AI+软件开发实践指南

近年来大语言模型（LLMs）技术及工具如雨后春笋得到快速发展，如ChatGPT、DeepSeek、Copilot、Cursor 等等。它们对软件工程领域产生了极大冲击，不仅可为软件开发和运维的各项活动提供智能化支持，提升软件开发的效率和质量，而且改变软件开发方式，产生新的软件开发范型。使用大模型工具来辅助软件开发已经成为重要的趋势，结对大模型工具来开发软件已成为每个软件工程师必备的能力。

# LLMs 工具辅助软件开发实践

整体而言，大模型工具可以辅助学生完成以下几个方面的软件开发实践活动， 包括需求分析、软件设计、代码编写、软件测试、运行维护等。

## 需求分析

目标：确定软件项目，构思和明确软件需求，定义软件功能和非功能需求。

## 项目选题和需求构思

**辅助创意生成**：可以利用大模型，针对特定应用领域或技术栈，生成软件项目的需求创意。

例如，输入“假设你是一个非常善于头脑风暴的创意工程师，请你帮我生成基于 AI 的教育应用创意”，模型可以生成多个项目想法，如“智能作业批改系统” 等。或者，在已经提供的课题项目中进行分析筛选，选择更有使用价值的、实现难度合适的项目。

**市场调研辅助**：模型可以帮助分析某个领域的市场需求或技术趋势，提供相关背景信息，帮助团队确定项目的可行性和创新性。（纯生成式大模型工具不合适直接用于市场调研，建议结合有智能搜索功能的大模型工具，例如纳米 AI 搜索、腾讯元宝、豆包等）

## 细化和分析软件需求，生成软件需求文档

**用户故事生成**: 根据用户角色和场景，生成符合实际应用场景的用户故事。例如：输入“作为顾客，想要搜索商品，以便找到想要购买的商品，请你根

据顾客的需求生成用户故事”， 大模型工具可以生成详细的用户故事，包括前置条件、后置条件、正常流程、异常流程等。

**需求澄清与角色扮演**：让大模型扮演不同的客户和用户角色，通过模拟对话的方式，帮助学生澄清模糊的需求，完善需求细节。例如：

场景 1：同学提出“网站要美观”， 大模型可以扮演设计师，询问具体的设计风格、配色方案、字体选择等，帮助学生明确“美观”的具体含义。

场景 2：同学提出“商品搜索功能”，大模型可以扮演电商平台商家，询问搜索算法的精准度、排序规则、筛选条件等，帮助学生优化搜索功能的设计。

**需求文档生成**：利用大模型工具，自动生成初步的需求文档模板。

例如：输入“假设你是一个资深的软件需求分析师，我正在开发一个在线购物网站，请你帮我生成包含功能列表、用户角色以及用例图等内容的需求文档框架，包括功能需求、非功能需求（如性能、安全性等）。”

## 软件设计

目标：逐步明确其解决方案，将需求转化为软件设计方案，包括体系结构、用户界面、数据等方面。

## 软件体系结构设计

**辅助架构设计**：利用大模型，根据需求描述，生成初步的软件体系结构图， 并推荐合适的设计模式和框架。

示例 1：输入“假设你是一个经验丰富的软件架构师，我正在开发一个在线购物网站，请你帮我设计一个可扩展、高性能的系统架构，并推荐合适的技术栈。” 大模型可以生成包含前端、后端、数据库、缓存等组件的架构图，并推荐使用Spring Boot、React、Redis 等技术。

示例 2：输入“我正在开发一个基于微服务架构的社交网络应用，请你帮我设计服务划分方案，并推荐合适的通信协议。” 大模型可以生成用户服务、内容服务、消息服务等微服务划分方案，并推荐使用 RESTful API 或 gRPC 进行通信。

## 用户界面设计

**辅助界面原型设计**：利用大模型根据用户角色和功能需求，生成初步的界面原型图，并提供设计建议。

示例 1：输入“假设你是一个资深 UI 设计师，我正在开发一个在线教育平台，请你为老师和学生角色分别设计课程管理界面和学习界面，并提供设计建议。” 大模型可以生成包含课程列表、视频播放、在线测试等元素的界面原型，并建议使用简洁明了的布局和符合教育主题的配色方案。

示例 2：输入“我正在开发一个移动端电商应用，请你设计商品详情页的界面布局，并考虑用户体验和交互设计。” 大模型可以生成包含商品图片、价格、购买按钮等元素的界面原型，并建议使用大图展示、滑动查看、一键购买等交互方式。

