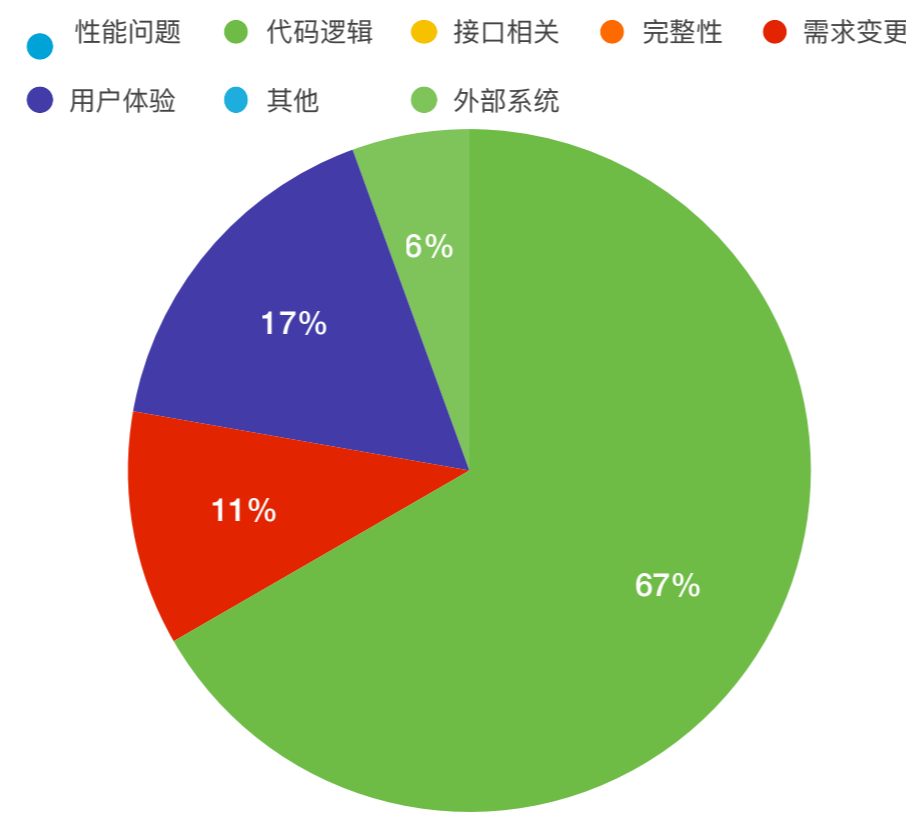


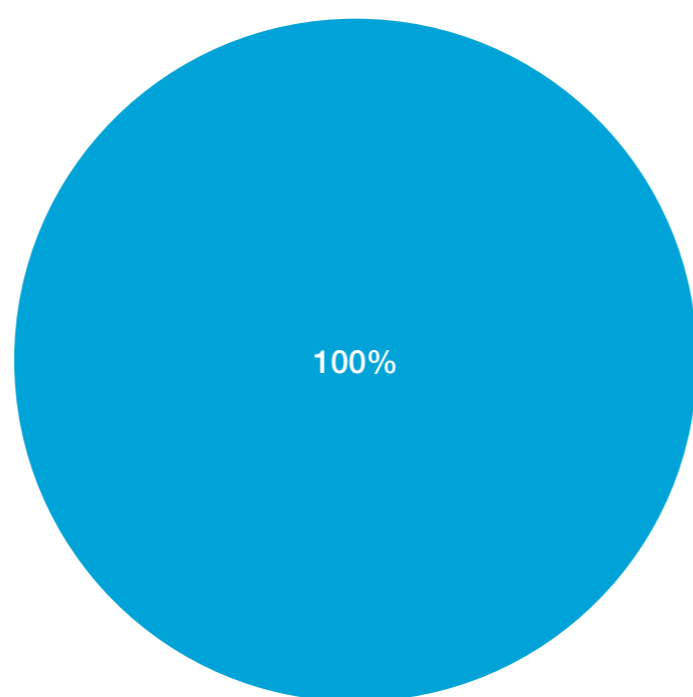
BUG类型			
BUG类型	数量	BUG类型细分	数量
性能问题	-	数据越界	-
		数据类型	-
		逻辑问题	-
		版本兼容	-
		未知原因	-
代码逻辑	12	逻辑错误	11
		网络机制	1
接口相关	-	接口问题	-
		环境问题	-
		请求参数问题	-
完整性	-	考虑不全 (数据联动)	-
		考虑不全 (网络差)	-
		考虑不全 (多用户)	-
		考虑不全 (版本兼容)	-
		边界测试	-
		功能完整性	-
需求变更	2	需求文档不明确	1
		需求变动	1
用户体验	3	交互处理	-
		样式	3
		兼容性	-
其他	-	-	-
外部系统	1	1	1



