武汉大学计算机学院
本科生课程报告

个人总结与课程反思报告

专 业 名 称 ：软件工程

课 程 名 称 ：软件工程

指 导 教 师 ：何露露 讲师

学 生 学 号 ：2023302111184

学 生 姓 名 ：陶恒轩

二〇二五年六月

个人项目总结与课程反思报告

1. 项目概述

个人角色与职责：

作为“闻道智能学习平台”的前端开发成员，我主要负责整个系统的前端架构设计和核心功能实现。

在技术层面，我主要负责四个核心模块的设计与开发工作。视频学习模块是整个平台的基础功能，我实现了视频播放器的自定义控件、进度记录、断点续播等功能，并通过组件化设计提高了代码复用性。AI 问答模块是系统的智能交互核心，我使用 WebSocket 实现了流式响应，让用户可以实时看到 AI 的回答过程,并设计了重连机制处理网络异常情况。知识图谱可视化模块通过 ECharts 将知识点之间的关系以图形化方式展现，帮助用户理解知识体系。用户中心模块则包含了用户认证、个人信息管理、学习进度追踪等功能，为整个平台提供了完整的用户体验。

作为前端开发的主要负责人，我还需要与后端团队密切配合，确保接口的顺利对接，参与技术方案的讨论，并负责前端代码的审查工作。这种角色让我不仅要专注于具体的编码实现，还要从整体架构的角度思考问题，这对我的技术能力和项目管理能力都是很大的提升。

个人技术栈与工具：

在开发技术方面，前端我选择了 Vue 3 + TypeScript + Vite 的现代化技术栈。Vue 3 的 Composition API 提供了更好的代码组织方式和类型推导支持，TypeScript 则增强了代码的可维护性和可靠性,Vite 作为新一代构建工具提供了极速的开发体验。

在核心功能模块实现上，UI 框架选择了 Vuetify 3，它提供了丰富的 Material Design 风格组件，大大提升了开发效率。状态管理采用了 Pinia，相比 Vuex 具有更好的 TypeScript 支持和更简洁的 API。视频播放器模块基于原生 HTML5 video 元素进行封装，实现了自定义控件、进度记录等功能。AI 问答模块使用 WebSocket 实现了流式响应，通过自定义的重连机制提升了连接稳定

性。知识图谱可视化则采用了 ECharts，它强大的图表功能满足了我们的可视化需求。HTTP 请求处理使用 Axios，通过拦截器统一处理了请求和响应。

在开发工具方面，我主要使用 VS Code 进行开发，配合 Volar 插件获得了很好的 Vue 开发体验。项目使用 头歌 Git 进行版本控制和协作。在软件工程方法上，我们采用了敏捷开发模式。前后端接口采用 RESTful 风格设计，并通过详细的接口文档确保了开发过程的顺畅。

2. 项目实施过程

需求分析与设计：

需求调研：

通过在线文档的方式，与另一组同学运用 KJ 法进行头脑风暴，明确各个优先级与需求调整与考量。此外，进行了竞品分析，参考主流教育平台的功能特点，逐步明确、优化核心需求。

设计思想：

项目采用了现代化的软件工程方法论和架构设计，主要采用前后端分离架构，前端使用 Vue 3 + TypeScript 技术栈，后端使用 Flask + Python。

在架构设计上，我们实现了清晰的分层结构，包括展示层、组件层、状态管理层、服务层和工具层，并通过 Vuetify 3 确保了界面的美观性和一致性。

在模块化设计方面，采用 Vue 3 的 Composition API 实现了更细粒度的代码组织和逻辑复用，并使用 TypeScript 强类型系统提高了代码的可维护性。

在功能实现上，我们集成了大语言模型实现智能问答功能，使用 RAG 技术提升问答准确性，并实现了知识图谱可视化支持知识点关联分析。

在安全性方面，实现了基于 JWT 的用户认证机制和 RBAC 权限管理。同时，我们注重性能优化，实现了组件的按需加载、虚拟滚动和资源预加载等策略，并通过响应式布局和渐进式加载提供了良好的用户体验。

编码与实现：

在编码实现阶段，我主要负责前端核心模块的开发工作。在视频学习模块中，我采用 Vue 3 的 Composition API 开发了自定义的视频播放器组件，结合后端实现了视频进度记录、断点续播、章节跳转、倍速播放等功能。在 AI 问答模块中，我实现了基于 WebSocket 的流式响应机制，通过分片传输和动态渲染解决

了长文本加载的用户体验问题，同时集成了 Markdown 渲染引擎，支持富文本格式的答案展示。在知识图谱可视化模块中，我使用 ECharts 实现了交互式的知识点关系图，通过力导向图算法优化节点布局，实现了知识点的关联展示和学习路径推荐。在性能优化方面，我实现了组件的按需加载、虚拟滚动和资源预加载策略，显著提升了系统响应速度。

在开发过程中，我遵循 Vue 3 官方的编码规范，采用统一的命名约定（如组件使用 PascalCase，变量使用 camelCase），并为复杂逻辑编写详细的注释。我们团队采用 ESLint 和 Prettier 进行代码规范检查，通过 Git Hook 在提交前自动格式化代码。在架构设计上，我采用了 Pinia 进行状态管理，实现了数据流的可预测性，并通过模块化设计提高了代码的可维护性。通过定期的代码审查和团队技术分享，我们不断改进代码质量，确保了项目的可持续发展。

测试与迭代：

在测试与迭代阶段，我主要采用了手动测试和基本的自动化测试相结合的策略。在前端开发过程中，我首先会在本地环境进行功能测试，重点测试用户界面交互、数据展示的正确性以及与后端 API 的交互。特别是在视频播放模块中，我详细测试了视频加载、播放控制、进度记录等功能，在 AI 问答模块中，重点测试了问答的实时响应和历史记录功能。在发现问题时，我使用 Vue DevTools 和浏览器开发者工具进行调试，通过 console.log 和网络请求监控来定位问题。