## 数据设计

**辅助数据库设计**：利用大模型，根据需求描述，生成初步的数据库模型，并推荐合适的数据库类型和数据结构。

示例 1：输入“假设你是一个数据库专家，我正在开发一个博客系统，请你帮我设计数据库表结构，并考虑文章的存储、分类、标签等功能。” 大模型可以生成包含用户表、文章表、分类表、标签表等表结构的 ER 图，并推荐使用 MySQL 或 PostgreSQL 数据库。

示例 2：输入“我正在开发一个实时聊天应用，请你设计消息存储方案，并考虑消息的实时性、可靠性和可扩展性。” 大模型可以生成使用 NoSQL 数据库（如MongoDB）存储消息的方案，并建议使用消息队列（如 Kafka）实现消息的异步处理。

## 详细设计

**辅助详细设计文档生成**：利用大模型，根据体系结构设计和用户界面设计， 生成类图、时序图等详细设计文档，并提供代码实现的思路和建议。

示例 1：输入“假设你是一个资深软件工程师，我正在开发一个在线购物网站，请你根据之前设计的系统架构，生成用户登录模块的类图和时序图，并提供代码实现的思路。” 大模型可以生成包含用户类、认证服务类等类图，以及用户登录流程的时序图，并建议使用 Spring Security 框架实现用户认证功能。

示例 2：输入“我正在开发一个基于微服务架构的社交网络应用，请你生成用户关注功能的 API 接口文档，并提供代码实现的示例。” 大模型可以生成包含请求方法、URL、参数、返回值等信息的 API 接口文档，并提供使用 Spring Boot 和 RESTful API 实现用户关注功能的代码示例。

## 编写代码

目标：将软件设计方案转化为可执行的代码。

## 代码生成

**辅助代码生成**：利用大模型，根据设计文档和自然语言描述，生成代码片段或完整函数。

示例 1：输入“假设你是一个资深 Python 开发工程师，我正在开发一个在线购物网站，请你根据之前设计的用户登录模块，生成用户注册功能的 Python 代码。” 大模型可以生成包含用户注册表单验证、密码加密、数据库操作等功能的代码片段。

示例 2：输入“我正在开发一个基于 React 的移动端应用，请你生成一个商品列表组件，并实现下拉刷新和上拉加载更多功能。” 大模型可以生成包含商品列表展示、下拉刷新、上拉加载更多等功能的 React 组件代码。

## 代码适配

**辅助代码适配**：从开源网站、问答社区或其他代码资源中找到可复用的代码片段后，利用大模型将其修改为适配当前项目上下文的代码。

示例 1：输入“我从 GitHub 上找到了一段用于用户认证的 Python 代码，但它是基于 Flask 框架的，而我的项目使用的是 Django 框架。请你帮我将这段代码适配到 Django 框架中。” 大模型可以将 Flask 的认证逻辑转换为 Django 的认证逻辑，并生成适配后的代码。

示例 2：输入“我从 Stack Overflow 上找到了一段用于处理文件上传的JavaScript 代码，但它是基于原生 JavaScript 的，而我的项目使用的是 React。请你帮我将这段代码适配到 React 组件中。” 大模型可以将原生 JavaScript 代码转换为 React 组件代码，并确保其与项目的状态管理和事件处理机制兼容。

## 代码优化

**辅助代码优化**：利用大模型，分析代码性能瓶颈，提供优化建议，并重构代码以提高代码可读性和可维护性。

示例 1：输入“**假**设你是一个性能优化专家，请你分析以下 Python 代码的性能瓶颈，并提供优化建议。” 大模型可以分析代码性能瓶颈，例如循环嵌套过深、数据库查询次数过多等，并提供优化建议，例如使用缓存、优化算法等。

示例 2：输入“我正在开发一个大型软件项目，请你重构以下代码，提高代码的可读性和可维护性**。**” 大模型可以重构代码，例如提取函数、消除重复代码、添加注释等，以提高代码的可读性和可维护性。

## 代码注释生成

**辅助代码注释生成**：利用大模型自动生成代码注释，解释代码功能和逻辑。示例 1：输入“请你为以下 Python 函数生成注释，解释函数的功能和参数。”

大模型可以生成详细的代码注释，解释函数的功能、参数、返回值等信息。

示例 2：输入“我正在开发一个开源项目，请你为以下 Java 类生成注释，解释类的功能和成员变量。” 大模型可以生成详细的代码注释，解释类的功能、成员变量、方法等信息。

## 软件测试

目标：通过一系列的测试，尽可能地发现并修复程序代码中的缺陷，提高软件质量。

## 测试用例生成

**辅助测试用例生成**：利用大模型，根据需求文档、设计文档或代码逻辑，自动生成测试用例。

示例 1：输入“假设你是一个资深测试工程师，我正在开发一个在线购物网站，请你根据用户登录功能的需求文档，生成测试用例，包括正常登录、密码错误、用户名不存在等场景。” 大模型可以生成详细的测试用例，包括测试步骤、预期结果、实际结果等。

