<UniGPT v2>

软件项目计划

**（简化版）**

版本 <1.1>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <26/06/24> | <1.0> | 初步设计项目计划 | 陈启炜、韦东良、叶懿芯、黄峻涛 |
| <29/07/24> | <1.1> | 根据老师指导建议进行修改，主要修改了4.3节“开发计划” | 韦东良 |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 范围 4

1.3 定义、首字母缩写词和缩略语 4

1.4 参考资料 4

2. 项目概述 4

2.1 项目的目的、规模和目标 4

2.2 假设与约束 4

2.3 项目的可交付成果 4

3. 项目组织 4

4. 项目计划 4

4.1 风险分析 4

4.2 方法和工具 5

4.3 开发计划 5

4.4 质量保证计划 5

4.5 项目沟通计划 5

4.6 培训计划 5

5. 附录 5

软件项目计划

# 简介

## 目的

本文档的目的是为UniGPT 2.0项目的开发提供一个清晰的概述和指导。它旨在明确软件项目的规模、目的、目标、约束和可交付成果，明确项目组织，进行项目风险分析，并按照风险制定迭代开发计划及其他各类计划。此文档将作为项目规划、实施和评估过程中的基础参考，以便项目团队和相关利益相关者理解整个项目的核心信息。

## 范围

此软件项目计划的范围是即将开发的UniGPT 2.0项目和已有的UniGPT项目。

UniGPT是《软件工程原理与实践》课程的大作业成果，是一个面向交大学生的LLM问答网站。UniGPT旨在帮助用户方便地使用LLM。网站提供prompt engineering的多种模板与技巧，用户可以自定义prompt并用于与LLM对话，也可以上传自定义prompt到应用市场，供其他用户使用。

UniGPT 2.0项目是基于UniGPT项目的进阶项目。

## 定义、首字母缩写词和缩略语

* **LLM**：大型语言模型（Large Language Models）
* **Prompt:** 指提示，即向模型提供输入以引导其生成特定输出的文本或指令，用于描述用户想要从模型获取的信息、回答、文本等内容。Prompt 的目的是引导模型产生所需的回应，以便更好地控制生成的输出。
* **Prompt Engineering**：提示工程，指优化输入LLM的提示以获得更好的输出结果的过程
* **API**：应用程序编程接口（Application Programming Interface）

## 参考资料

《软件工程原理与实践》沈备军等编著。

# 项目概述

## 项目的目的、规模和目标

本项目旨在开发一个基于提示工程技术的AI大模型社区。该社区将为大学生提供一个共享和合作的平台，促进AI技术的发展和应用。

项目规模包括以下几个方面：

平台开发：包括前端、后端和数据库设计。

模型整合：整合多种AI大模型，提供API接口。

用户社区：建立用户论坛、知识库和教程。

运维管理：包括服务器部署、安全措施和性能优化。

项目目标：

创建一个功能完备的AI大模型社区平台。

提供高效的提示工程技术支持和LangChain集成。

提供高质量的文档和教程，帮助用户快速上手。

实现稳定、安全和高性能的平台运行。

## 假设与约束

假设：

项目团队具备必要的技术能力和经验。

用户对AI技术有基本的理解和兴趣。

市场对AI大模型和提示工程技术有较高的需求。

约束：

人员： 项目团队所有成员均为开发人员，其中有1名产品经理、2名运营人员。

设备： 需要高性能服务器、存储设备和网络基础设施。

时间表： 项目计划在3个月内完成，从2024年6月开始，到2024年9月结束。

法律合规： 平台需遵守相关数据隐私和安全法规。

## 项目的可交付成果

|  |  |  |
| --- | --- | --- |
| **类别** | **可交付成果** | **描述** |
| 文档 | 系统测试报告 | 整个系统测试过程的结果，包括测试范围、测试用例、测试结果、缺陷统计和修复状态。 |
| 文档 | 系统测试用例 | 系统测试阶段执行的各项测试，包括输入、执行步骤、预期结果和实际结果。 |
| 文档 | 软件架构文档 | 系统的总体结构，包括模块划分、组件交互、技术选型和设计模式等内容。 |
| 文档 | 软件需求规约 | 系统的功能需求和非功能需求，明确了系统必须实现的各项功能和性能指标。 |
| 文档 | 软件项目计划 | 项目的目标、范围、时间表、资源分配、风险管理和沟通策略等内容。 |
| 文档 | 迭代计划 | 每个迭代周期的目标、任务、资源分配和时间安排，确保项目按计划逐步推进。 |
| 文档 | 迭代评估报告 | 每个迭代周期的工作成果、完成情况、存在的问题和改进建议。 |
| 文档 | 项目总结报告 | 回顾了整个项目的实施过程，评估了项目的成功度、经验教训和未来改进建议。 |
| 代码 | 前端代码 | 社区平台的前端代码，包括网页设计和用户界面。 |
| 代码 | 后端代码 | 社区平台的后端代码，包括API接口和数据处理逻辑。 |
| 代码 | 数据库脚本 | 数据库的创建和初始化脚本。 |
| 数据 | 初始数据集 | 包含平台启动所需的初始数据。 |
| 数据 | 日志数据 | 包括系统运行日志和错误日志。 |

