + 业务逻辑层：负责接收前端请求，调用 AI 模型或第三方 API 生成题目；实现题库管理（增删改查）、用户管理、测评记录管理等功能。
     + 数据访问层：通过 ORM 或数据库客户端连接数据库，完成数据存储和查询。
     + AI 模型接入：可直接对接 API（如 OpenAI），也可引入本地部署模型（需要考虑性能和服务器资源）。
2. **模块划分**
   * **用户管理模块**
     + 用户注册登录：基于角色进行权限控制（教师 / 学生）。
     + 用户信息管理：如姓名、年级、课程偏好、做题记录等。
   * **AI 题目生成模块**
     + 参数配置：年级、学科、难度、题型等。
     + 模型调用：通过预设提示（Prompt），向语言模型发起请求并获得题目文本与答案。
     + 生成结果处理：对生成结果进行去重、筛选、格式化后存入题库。
   * **题库管理模块**
     + 题目增删改查：可按学科、知识点、难度等多维度检索和管理。
     + 题目编辑：对自动生成的题目进行人工修订和补充，保证质量。
   * **练习与测评模块**
     + 练习题列表：学生可选择自己想要练习的范围，系统自动或手动分配题目。
     + 作答与判分：对客观题自动判分，对主观题可由教师或系统辅助判分。
     + 学生测评记录：统计正确率、用时、薄弱知识点等信息，用于个性化推荐或成绩分析。
   * **数据统计与分析模块**
     + 题目使用率、正确率、难度曲线等。
     + 学生成绩增长、薄弱知识点统计等。

**五、关键技术与实现方案**

1. **AI 题目生成技术**
   * **Prompt Engineering**：通过编写合适的 Prompt 引导模型生成符合需求的题目，包括题干、选项、答案和解析。
   * **模型选择**：
     + 如果学校或团队许可，可使用国外的 GPT-3.5 / GPT-4 API。
     + 若需本地部署，可选择开源中文模型（如 ChatGLM），在本地服务器上部署并进行微调。
   * **后处理**：对生成的文本进行关键词提取或正则处理，确保题目的格式统一。
2. **后端实现**
   * **API 设计**：基于 FastAPI 定义接口，如 /generate\_question、/get\_questions、/submit\_answers 等。
   * **参数校验**：使用 Pydantic 定义数据模型，校验前端传入的参数合法性，例如生成题目的请求体结构、用户信息结构等。
   * **数据库设计**：
     + **用户表**（id、username、password、role、profile...）
     + **题目表**（id、subject、grade、difficulty、type、content、options、answer...）
     + **做题记录表**（id、user\_id、question\_id、is\_correct、time\_spent...）
   * **安全与权限控制**：可通过 JWT 或 Session 在 FastAPI 中实现身份认证与访问权限管理。
3. **前端实现**
   * **React 功能组件**：使用函数式组件 + Hooks，拆分为登录注册、题目生成、题库管理、练习测评、统计分析等部分。
   * **UI 框架**：Tailwind CSS + shadcn/ui，保证样式简洁统一，并可根据需求灵活定制。
   * **HTTP 请求**：利用 Axios 与后端对接，处理数据的增删改查、题目生成请求等。
   * **数据状态管理**：可选择 React 内置的 Context 方案或第三方库（Redux / Zustand 等）来统一管理应用全局状态。

**六、进度安排**

1. **需求分析与方案设计（第 1~2 周）**
   * 明确功能需求及系统架构，完成数据库、API 设计与原型图设计。
2. **后端开发（第 3~5 周）**
   * 搭建 FastAPI 项目结构，完成用户管理、题库管理、AI 接口调用等核心模块。
3. **前端开发（第 6~8 周）**
   * 结合设计稿实现页面布局及样式；对接后端 API，完成题目生成与题库管理等功能。
4. **测试与优化（第 9~10 周）**
   * 前后端联调；单元测试与集成测试；对功能和 UI 进行修改与完善。
5. **文档撰写与答辩准备（第 11~12 周）**
   * 编写使用手册、技术文档；准备演示 demo 与答辩 PPT。

**七、预期成果**

* **系统原型**：可在网页端进行题目生成、查看与测评的在线平台。
* **技术文档**：包含系统架构设计、数据库设计、接口说明、关键技术实现等。
* **可拓展方向**：后续可进一步结合机器学习算法实现更精准的个性化推题；或者与在线教育平台对接，形成更完整的教学解决方案。

**八、总结**

本项目以“AI 智能生成题目”为核心亮点，通过合理的技术架构设计与功能扩展，能够有效地展示在教育领域中应用人工智能的潜能，具有较好的实践价值和创新性。基于 Python + FastAPI 实现后端服务，React + Tailwind CSS + shadcn/ui 实现前端页面，不仅能够保证项目结构清晰易维护，也能够满足毕业设计对系统完整性与功能深度的要求。后续在开发过程中，需要重点关注生成题目的质量控制与安全性，确保系统在实际应用中具有稳定性与可扩展性。