**项目总结报告**

日期：2021.6.24

|  |  |  |  |
| --- | --- | --- | --- |
| 组号 | 14 | 项目名称 | 天天爱K歌 |
| 编程语言 | Java、Python、C++、C、JS、xml | 开发平台和框架 | Android Studio、Vue.js、Flask、Visual Studio |

|  |  |
| --- | --- |
| **项目工作小结** | |
| 1. 项目需求完成情况   本项目按时完成了立项时定下的所有需求，且完成质量较高。除此之外，还完成了三项额外需求：用粒子特效和多种第三方库对界面进行美化；为打分算法添加了音准、情感、气息、演奏四项额外指标，增加了打分的科学性。  二、架构风格与设计模式  粒子特效系统：抽象工厂，装饰器模式，享元模式，状态模式，单例模式  和弦解释器：解释器模式，命令模式，门面模式  演唱（演奏）打分系统：策略模式，门面模式，适配器模式  UI组件：建造者模式、装饰器模式  控制逻辑：状态模式   1. 技术方案亮点 2. 手机端APP 3. 大量使用多线程，最大限度地保证了在进行高负荷运算时UI的流畅度。 4. 适应性界面组件及适应性渲染逻辑使APP在各种分辨率的手机上都能正常显示。   3）演唱（演奏）打分系统从音准，气息，情感，演奏等多个指标进行演唱（演奏）评价，以科学的评价标准吸引用户。  4）演唱（演奏）打分系统属于APP端计算最密集的模块，APP端全部代码采用C++实现以及端云结合的设计保障系统性能，降低手机耗电。  5）在录音时将录音切分为多个音频文件，并通过多线程不断地输入演唱打分系统，使打分和用户的演唱同步进行，极大地提高了打分速度。  6）lrc文件结合Bitmap实现了歌词同步逐字高亮。  7）自弹自唱的特效实现采用了粒子系统，以界面美观性吸引用户。  8）使用雷达图，以炫酷的方式展现打分系统结果。  2. 管理员系统  1）云端采用神经网络算法实现人声伴奏分离以及乐器音轨分离，为自弹自唱模式提供“DIY伴奏”的功能以吸引用户。  2）通过socket实现前端系统多线程上传的进度提示，实时提示管理员上传进度。   1. 测试情况   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？   1. 单元测试 2. 系统测试   系统测试100%覆盖所有功能性需求，并重点对手机端APP进行了测试，重点测试了界面的响应能力及健壮性。   1. 兼容性测试   手机端APP在以下机型上进行测试，且均能正常安装和运行：Vivo x27，华为Nova4，Honer magic 2，Redmi Note 5，华为FLA-AL20、小米8SE。  管理员系统前、后端在Windows和macOS BigSur系统上均进行了测试，可以正常安装和运行。测试使用浏览器为Chrome Version 91.0.4472.106、Chrome Version 91.0.4472.114、Microsoft Edge Version 91.0.864.54、Safari Version 14.1。 | |
| **项目组成员对项目的贡献度（%）** | |
| 郭志东36% 江雨泽33% 曹沅欣31% | |
| **软件度量** | |
| 软件代码行数（不包括注解行、空行和复用代码）： | 9854 |
| 复用他人代码行数： | 12793 |
| 类的个数： | 103 |

|  |
| --- |
| **经验、教训和建议** |
| 1. 在多人合作开发时，要先明确各自提供的接口以及负责部分的预期效果，才能避免合并时的大量重写工作。 2. 在开发时，要充分考虑到不同平台的性能差距和兼容性。例如，在开发打分系统的过程中，由于没有意识到电脑和手机的性能差距，导致打分系统的第一版实现有极长的打分延迟，必须进行大量重写，导致开发进度紧张。   2. 在测试软件时，要考虑到各种使用情况及边界条件；除了测试功能性需求，还要测试非功能性需求，例如是否有未释放的资源、无法结束的线程等。  3. 开发过程中要严格按照开发进度进行开发，不能因为自己的任务可以在短时间内完成就推迟开发，影响团队的开发进度。 |

项目组各成员签字：