**技术原型迭代评估报告**

　　　　　　　　　　　　　　　　　　评估日期：2023.7.6

|  |  |  |  |
| --- | --- | --- | --- |
| 组号 | 2 | 项目名称 | “Testland”--在线代码测评平台 |
| 迭代名称 | 技术原型迭代 | 实际起止日期 | 2023.6.27-2023.7.6 |
| **任务达成情况**  **项目基本框架搭建**   * 前后端基本框架搭建完成，后端能够正确处理请求与提供数据，前端能正确发送请求和解析展示数据。 * 多数基本功能已经实现。 * 前端页面设计美观实用，符合用户操作习惯。 * 前后端成功实现跨域通信。   **远程评测功能**   * 后端能够正确将前端用户上传的代码保存到指定路径，能正确取到测试用例的地址以及测试的基本信息。 * rabbitmq能够将测评所需信息正确发送到服务器上的消息队列，测评部分能正确取到消息队列中的测评信息并依次处理，并通过另一个消息队列将测评结果返回给后端，在后端实时监听并更新数据库。 * 评测模块在docker容器中运行评测代码和用户代码，保证了评测机环境的安全。   **前端**   * 基于chakra ui库实现响应式布局和深浅模式切换。 * 允许用户以markdown形式发帖，题目详情以markdown形式呈现。 * 已完成用户登录、注册、管理员界面以及主页、代码评测、题库、排行榜、讨论区等界面，搭建基本框架。   **后端**   * 问题、讨论、用户等信息的增删查。 * 答题数据统计，发送测评结果等。 * 连接评测模块，完成消息队列发送、接收及后续数据库操作等。 |  |  |  |
| **问题、变更和返工**  **问题**  **评测流程设计**  当前采用的评测流程是“从指定路径读取用户代码-将评测程序、测试用例和用户代码复制到docker进行评测-完成评测，将评测结果发送至消息队列中间件”，对于每一段用户代码，都需要从生成dockerfile开始，构建docker镜像和容器，前期准备时间较长，资源利用率较低，影响了评测效率，后期拟优化评测流程。  **环境配置**  本次迭代过程中，对于docker环境的配置出现了一些问题，导致了进度的延误。一方面，用于开发的电脑上原本已经配置好的docker环境出现了cannot start docker engine的问题，且根据详细错误信息进行的搜索没有在网上找到太多可用解决方案，最终对docker进行了重装解决，另一方面，本地开发电脑的系统为windows，而部分代码实际运行环境为linux（WSL, docker），系统的切换导致开发环境经常出现一些错误。  **变更**  **用户代码内容安全性过滤功能**  由于进度问题，没有来得及实现用户代码安全性过滤功能。但是由于所有的用户代码只会在容器环境下执行，对于服务器自身安全威胁较小，因此计划变更到下一次迭代前完成。  **并发处理**  通过消息队列中间件，评测部分可以异步处理并发的代码提交请求，但在处理效率上较低，原因是在开发过程中评测部分尝试使用线程/进程进行并行化评测，但是一旦采用以上方案就会造成程序卡死，原因暂未成功排查。下一阶段考虑提高评测效率。  **用户友好设计**  本次迭代实现的用户上传题目过程较繁琐，可增加压缩包上传题目选项。 |  |  |  |
| **经验和教训**   * 进度管理。确保明确的项目计划和里程碑，以便在整个迭代过程中有清晰的目标和时间表；使用项目日志等来跟踪任务、进度和问题，每日立会内容及时记录；在迭代开始之前，评估和分配合理的时间和资源，以确保能够按时完成计划。 * 版本管理：使用版本控制系统（Git）管理代码和追踪变更，本项目开发过程中按照成员和功能对代码仓库进行了分支的划分，正式开始开发前保证所有成员学会了git的基本操作，并及时提交修改后的代码，保证了团队信息互通。 * 分工合作：在团队中明确定义角色和责任，并确保每个成员都清楚自己的任务和目标；每日例会，日常进行沟通和协作，分享进展、解决问题和调整计划。 * 了解用户需求并进行优先级排序，以便在迭代过程中专注于最重要和具有最高价值的功能。 * 迭代结束后，进行详细的回顾和总结，包括技术、流程和合作方面的经验教训，以便在下次迭代中做出改进。 |  |  |  |