在开发过程中遇到的主要技术难点包括：视频播放的断点续播功能实现、基于 WebSocket 的 AI 问答流式响应处理、以及大规模数据的前端性能优化等。通过与后端团队的密切配合和反复测试，我们最终解决了这些问题，并在此过程中建立了更规范的测试流程和问题处理机制。

项目管理：

在项目管理方面，我们团队采用了敏捷开发的方式，通过 GitHub 进行协作开发和版本控制。项目初期，我们制定了开发计划，将整个项目分为需求分析、架构设计、核心功能开发和系统优化四个阶段。在实际执行过程中，确实遇到了一些进度延误，主要是在 AI 问答功能的开发阶段，由于需要解决大语言模型接入和流式响应的技术难点，比预期多花了几天时间。为了追赶进度，我们及时调整了开发策略，优先实现核心功能，将一些非关键的功能（如数据分析面板）延

后处理。在团队协作方面，我们主要通过线上交流、线下共同开发等进行同步，使用 QQ 群进行日常交流，遇到技术问题时通过腾讯会议等进行远程协作。这种即时沟通的方式在大多数情况下都能高效解决问题。

3. **贡献与达成度**

个人贡献度：

作为项目的前端开发人员，我主要负责了整个前端架构的搭建和核心功能模块的开发。在代码贡献方面，我进行了 52 次代码提交，完成了包括用户认证、视频播放器、AI 问答、知识图谱可视化、笔记本页面等核心模块的开发，编写了 9920 行有效代码（包括新增 7012 行，删除 2908 行）。在文档方面，我参与编写了详细的前端架构文档、需求报告和开发任务书，此外，承担了项目演示视频的初步剪辑以及 PPT 的制作工作。在性能优化方面，通过实现组件懒加载和虚拟滚动，减少了页面首屏加载时间；通过对 AI 问答功能实现流式响应，显著提升了用户体验，响应延迟从原来的整体等待变为实时显示。

目标达成度：

对比项目初期目标，我们成功完成了所有基础功能，包括视频学习、AI 智能问答、课程管理等核心模块，其中 AI 问答功能的实现甚至超出了预期，增加了知识库定制和上下文关联等特性。但由于时间限制，我们也不得不简化了一些原计划的功能，如数据分析面板只实现了基础的数据展示，原计划的学习行为分析和预测功能暂时搁置；知识图谱的交互功能也做了精简，目前只支持基础的节点展示和关系查看，而这些功能将在后续的开发中逐步完善。

4. **问题与改进**

技术层面：

在项目开发过程中，我遇到了一些尚未完全解决的技术难点：首先是在大规模数据渲染时的性能优化问题，虽然通过虚拟滚动部分改善了性能，但在复杂的知识图谱展示时仍存在卡顿现象；其次是 WebSocket 在弱网环境下的连接稳定性问题，目前的重连机制还不够完善。针对这些问题，我计划深入学习 Vue 3 的性能优化技术，以及研究 WebSocket 的断线重连和消息队列解决方案。

协作层面：

在协作层面，我们遇到的主要问题是远程协作效率不够理想，特别是在前后

端联调时，由于缺乏完整的接口文档和测试用例，经常需要反复沟通确认接口细节。例如，在实现视频播放进度同步功能时，就因为接口参数格式理解不一致，导致反复修改了多次代码。对此，我建议在今后的项目中引入更规范的接口文档管理工具，并建立完整的接口测试机制。

管理层面：

在项目管理方面，我们主要使用 GitHub 进行代码版本控制和问题追踪，使用 QQ 群和腾讯会议进行日常沟通。GitHub 的优势在于代码管理和问题追踪比较方便，但在项目进度管理和任务分配方面功能较弱。考虑到这些限制，在下一个项目中，我建议引入专门的项目管理工具（如 Jira）来更好地管理开发进度和任务分配，同时使用 Swagger 或 YApi 这样的接口文档工具来规范接口定义和测试流程，这样可以更好地提升团队的协作效率。

5. 课程反思

课程收获：

通过这门软件工程课程的学习和项目实践，我在多个方面都获得了显著成长。在知识体系构建方面，我系统地掌握了软件工程的基本理论和方法论，特别是深入理解了软件开发的完整生命周期，从需求分析、系统设计到测试维护的各个环节。在技术能力方面，我不仅提升了前端开发技能，掌握了 Vue 3、TypeScript 等现代化开发技术，还学会了如何在实际项目中应用设计模式和架构原则。更重要的是在思维模式上的转变，从原来只关注代码实现转变为注重软件质量和可维护性，学会了从项目整体角度思考问题，这让我对软件工程有了更深刻的认识。在软技能方面，通过团队协作，我提升了沟通能力和项目管理能力，学会了如何更好地与团队成员协作以及处理项目中的各种挑战。

课程改进建议：

对于课程改进建议，我认为在现在较为完善的课程上，也可以在以下几个方面进行优化，实现锦上添花的效果：

首先，建议增加更多实际工业界案例的分析和讨论，帮助学生更好地理解理论知识在实践中的应用；其次，在项目实践环节，可以引入更多的敏捷开发实践，如每周的 Sprint Review 和 Retrospective，让学生更真实地体验软件开发流程；最后，在考核方式上，建议增加对开发过程的考核比重，不仅关注最终成果，也

要重视学生在开发过程中的表现，包括代码质量、文档完整性、团队协作等方面。这样可以更全面地评估学生的软件工程能力。