示例 2：输入“我正在开发一个基于微服务架构的社交网络应用，请你根据用户关注功能的 API 接口文档，生成测试用例，包括关注成功、关注失败、重复关注等场景。” 大模型可以生成详细的测试用例，包括请求参数、预期响应、实际响应等。

## 测试脚本生成

**辅助测试脚本生成**：利用大模型，根据测试用例，自动生成测试脚本。

示例 1：输入“假设你是一个资深测试工程师，请你根据以下用户登录功能的测试用例，生成 Python 的单元测试脚本。你可以使用 unittest 或 pytest 框架的单元测试脚本。”

示例 2：输入“我正在开发一个 Web 应用，请你根据以下商品搜索功能的测试用例，生成 Selenium 的自动化测试脚本。” 大模型可以生成使用 Selenium 的自动化测试脚本，模拟用户操作浏览器进行测试。

## 测试报告生成

**辅助测试报告生成**： 利用大模型，帮助分析测试结果，识别潜在问题并提供改进建议。例如，输入测试失败日志，模型可以分析失败原因并建议修复方法。自动生成测试报告。

示例：输入“我正在开发一个移动端应用，请你根据以下自动化测试结果， 生成测试报告，包括测试覆盖率、性能指标、用户体验评分等信息。并提供优化建议。” 大模型可以生成全面的测试报告，帮助团队评估软件质量，并针对性的优化。

## 运行维护

目标：将软件部署到目标环境，确保其稳定运行，并根据用户反馈和运行数据持续改进软件。

## 部署与配置

**脚本生成**： 利用大模型，根据目标环境，自动生成部署脚本，提供目标环境的配置建议，例如服务器配置、网络配置、安全配置等。

示例 1：输入“假设你是一个 DevOps 工程师，我正在开发一个基于 Docker 的 Web 应用，请你生成一个 Docker Compose 文件，用于部署应用的前端、后端和数据库。” 大模型可以生成包含服务定义、网络配置、环境变量等内容的Docker Compose 文件。

示例 2：输入“假设你是一个系统管理员，我正在部署一个高并发的 Web 应用，请你提供服务器配置建议，包括 CPU、内存、磁盘等。” 大模型可以根据应用的需求，提供服务器配置建议，例如使用多核 CPU、大内存、SSD 磁盘等。

## 故障排查与修复

**故障排查与修复**： 利用大模型，分析系统日志，识别故障原因，提供修复建议。

示例：输入“假设你是一个故障排查专家，我正在排查一个 Web 应用的 500 错误，请你帮我分析以下日志文件，识别故障原因。” 大模型可以分析日志文件， 识别故障原因，例如数据库连接失败、代码逻辑错误等，并提供修复建议。

## 用户反馈分析与功能更新

**辅助用户反馈分析**： 利用大模型，分析用户反馈，识别软件缺陷和改进点。示例：输入“我正在分析一个在线教育平台的用户反馈，请你识别用户最常

提到的问题和改进建议。” 大模型可以分析用户反馈，识别常见问题，例如课程加载慢、界面不友好等，并提供改进建议。

**辅助功能更新与迭代**： 利用大模型，根据用户需求和市场趋势，提供功能更新建议，并生成功能更新文档。

示例：输入“假设你是一个产品经理，我正在规划一个电商应用的下一版本， 请你根据用户反馈和市场趋势，提供功能更新建议。” 大模型可以提供功能更新建议，例如增加直播带货功能、优化推荐算法、生成功能更新文档，帮助团队明确开发目标和优先级。

# 如何应用 LLMs 工具辅助软件开发

如何利用好大模型工具，解决你的问题？在与大模型或其他专业人员交互时， 如何组织问题、明确需求以及有效沟通是至关重要的。假设你能找到世界上任何你所能想到的专业人员来帮你解决问题，你应该如何组织你的问题，如何向他提问？如何科学有效地进行 Prompt 的设计，解决在软件开发实践过程中遇到的各个问题？

## 明确问题

问题的核心：首先明确你要解决的核心问题是什么。避免模糊或过于宽泛的问题。反例：不要问：“如何做一个好项目？”。正例：而是问：“在开发一个在线考试系统时，如何设计一个安全且高效的用户认证模块？”

问题的类型：确定问题的类型（如技术问题、设计问题、理论问题等），以便选择合适的提问方式。技术问题：例如，“如何在 Spring Boot 中实现 JWT 认证？” 设计问题：例如，“如何设计一个支持高并发的用户认证模块？”理论问题： 例如，“什么是微服务架构的最佳实践？”

