**迭代评估报告**

　　　　　　　　　　　　　　　　　　评估日期：

|  |  |  |  |
| --- | --- | --- | --- |
| 组号 | 18 | 项目名称 | UniGPT |
| 迭代名称 | 技术原型迭代 | 实际起止日期 | 3.19-5.12 |
| 任务完成情况   |  |  |  |  | | --- | --- | --- | --- | | No | 任务 | 进度 | 质量 | | 1 | 后端API设计 | 完成 | 良好 | | 2 | 数据库设计 | 完成 | 良好 | | 3 | 逻辑架构视图（作业） | 完成 | 完备 | | 4 | Spring Boot实现后端API | 完成 | 优秀 | | 5 | 后端封装大模型接口 | 完成 | 完备 | | 6 | 连接前后端 | 完成 | 良好 | | 7 | 前端markdown渲染 | 完成 | 优秀 | | 8 | 接入jaccount单点登录 | 完成 | 优秀 | | 9 | 使用WebSocket实现机器人对话 | 完成 | 优秀 | | 10 | 使用C语言开发Linux图片服务器 | 完成 | 良好 | | 11 | 小组内部评审和改进技术原型 | 完成 | 完备 | | 12 | 编写《迭代评估报告》 | 完成 | 完备 | | 13 | 设计模式作业 | 完成 | 优秀 | | 14 | 修复问题和漏洞 | 进行中 | 良好 | | 15 | 单元测试和端到端测试 | 进行中 | 良好 |   **完成的功能**：  创建新的机器人  修改机器人信息  查看与机器人的历史记录  与机器人进行对话  修改、添加、删除机器人的提示词表单  访问机器人市场并交互  图片上传和图片存储功能  接入jaccount单点登录 | | | |
| 评审/测试的结果：（执行了哪些评审和测试？评审和测试的结果如何？）  **API接口文档：**小组集体对API接口进行了讨论，并提出修改建议  评审结果：API文档总体合理可行，创建对话历史、查询和修改表单等结构的响应体需要修改，需要增加获取自身用户信息的接口。  **数据库设计：** 小组每个人负责一部分的数据库表设计。  评审结果：数据库设计合理。  **API接口实现：**小组每个人负责API文档的一部分，使用Java Spring Boot进行实现并使用Apifox进行测试  评审结果：代码实现基本正确，部分接口在实现的过程中进行了微调。  **代码评审**  **组内评审：**小组其他成员对代码进行评审，加强代码规范  评审结果：代码基本符合开发规范，部分不符合规范的代码已经进行重构。  **自动化代码质量检查：**使用Checkstyle进行代码质量检查  评审结果：代码质量合格  **功能测试**  **单元测试：** 针对后端代码的每一层进行单元测试（controller, service层），在代码发生修改后运行单元测试，保证代码的正确性不变。  测试结果：大部分接口和服务符合预期，能够较好地应对特殊情况，少部分函数鲁棒性不足，存在返回结果不正确，抛出未处理异常等bug  **自动集成和自动部署：** 利用Github Action进行CI/CD，保证合并到主分支代码的正确性。  测试结果： 合并到主分支的代码正确。  **用户接受度测试**  用户体验：邀请组外其他同学体验应用功能，收集反馈，确定应用是否满足用户的需求和期望  测试结果：能满足用户的大部分需求和期望。少部分组件交互逻辑不够自然，仍可新增一些功能改善用户体验  **兼容性测试**  浏览器兼容性：确保应用在不同的浏览器和版本上能够正确运行，因条件限制，目前在Edge, Safari，FireFox，Chrome，Arc的最新版本浏览器进行测试  测试结果：兼容性良好  设备兼容性：测试应用在不同设备的表现，目前在Windows、Mac、Linux笔记本上都进行了测试  测试结果：兼容性良好  跨平台尝试：使用Tauri开源框架对Web前端进行了打包，PC客户端支持Windows,MacOS,Linux，手机端支持Android, iOS。  测试结果：具有可行性，布局仍需要优化。  **国际化和本地化测试**  多语言支持：测试应用是否能够支持多种语言，内容是否正确地根据用户的语言偏好显示。  测试结果：能支持中英文切换，需要加强对中文的支持，考虑后续封装成语言包，支持更多语言。 | | | |
| 问题、变更和返工：（遇到的问题、发生的变更、是否需要返工等）  1. 问题：页面BotChatPage请求api流程不清晰，每个人理解不同  返工：通过一个人撰写文档，以更加清晰地描述需求。  2. 问题：后端的代码风格不一致  返工：小组讨论开会，统一商议后端开发规范，并按照此规范重构代码。  3. 问题：版本管理混乱，包括但不限于分支命名不规范，提交记录混乱，分支同步不统一。  返工：一个人撰写Git协作文档，规范了分支命名、开发工作流等事宜。   1. 问题：BotChat页面中，初始设计的API文档不符合前端的需求。   变更：前后端进行联调，确认后端API的最终方案。  返工：修改后端API的实现。   1. 问题：存储图片存在多个实现方案。   返工：开会讨论后，商议存储图片的最终方案并实现。   1. 问题：评论区无法显示用户的名称。   变更：修改后端响应JSON字段名称   1. 问题：当前端请求后端遇到异常时，没有出错的提示。   变更：使用Mui的提示组件，在请求后端出错时弹出错误的提示。  返工：重构部分前端代码，以满足错误提示和处理的需求。   1. 问题：创建机器人页面中，解析提示词中的变量存在重复问题。   变更：修改解析字符串的算法，以正确解析提示词中的变量名。 | | | |
| 经验和教训：   1. 后端接口设计应当遵循简洁直观的原则，不要过度设计，以保证接口易于理解和使用。 2. 后端开发也需要考虑用户需求和使用习惯，保证后端服务的稳定性和性能。 3. 借鉴现有设计和功能：参考已有的设计和功能可以减少重复开发工作，提高效率。 4. 后端开发过程中问题和变更是正常现象，需要灵活应对并及时迭代返工，以确保后端应用符合预期功能和效果。 5. 后端开发同样需要进行兼容性测试，确保后端服务在不同环境下都能正常运行。 6. 代码质量的保证：及时进行代码评审，采用自动化代码质量检查工具，以提高代码的可读性和可维护性。 7. Git分支管理的规范：遵循良好的Git分支管理流程，确保团队协作顺畅，代码版本控制清晰可控。 8. 团队协作和沟通：与组员之间的协调沟通至关重要，定期开会交流进展、解决问题，建立团队合作氛围。 | | | |