【性能问题】类型细分

BUG类型细分	数量
数据越界	0
数据类型	0
逻辑问题	0
版本兼容	0
未知原因	0

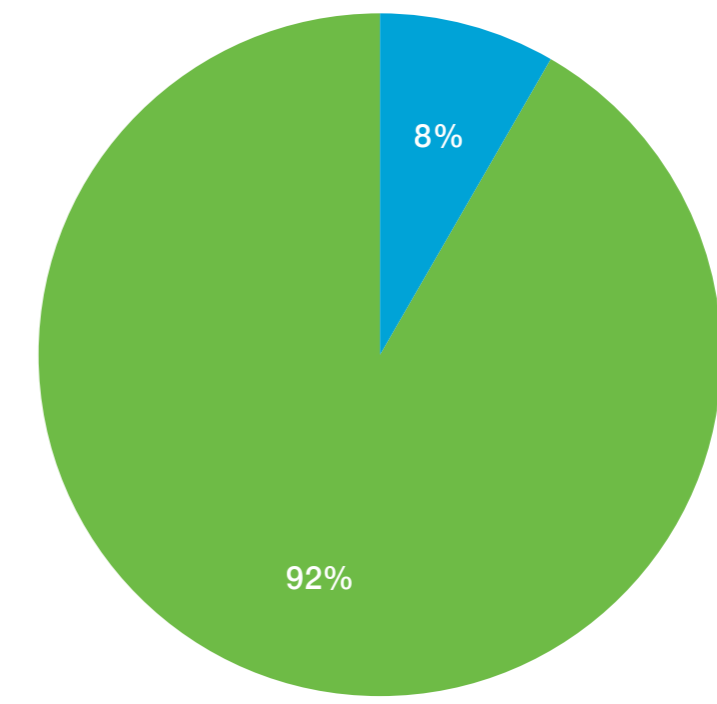
Legend: 未知原因 (Unknown), 版本兼容 (Version Compatibility), 逻辑问题 (Logic Problem), 数据类型 (Data Type), 数据越界 (Data Boundary)



【代码逻辑】类型细分

BUG类型细分	数量
逻辑错误	11
网络机制	1

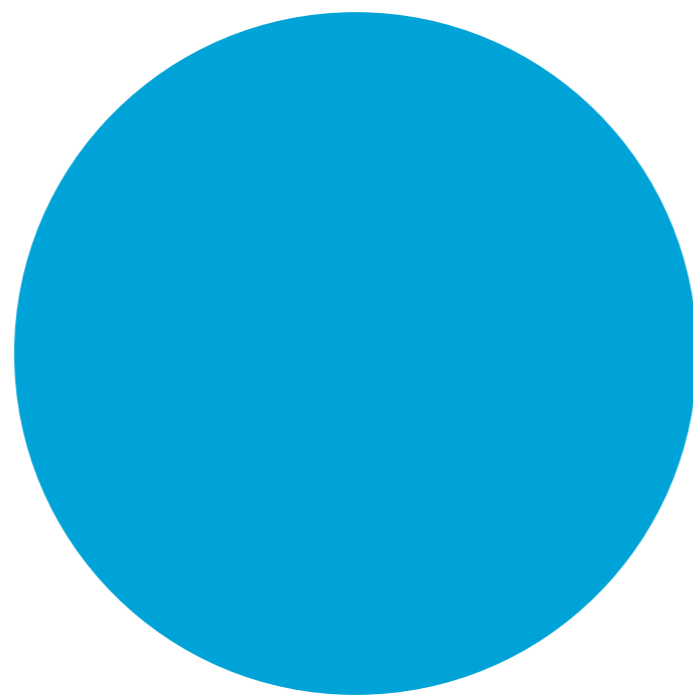
Legend: 网络机制 (Network Mechanism), 逻辑错误 (Logic Error)