## 交代背景和领域

角色预设：假设大模型是一个某某领域非常有经验的专家。示例：“假设你是一位资深的后端开发专家，专注于高并发系统的设计与优化。”

项目背景：提供项目的背景信息，包括目标、用户群体、技术栈等。示例： “我们正在开发一个在线考试系统，目标用户是大学生和教师，技术栈是 React 前端和 Spring Boot 后端。”

领域知识：如果问题涉及特定领域（如机器学习、区块链等），简要说明相关背景知识。示例：“我们正在开发一个基于机器学习的推荐系统，使用的是Python 和 TensorFlow。”

## 问题的组织与分解

问题分解：将复杂问题分解为多个子问题，step-by-step 逐步解决。示例：对于“如何设计一个在线考试系统？”可以分解为：“如何设计用户认证模块？如何实现考试创建和管理功能？如何确保系统的安全性和稳定性？”

优先级排序：根据问题的紧急程度或重要性排序，优先解决关键问题。示例：

在开发初期，优先解决用户认证和考试管理功能，后期再优化性能和安全性。

## 向大模型提问（Prompt 设计）

结构化提问：使用清晰、结构化的语言提问，确保问题易于理解。虽然它是机器，但是可以将它想象成一个专业人员，考虑它是否能够清晰的理解。

## 清晰的表达+弱大模型的意图理解优于模糊的表达+强大模型的意图理解。

示例：“在开发一个在线考试系统时，如何设计一个支持高并发的用户认证模块？我们目前使用的是 Spring Security，但担心性能问题。”

提供上下文：在提问时提供足够的上下文信息，避免对方需要猜测你的需求。如果可能，提供具体的示例或代码片段，帮助对方更好地理解问题。

示例：“我们正在开发一个 XX 电商平台，使用微服务架构，目前遇到的问题是订单服务的响应时间较慢。以下是我们当前的代码，看起来是循环嵌套太多， 如何优化？”

[附上相关代码片段]

## 汇总回答后进一步提问

总结回答：在获得回答后，先复述总结对方提供的信息，确保你理解正确。示例：“你建议我们使用缓存来优化订单服务的性能，具体可以使用 Redis，对吗？”

追问细节：如果回答不够详细或存在疑问，可以进一步追问。

示例：“关于使用 Redis 缓存，你能提供一些具体的实现示例吗？”

建议验证：如果对方提供了解决方案，但其超出了我们的知识范围，可以让大模型工具提供方案验证其可行性或请求更多参考资料。

示例：“你上面提到的微服务架构优化方案，是否有相关的案例研究或文档可以参考？”

## 迭代与反馈

迭代提问：根据对方的回答，逐步深入问题，直到获得满意的解决方案。示例：“你提到的缓存策略解决了性能问题，但我们现在遇到了数据一致性

问题，如何解决？”

反馈：在问题解决后，提供反馈并明确表达大模型提供的解决方案是你需要的，这样有利于和大模型共同确认解决问题的方向。

示例：“你的建议非常有用，方案可行，根据你的建议，我已经成功优化了

XX 系统的性能。”

## 总结梳理

多轮对话过程梳理总结：根据和大模型的沟通交互，不断地明确共识和分歧， 多次迭代后，最终达成共识，将最终的方案总结梳理出来。

示例：在与大模型的多轮对话后，“经过与你的多反讨论，我们确定了以下解决方案：（1）使用 Redis 缓存优化订单服务的性能。（2）采用分布式锁解决数据一致性问题。（3）使用消息队列异步处理订单，进一步提高系统吞吐量。你看看还有什么其他的建议吗？没有问题我们就按照这个方案实施了。”

## 总结最佳实践

持续学习，通过不断实践，提升与大模型的交互能力，逐步掌握更高效的

Prompt 设计技巧。

乐于接受新的工具、新的技术，在使用的过程中自我总结并记录适合自己的最佳实践。

# 需要注意的问题

在运用大模型辅助软件开发实践过程中，需要提醒学生注意以下几个方面的问题。

* 囫囵吞枣，对大模型生成的内容不加分析思考，笼统接受，导致对生成的内容不求甚解，不能从中学习和提升。
* 不辨是非，不对大模型生成内容的质量（如正确性）进行辨析，发现不了大模型生成内容存在的问题，导致无法有效地解决实践问题。
* 滥用泛用，一遇到困难和问题就依靠大模型来解决，缺乏自身独立解决问题的实践，导致自身能力和技能得不到提升。

摘选自《大语言模型辅助的软件开发实践指南》，毛新军，杨康，国防科技大学计算机学院.