**项目总结报告**

日期：2021/1/7

|  |  |  |  |
| --- | --- | --- | --- |
| 组号 | 第十组 | 项目名称 | 时间管理app 一心 |
| 编程语言 | 后端：Java  前端：dart | 开发平台和框架 | 开发平台：IntelliJ IDEA 2019.3.3 x64和Android Studio  框架：基于Spring Cloud的微服务架构 |

|  |  |
| --- | --- |
| **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。  实现了绝大部分的功能性需求，实现了易用性、可靠性、高性能、可支持性和设计约束等非功能性需求。  新增需求：青蛙的养成功能  未实现的需求：分享功能  2.采用哪种架构风格？哪些设计模式？  架构风格：  我们将采用微服务架构，把一个大型的单个应用程序和服务拆分为数个甚至数十个的支持微服务，它可扩展单个组件而不是整个的应用程序堆栈，从而满足服务等级协议，用一些功能比较明确、业务比较精练的服务去解决更大、更实际的问题，在分散的组件中使用云架构和平台式部署、管理和服务功能。我们主要分为以下几个服务：登录注册服务、安全服务、好友社交服务、时间管理服务和闹钟服务，每个服务有着自己的数据库、可独立地进行开发、管理和迭代。      设计模式：  1.工厂方法Factory Method  任务类对应着很多小游戏实例，他们的创建和使用是可以分离的，故可以使用工厂方法实现task的创建和使用的分离：  C:\Users\yy\AppData\Roaming\Tencent\Users\1648645367\QQ\WinTemp\RichOle\PT88U)J`{]]_~(Z6T1V`OVX.png  Task Factory仅负责给出GameFactory必须实现的接口，不负责具体游戏的创建，将具体创建的工作交给子类去做，而不接触哪一个产品类应当被实例化这种细节。  2.原型模式Prototype  学习报表的基本模型是一样的，不同的是各个用户具体的数据，故可以使用原型模式来实现学习报表的快速创建： C:\Users\yy\AppData\Roaming\Tencent\Users\1648645367\QQ\WinTemp\RichOle\JFBL2(OM@S1HOU{QN~WX3TG.png  图中，StudyLog充当具体原型类，Object类充当抽象原型类，clone()方法为原型方法。  3.技术方案有哪些亮点？  1）养成系统和激励机制  1）实现了图像识别功能  2）实现了重力感应功能  3）实现了进程检测功能  4）实现了微服务架构  4.是否做了单元测试？是否做了系统功能测试？是否做了性能测试？是否做了兼容性等其他非功能测试？  完成了单元测试、系统功能测试、性能测试、兼容性测试、易用性测试、可靠性测试等测试。 | |
| **项目组成员对项目的贡献度（%）** | |
| 注：总计100%  沈佳威：  江玙璠：  朱益成：  陈嘉轩： | |
| **软件度量** | |
| 软件代码行数（不包括注解行、空行和复用代码）： | 13000+ |
| 复用他人代码行数： | 500+ |
| 类的个数： | 91（后端）+63（前端） |

|  |
| --- |
| **经验、教训和建议** |
| 1.在迭代的计划上，应该经常对迭代计划的任务安排进行重新评估，结合项目实际的开发进度和需求，调整、细化时间安排，让计划能够更好地指导开发工作的进行。  2.在使用复杂的组件前，可以先单独做一个demo，预览一下实现效果同时熟悉使用方法，评估一下开发的复杂性与实用性，然后再决定是否实际应用到项目中。在开发前期或者时间紧张的情况下，合理地使用简单的组件进行替换。  3.要合理分工，使得小组整体的效率高于四个个人的效率。  4.要加强小组内的沟通，及时对完成的任务进行评审，明确接下来的任务，尽量不产生冗余  5.编写迭代计划及其他文档时，考虑要尽量详细完善，不能因为自己是开发小组的成员，对项目比较熟悉，就在一些需要明确指定的地方简略言之，导致给项目的迭代引入风险。  6.在迭代的计划上，应该经常对迭代计划的任务安排进行重新评估，结合项目实际的开发进度和需求，调整、细化时间安排，让计划能够更好地指导开发工作的进行。  7.在架构设计时，要考虑到自身项目的实际情况，选择适合自己的架构设计模式。 |