**项目总结报告**

日期：2021-09-13

|  |  |  |  |
| --- | --- | --- | --- |
| 组号 | 20 | 项目名称 | 交我赚-基金分析系统 |
| 编程语言 | Java, JavaScript, CSS, TypeScript, LESS, Python, SQL, Shell, HTML | 开发平台和框架 | 框架React Native, Spring Boot, Ant Design Pro, TensorFlow, Scrapyd, Spring Security.  IDE: IDEA, WebStorm, PyCharm  协作工具：Git，Huawei DevOps |

|  |  |
| --- | --- |
| **软件需求特性** | |
| 1. 是否实现了项目立项时的所有需求？列出实现的新增需求和未实现的需求。   实现了立项时学院要求的所有需求，还新增了以下需求：   * 允许用户创建公开和私有比赛，允许用户加入比赛； * 允许用户将基金加入比赛，允许用户查看自己比赛的剩余资产和全赛区资产排名； * 允许管理员监控系统超30种运行状态，进行多项服务操作；  1. 采用哪种架构风格？哪些设计模式？  * 采用了C-S架构风格用于开发App和B-S架构风格用于开发Web端。 * 设计模式包括：   + 内存缓存采用了单例模式；单例模式还包括Spring Security等配置上的Bean & Components等；   + 代理模式：机器学习对于数据过少的基金直接套用模型。预测时只关心这个模型能不能满足基金的需求，而不关心模型是否是该基金训练出来的；   + 门面模式：基金的数据复杂繁复，来自五张以上的MySQL表。而用门面模式我们将其打包为对象储存在内存中，只需要提供基金代码就能获取全部数据。  1. 技术方案有哪些亮点？ 2. 强大的爬虫引擎，采用微服务架构，数据爬取稳定、服务易用。 3. 流畅美观的UI，由React Native开发，用户体验良好。 4. 任务调度拓展了线程池功能，易于定位线程服务状态，支持按工作日，按交易时间执行。后台支持每日默认的任务调度。管理测试中台功能完善，可以对于后台服务进行管理、监测系统运行状态。 5. 对于高并发的请求，使用Java Hash Map缓存页面；将分钟级更新的页面放在内页中，如非默认显示的Tab中，单独向服务器发送请求；对数据进行预计算，将update操作放在闲时进行，在业务高峰时期仅留存查询操作；对查询结果进行枚举缓存，进行Hash后使用二分 | |
| 查找和数据库查找分流优化查找性能；针对用户经常访问，且时效性不强的数据，如首页的涨幅排名等，在前端使用Async Storage进行缓存，防止重复调用后端接口，减轻服务器压力。   1. 机器学习模型功能强大，针对不同基金建立不同模型，预测结果有较高准确度。 2. 采用Spring Security进行精细化的权限控制，使访问更加安全。 | |
| **项目组成员对项目的贡献度（%）** | |
| 注：总计100% | |
| **软件度量** | |
| 软件代码行数（不包括注解行、空行和复用代码）： | 48955 |
| 复用他人代码行数： | 19876 |
| 类的个数： | 358 |

|  |
| --- |
| **经验、教训和建议** |
| 1. 测试驱动开发：我们在进度风险的压力下仍然将测试压到后50%的时间进行，这导致部分bug发现过晚，导致压力过于集中，可能出现软件质量下降； 2. 在同样的组内进行协作应该保证环境一样。这次开发过程中前后端的环境不同造成了大量的开发困境，有一些中间版本出现了只有Mac OS或者Linux或者Windows才能跑起来的情况，这严重拖慢了开发。所以持续集成和交付是很重要的，使用Docker等环境也是解决方案。或者说团队内同一开发系统也可以大部分解决这样的问题； 3. 使用好Scrum开发对提高团队性能有重要意义。在团队高效立会的背景下，经常在白板上写下自己准备做什么和正在做什么对团队协作很重要，同时根据工作项追踪Git提交也能让我们时刻了解项目的进度。 |