**迭代计划**

　　　　　　　　　　　　　制定日期：2021-3-29

|  |  |  |  |
| --- | --- | --- | --- |
| 组号 | 15 | 项目名称 | 交我唱APP |
| 迭代名称 | 技术原型迭代 | 计划起止日期 | 2021-3-29~2021-5-4 |
| 任务、进度安排和人员分配：   |  |  |  |  | | --- | --- | --- | --- | | No | 任务 | 起止日期 | 人员 | | 1 | ·考察音频指标。  ·复现音频分析库并给出一个教程。  ·面向对象分析，识别概念类，建立概念类图。  ·考察音频相似度指标。  ·给出一版音频相似度评分demo 给出音频知识点教程。 | 2021-3-27~2021-3-28  2021-3-29~2021-3-30  2021-3-31~2021-4-1 | 鲍辰 | | 2 | ·配置ffmpeg模块的环境，顺便看看K歌软件还需要些什么算法。  ·尝试使用ffmpeg模块  ·尝试使用已有的人声加强算法，探究回声消除算法。 | 2021-3-27~2021-3-28  2021-3-29~2021-3-30  2021-3-31~2021-4-1 | 乐辰阳 | | 3 | ·组内讨论分配三部分工作，找云服务器。  ·实现音视频的传输，并搭建音视频存放的数据库。  ·跟前端进行进一步对接。 | 2021-3-27~2021-3-28  2021-3-29~2021-3-30  2021-3-31~2021-4-1 | 梁晨 | | 4 | ·引入录音组件并修改风格。  ·导出音频文件并修改风格。  ·引入jAccount相关组件，准备通信。  ·搭建社区模块 | 2021-3-27~2021-3-28  2021-3-29~2021-3-30  2021-3-31~2021-4-1 | 郑博洋 | | 5 | ·配置音频消音环境，复现音频消音代码。  ·将消音代码部署到后端。 | 2021-3-27~2021-3-28  2021-3-29~2021-3-30  2021-3-31~2021-4-1 | 朱家琛 | | 6 | ·完善前端代码，引入录音组件导出音频，完善页面对录音和播放的控制。  ·引入jAccount相关，搭建用户账号相关部分，引入社区账户与jAccount相关联。  ·视进度引入歌曲打分算法、人声提取算法。  ·整合前端功能，与后端进行联动。 | 2021-3-29~2021-4-15 | 郑博洋 | | 7 | ·完成网易云api的集成。  ·完成消人声算法的集成。  ·完成跟前端所有现有功能的对接。 | 2021-3-29~2021-4-15 | 梁晨 | | 8 | ·掌握第三方库的交叉编译，掌握jni调用原生代码技术，掌握react native调用原生代码技术  ·掌握react-native-ffmpeg的使用，学会编码，解码，合并等操作。  ·尝试使用rnnoise进行噪声消除  ·研究回声消除相关的算法，并实现  ·研究是否还有其他需要的音频算法。 | 2021-3-29~2021-4-15 | 乐辰阳 | | 9 | ·复现成功消声算法，并部署到后端。 | 2021-3-29~2021-4-15 | 朱家琛 | | 10 | ·音频打分算法封装API，成功部署到前端。  ·给出音频打分教程。  ·根据课程进度，撰写相应的文档。 | 2021-3-29~2021-4-15 | 鲍辰 | | 11 | ·学习混音，效果器等高级音频组件的实现。  ·调整音频处理代码，封装函数，并和其他人的工作对接。  ·进行性能优化。 | 2021-4-16~2021-5-4 | 乐辰阳 | | 12 | ·用CodeIgniter架构去分层设计好后端架构。  ·修改代码风格以符合项目的编程规范。  ·学习后端优化方法以提高并发量，并找到相对可靠的反应性能的参数。  ·参考测试同学给出的测试结果，相应地优化服务器的负载能力，至少达到vision文档里的预期水平。  ·完成优化过程的笔记或复现教程，以供其他组员参考。 | 2021-4-16~2021-5-4 | 梁晨 | | 13 | ·学会appium自动化测试，并能成功测试交我唱app，并完成测试报告。 | 2021-4-16~2021-5-4 | 朱家琛 | | 14 | ·掌控全组进度。  ·转向后端开发，主要负责后端新技术的应用，如DevOps、Docker、kubernetes等。 | 2021-4-16~2021-5-4 | 鲍辰 | | 15 | ·完成前后端数据互通，用户体验正常流程实现。  ·完善代码架构、界面风格，根据测试结果对前端进行进一步改进。 | 2021-4-16~2021-5-4 | 郑博洋 | | | | |
| 预期成果：  1.用例模型  2.软件需求规约  3.软件架构文档  4.编程规范  5.UML分析与设计模型  6.技术原型的代码  7.ffmpeg模块的api封装，噪音抑制模块的api封装，回声消除模块的api封装，混音器，效果器的雏形，整个K歌流程通过。  8.成功部署伴奏消声算法。给出源代码以及环境需求，给出对应教程。  9.成功部署唱歌实时打分算法。给出源代码以及环境需求，给出对应教程。  9.实现测试，给出测试方法，提供测试报告。  10.实现账户登录用例、实现K歌用例、实现完整界面和美观UI。  11.后端完成所有对接任务，达到预期水平，尽可能优化算法性能。 | | | |
| 主要的风险和应对方案：   1. 第二轮迭代的技术风险和进度风险。按照交我唱vision文档中的要求，在本迭代中需要实现，伴奏消声算法、唱歌打分算法、降噪算法、回声消除算法、混音器及其算法、效果器及其算法。这对于我们的知识面和知识储备是一个不小的压力，所以我们在前后端一个主要负责人不变的前提下，其他三位同学成立算法组，各自负责一个方面的算法进行总攻。我们以开源代码为基石，以复现——对接——部署三步走的战略，流水线化整个算法开发流程。两位前后端的负责人在实现他们本身的任务的同时，也负责了对接和部署环节中的主要部分，这样打通了组员之间的信息壁垒，使得整个开发流程更加顺畅。同时增加了组员之间的沟通渠道，提升了组员的参与感，使得组员们更有效率地使用更加便捷的流程开发更加强大的软件。 2. 第二轮迭代的质量风险。由于本迭代中涉及的已有的和新开发的模块众多，如果一股脑地去开发很容易导致返工的任务量巨大。所以，我们在第一轮迭代中设立的测试组成员将继续负责测试的部署。由于已经学习了一个月的知识，从当前迭代开始研究自动化测试脚本变得可行，降低了学习的门槛。同时，测试和算法开发齐头并进，这样一旦有错误就可以及时发现，降低了在不同环节流通中将BUG放大的可能性。这样能够有效地提升质量。 3. 总体进度风险。目前来说，进度还是在可控范围内，但是如此大体量的开发过程，难免在未来可能遇到一些进度上的问题。所以，我作为组长，在已有的开发过程中，我还将去尝试DevOps、CI/CD等一系列能够提升开发效率的新技术，这作为试验性的技术投资。可能会在第三轮迭代中大展宏图。 | | | |