# 项目组织

团队由陈启炜担任组长，主要负责项目管理。

组员：黄峻涛 主要负责：代码管理 测试

组员：叶懿芯 主要负责：设计 LangChain应用

组员：韦东良 主要负责：代码编写 代码优化

# 项目计划

## 风险分析

|  |  |  |  |
| --- | --- | --- | --- |
| **优先级** | **风险名称** | **风险描述** | **风险缓解措施** |
| 1 | 技术风险 | 开发人员对 langchain 技术不熟悉，可能导致开发效率低下 | 寻找类似项目，辅助参考项目实现；参考切实可行的教学视频 |
| 2 | 进度风险 | 因技术不熟悉，开发周期未知，可能导致最后进度难以赶上 | 将需求分级分层，按优先级实现敏捷开发 |
| 3 | 质量风险 | 若进展情况紧张，可能导致代码质量低下或项目性能较差 | 定期代码重构保证质量，定期进行性能测试，分出部分组员做性能提升工作 |
| 4 | 成本风险 | 开发过程中可能使用大量 LLM API 额度，或导致经费消耗较大的情形 | 采用更便宜的模型 API 做测试，最后运行时采用所需 API |

## 方法和工具

建模工具：Feishu, Powerdesigner

IDE: Visual Studio Code, IntelliJ IDEA

测试工具: JUnit, APIfox，Jmeter

版本管理工具: Git, 华为云 Code Arts

项目管理工具: Feishu，Wechat, Github

## 开发计划

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **迭代名称** | **起止时间** | **所缓解的风险** | **需完成的任务** | **需提交的成果** |
| 1 | 6.24 – 6.30 | 技术风险 | 1. 调研与分析需求，编写需求文档； 2. 进行界面原型的设计、实现与测试； 3. 初步学习主要工具链，搭建对应环境； 4. 重构与调试既存代码，提高性能，修复 bug； 5. 评审需求规约文档，根据评审意见进行改进。 | 项目计划 需求文档  界面原型设计稿 迭代计划 version - 0 代码 |
| 2 | 7.1 – 7.10 | 技术风险 | 1. 分析当前版本的需求； 2. 分析架构，设计架构，编写架构文档； 3. 架构原型的实现与测试； 4. 前后端分别开发，实现插件功能与知识库这两个功能点，进行云端部署，发布一个可运行的版本； 5. 探索测试框架，完成新功能的单元测试。 6. 评审架构文档，根据评审意见进行改进。 | 迭代计划 软件架构设计 version - 0.5代码 测试报告 |
| 3 | 7.11 – 7.19 | 技术风险、质量风险 | 1. 分析当前版本的需求； 2. 由单体架构转向微服务架构，进行微服务架构的设计； 3. 根据设计实现微服务，完成第一版应用； 4. 编写端到端测试及性能测试； 5. 对本次迭代进行评审，根据评审意见进行改进。 | 迭代计划version - 1.0代码 测试报告 |
| 4 | 7.19 - 9.13 | 技术风险，质量风险 | 1. 分析当前版本的需求； 2. 以性能测试的数据为指导，完善设计； 3. 采用Redis、Kubernetes等技术改善性能，代码实现； 4. 进行更加全面的测试，提升代码质量； 5. 对本次迭代进行评审，根据评审意见进行改进。 | 最终迭代成果 |

## 质量保证计划

**需求评审**：6.26 小组制定项目需求，并进行评审和修改

**设计评审**：6.26 – 6.27 叶懿芯完成初步前端设计，由小组成员评审

**代码评审**：  
 开发过程中，每次提交代码先由华为云 Code Arts 相关工具链一次评审，后提交 pull request 时由另外组员对本次提交进行评审；  
 每次迭代结束后，由全组成员评审，并进行对应的正确性测试和性能测试。

**单元测试 & 集成测试**：

每次负责开发的组员完成一个功能点后，负责测试的组员完成对应测试框架并进行测试  
**系统测试**：  
 每次 release 后，负责测试的组员应完成系统测试

## 项目沟通计划

内部沟通：

涉及2-3名的任务由组员间直接沟通，将沟通结果以文档形式同步至飞书并告知所有组员。 不定期召开，时间控制在5~15分钟

每日立会：每日9:30进行立会 定期召开

涉及4名组员的任务，视任务情况不定期召开全体组员会议，由一名组员负责会议纪要并同步至飞书。 不定期召开，时间控制在5~25分钟

向助教/老师汇报沟通：

每日立会：助教参与，获知团队最新信息。 定期

迭代汇报：向助教/老师汇报迭代成果 不定期

## 培训计划

LangChain技能培训：

成员：叶懿芯，陈启炜

方式：线上学习网络课程

预定日期：6月25日至6月27日

Git技能培训：

成员：黄峻涛

方式：参考《GitPro》、优秀方案

预定日期：6月24日至6月25日

成员：陈启炜，韦东良，叶懿芯

方式：小组成员黄峻涛知识分享

预定日期：6月26日

# 附录