【接口相关】类型细分

BUG类型细分	数量
接口问题	-
环境问题	-
请求参数问题	-

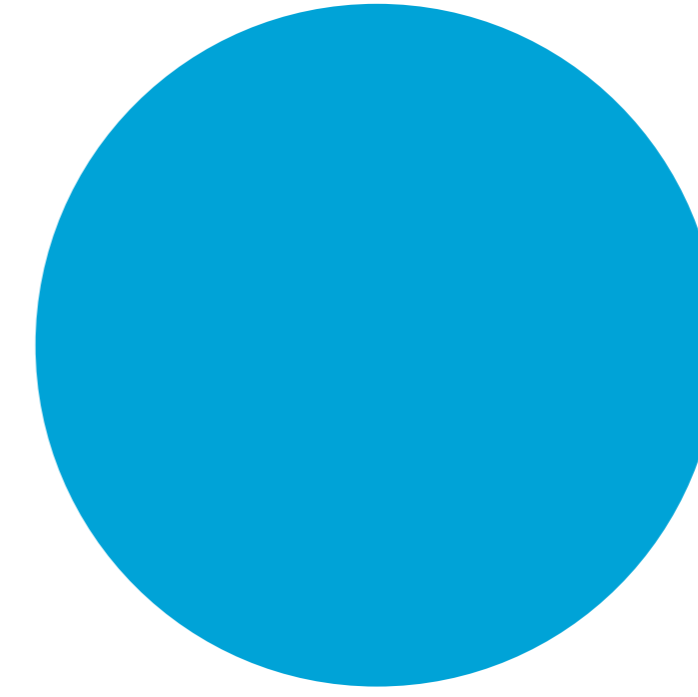
Legend: 请求参数问题 (Request Parameter Problem), 环境问题 (Environment Problem), 接口问题 (Interface Problem)



【完整性】类型细分

BUG类型细分	数量
考虑不全 (数据联动)	-
考虑不全 (网络差)	-
考虑不全 (多用户)	-
考虑不全 (版本兼容)	-
边界测试	-
功能完整性	-

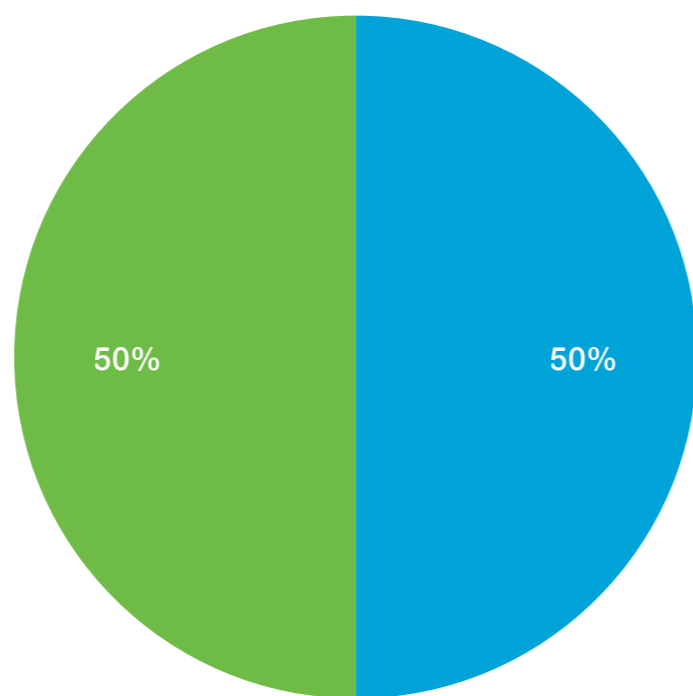
Legend: 边界测试 (Boundary Test), 考虑不全 (版本兼容) (Consideration Incomplete (Version Compatibility)), 考虑不全 (多用户) (Consideration Incomplete (Multiple Users)), 考虑不全 (网络差) (Consideration Incomplete (Network Quality)), 考虑不全 (数据联动) (Consideration Incomplete (Data Linkage))



【需求变更】类型细分

BUG类型细分	数量
需求文档不明确	1
需求变动	1

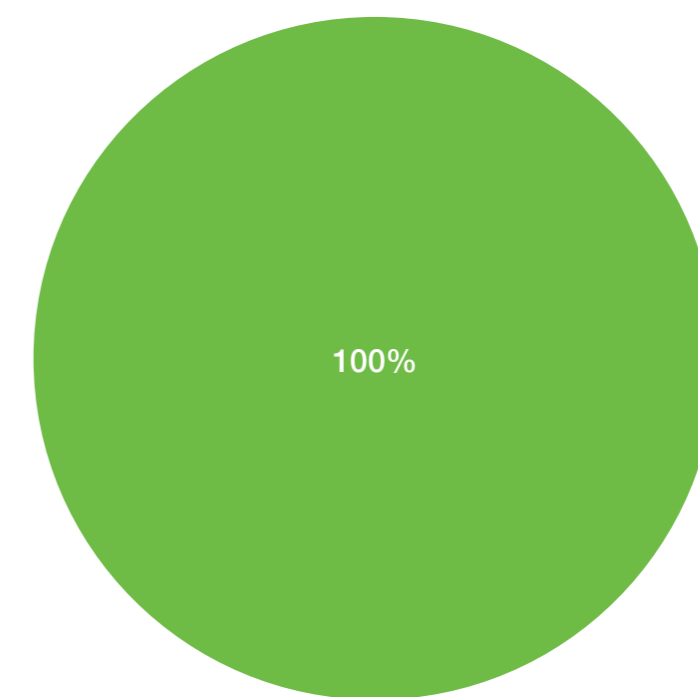
Legend: 需求变动 (Requirement Change), 需求文档不明确 (Requirement Document Unclear)



