**迭代评估报告（简化版）**

　　　　　　　　　　　　　　　　　　评估日期：2020/11/16

|  |  |  |  |
| --- | --- | --- | --- |
| 组号 | 1 | 项目名称 | 霍格沃茨地图（手机导游APP） |
| 迭代名称 | 技术原型迭代 | 实际起止日期 | 2020/10/15-2020/11/17 |
| 任务达成情况：（完成的任务、实现的功能、进度、质量等）   1. 迭代计划：在总迭代计划的基础上，对于每周的迭代计划进行细化，形成每周或者每两周的迭代计划细化表。 2. Vision文档：在第一次迭代Vision 文档的基础上，按照老师以及助教的建议进行修改完善，形成第二次迭代版本的Vision文档。 3. 用例模型：在第一次迭代的用例模型的基础上，结合老师助教的建议以及第二次迭代实施过程中发现的一些问题，对用例模型进行修改完善，形成第二次迭代版本的用例模型。 4. 软件需求规约：在第一次迭代软件需求规约的基础上，结合实践项目情况以及老师助教的建议，进行了一些调整与修改，形成了第二次迭代版本的软件需求规约。 5. 软件架构文档：首先由组员分工完成软件架构文档的初稿，经过小组评审与修改之后继续完善，并结合小作业上的点评进行修改调整，形成了第二次迭代版本的软件架构文档。 6. 编程规范：前端Dart语言编程规范，参考Dart官方文档，以及其他官方文档，形成了自己的Dart语言编程规范。后端则采用Google Java编程规范。 7. UML分析与设计模型：由小组讨论形成初步的UML类图，在完成初稿后进行小组评审，并进行修改和完善，形成最终版本的UML类图以及分析与设计文档。 8. 技术原型代码：前端使用Flutter框架由前端小组进行开发，后端采用Spring Boot框架由后端小组进行开发，同时搭建MongoDB数据库，阿里云OSS对象存储服务，以及时序数据库，同时进行与高德地图相关的地图地位，周边，导航等Demo的开发， | | | |
| 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？）   1. 进行了所有相关文档如：UML分析与设计，编程规范等的小组评审，针对文档的内容格式等进行了调整，比如UML分析中的一些设计模式的调整，以及编程规范中的一些要求的修改，从而形成最终版本的相关文档。 2. 进行了前端界面的评审，对不符合要求的界面进行重新设计实现，比如对于个人中心不必要信息的删除，以及景点展示方式的调整，最终得到美观并符合项目要求的前端界面。 3. 进行了技术原型demo的评审，对于与高德地图相关的技术进行小型项目实验，开发人员向小组进行展示，由小组提出意见并改进，比如对于定位需要调起陀螺仪监测方向改变，从而得到符合要求的demo。 | | | |
| 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. 对于景点展示方式的调整，由首页地图的搜索页，转移到专门的附近页面，便于用户使用以及模块化开发。需要对前端结构进行一定的调整，但是返工规模不大。 2. 对于定位方式的调整，从最初的一次定位到连续定位并调用陀螺仪跟随方向，更加符合用户需要。需要对代码进行再加工，增加陀螺仪调用等，是原基础上的改进，不需要进行返工。 | | | |
| 经验和教训：   1. 将最大风险放在最高优先级考虑，这样能够减少很多不必要的返工，比如此次项目提前尝试前端的开发与相关技术的小型项目测试，对于Android原生以及Flutter都进行了尝试，从而最终确定使用Flutter进行开发，避免了后期大量返工。 2. 经常进行小组评审与讨论，在工作进行中不断的与小组成员交流工作进展与方向，能够及时纠正错误，发现更多的想法，并且及时地进行修改与调整，这样能够避免一项任务进行到末尾才发现中间环节的问题而进行大量的返工。 3. 每周以及每天都应该细化迭代计划，制定每项任务的结束时间，这样能够使得所有任务按计划有序进行。 4. 在遵循计划的同时，也应该根据情况动态调整迭代计划的安排，有些任务超过预估难度可以适当增加时间以及负责人员，而有些任务与预期相比较为简单则可以适当减少时间并让负责人员参与到其他任务中。 | | | |