【用户体验】类型细分

BUG类型细分	数量
交互处理	-
样式	3
兼容性	-

Legend: 兼容性 (Compatibility), 样式 (Style), 交互处理 (Interaction Handling)



日均bug产生率

姓名	有效工作日	IOS 开发的BUG总数	BUG数/工作日
潘思嘉	12	8	0.666666666666667
林俊炳	12	9	0.75
胡志祥	2	0	0
总人日	26	17	0.653846153846154

bug激活情况

BUGID	激活次数	类型	所属人 (以首个修复者为淮)
95972	3	RN	林俊炳
96227	1	RN	林俊炳
96100	3	vue	潘思嘉

上线一周crash情况

日期	CRASH数	CRASH率
2017-07-31	1	0.0011
2017-08-01	0	0
2017-08-02	2	0.0006
2017-08-03	1	0.0003
2017-08-04	0	0
2017-08-05	1	0.0003
2017-08-06	1	0.0003

项目总结

姓名	总结
潘思嘉	<p>低估了工作量。和因为上个项目的影响，推迟了这个项目开始开发的时间，导致后期不够时间做自测，qa 找到很多不该出现的问题。</p> <p>项目开发经验不够，没有考虑到接口的不合理，例如没有按数据类型拆分接口，没有约定字符转码。</p> <p>由于项目紧张，没有对vue 的渲染进行优化，接下来会找机会尝试吴建斌同学的建议。</p> <p>由于初期技术分工和设计不完善，导致后期对各个开发都频繁提出修改需求，开发节奏有点乱，例如接口的跨域访问，参数变更，m 和vue 的传参改变，这些都建议下次找更熟悉项目的同事帮忙把关。</p> <p>Vue 这边第一次使用了vue cli 来处理，做到了比较好的项目代码管理，代码规范和最终代码的压缩，提高了开发质量。现在vue的代码放在了svn 的分支里面，需要教导其他开发的同事如何使用。</p> <p>记录一个下个版本的优化项，1. 如果终端页的内容比较大，vue 的渲染会长到1秒，而这一秒是没有那个loading 的咖啡杯的，体验不好。</p>
林俊炳	<p>技术实现：1、以RN和Vue数据分离来看，这种方式还是值得推荐的，就是一些细节可能要注意，比如稍微长的loading事件、加载本地HTML文件无法监听RN事件、两者传参的格式转换 2、后台不同，做出的接口有异，不能一味依赖已有代码逻辑，很有可能就不适用最新功能 3、涉及条件判断而且有number的，都做类型转换，设计师模块坑最多次数的问题 4、当前这种一个local，一个remote的vue开发模式，构建的时候太容易出错了，需要完善，最好维护一个vue仓库 沟通：波哥提的，还是面对面沟通效率高</p> <p>在有了居有范的技术实践之后，设计师模块开发工作算是比较顺利。不过由于终端页的技术实现跟之前有所不同，倒也是遇到了另外一些问题。 本次设计师模块的终端页，加载的是本地HTML文件，由Vue搭建基础网页骨架，并构建到APP的RN目录里面。数据由RN获取、缓存，达到页面与数据分离效果。除去必要的WebView协议处理，下面罗列的是遇到的一些问题及处理方法：1) RN调用HTML文件中的JS方法，与居有范的网页加载完后注入脚本不同，加载本地HTML文件的话，需要先把注入脚本添加到HTML文件中，由Vue手动填写即可；2) RN数据请求完毕，执行JS方法传递数据到Vue，需要在Vue加载完后，通过协议通知RN发起请求，否则会出现Vue模板没有加载完毕，就发起请求，而实际还没注册事件监听，导致执行JS方法无效的问题；3) 终端页的LoadingView显示由RN控制，隐藏则应由Vue加载完后，通过协议通知RN关闭。 按bug分布情况来看，出现较多RN与Vue协议交互导致的问题，如Vue传递数据不全、RN协议处理不全等；另外还有个别接口使用有误、参数类型不匹配的问题，此次也存在需求新增和变动情况。总结下来，技术实现仍是次要，更需要被重视的，可能还是与接口、前端同事的沟通协作，另外也是需要仔细地阅读接口文档，避免接口使用有误的问题出现。</p>
胡志祥	<p>家居杂志这次的版本需要原生开发的内容不多，包括一个更多标签的页面以及tabbbar的替换，整体来说开发任务不多。出问题的还是在测试阶段，由于RN模块开发时间紧，导致接口的测试情况不乐观（接口联调跟测试同时进行，以后真的不要这样做，很坑开发），直接带来的就是bug数往上飙。一个版本跟下来，RN与原生相比优点明显，缺点也同样明显，可优化的地方还是挺多的，未来可期。</p>