

**《软件工程》项目报告**

**题目： 基于WEB的音乐软件**

**课程名称： 软件工程**

**专业班级： 计算机2011**

**组 名：**

**同组成员： 学号： U202011067**

**姓名： 邱佳荣**

**学号：**

**姓名：**

**学号：**

**姓名：**

**学号：**

**姓名：**

**指导教师： 冯琪**

**报告日期： 2022.12.15**

**计算机科学与技术学院**

# 任 务 书

**一 总体要求**

1. 综合运用软件工程的思想，协同完成一个软件项目的开发，掌软件工程相关的技术和方法；

2. 组成小组进行选题，通过调研完成项目的需求分析，并详细说明小组成员的分工、项目的时间管理等方面。

3. 根据需求分析进行总体设计、详细设计、编码与测试等。

**二 基本内容**

根据给出的题目任选一题，自行组队，设计与开发中软件过程必须包括：

**1. 问题概述、需求分析：**正确使用相关工具和方法说明所开发软件的问题定义和需求分析，比如NABCD模型，Microsoft Visio，StarUML等工具 (20%)；

**2. 原型系统设计、概要设计、详细设计**：主要说明所开发软件的架构、数据结构及主要算法设计，比如墨刀等工具（35%）；

**3. 编码与测试**：编码规范，运用码云等平台进行版本管理，设计测试计划和测试用例（30%）；

**4．功能创新**：与众不同、特别吸引用户的创新（10%）；

**5. 用户反馈**：包括用户的使用记录，照片，视频等（5%）。

目 录

[任 务 书 I](#_Toc121938798)

[1 问题定义 4](#_Toc121938799)

[1.1项目背景与意义 4](#_Toc121938800)

[1.2 项目基本目标 5](#_Toc121938801)

[1.3 可行性分析 5](#_Toc121938802)

[1.4人员管理和项目进度管理 6](#_Toc121938803)

[2 需求分析 8](#_Toc121938804)

[2.1 E-R图、数据流图 8](#_Toc121938805)

[2.2 用例图 9](#_Toc121938806)

[2.3 原型系统设计 10](#_Toc121938807)

[3 概要设计和详细设计 13](#_Toc121938808)

[3.1 系统结构 13](#_Toc121938809)

[3.1.1 功能说明 13](#_Toc121938810)

[3.1.2 代码结构、接口设计 13](#_Toc121938811)

[3.2 类图和UML时序图 15](#_Toc121938812)

[3.3关键数据结构定义（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗） 17](#_Toc121938813)

[3.4 关键算法设计（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗） 17](#_Toc121938814)

[3.5 数据管理说明（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗） 17](#_Toc121938815)

[4 实现与测试 18](#_Toc121938816)

[4.1实现环境与代码管理（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗） 18](#_Toc121938817)

[4.2 关键函数说明（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗） 18](#_Toc121938818)

[4.3 测试计划和测试用例（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗） 19](#_Toc121938819)

[4.4 结果分析（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗） 19](#_Toc121938820)

[5 总结 20](#_Toc121938821)

[5.1 用户反馈 20](#_Toc121938822)

[5.1 全文总结（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗） 21](#_Toc121938823)

[6 体会(黑体小2号加粗居中) 22](#_Toc121938824)

[附录 (黑体小2号加粗居中) 23](#_Toc121938825)

# 1 问题定义

# 1.1项目背景与意义

**1.1.1NABCD模型**

以下用NABCD模型进行说明。

N(Need):

实现一个可以随手记录音乐的音乐软件。

在此基础上，可以实现一个自定义谱面的音乐游戏。

A(Approach):

用户通过哼鸣，记录一段音乐。

将此段音乐转为midi，然后用来继续混音。

也可以将此段音乐转为谱面。

将此段音乐用于打谱，然后实现一个自定义谱面的音乐游戏。

B(Benefit)

更加低的非专业用户的学习成本。

更加轻量级的编辑。

灵感随有随记的体验。

更加低的软、硬件设备需求。

更加个性化、自由度更加高的游戏体验。

C(Competitors)

改善了传统音乐软件的学习成本较高的问题。

改善了传统音乐软件不便携、麻烦的问题。

改善了传统音游个性化低、自由度低的问题。

D(Delivery)

可以进一步完善为一个音乐游戏，推荐给玩音游的朋友们。

也可以进一步完善成一个微信小程序或者手机app，上架给音乐制作人或者音乐爱好者。

**1.1.2项目分析**

1. 现有系统的不足

* 学习成本高
* 设备要求高，不能够随时记录灵感
* 音乐游戏比较固定，不能够打自己喜欢的曲子

2. 改进方式

* 点击即可录音记录
* 自动转化为midi和曲谱，不用用户操作
* 拖动即可混音
* 一键导出为pdf或者播放，一键导入音乐游戏打歌

3. 需求

* 注册
* 登陆
* 用户数据管理
* 录音记录
* .wav转.mid
* 音乐导出曲谱
* 音乐导入音乐游戏

# 1.2 项目基本目标

经过小组讨论我们的基本目标是在网页上实现一个音乐处理软件。要完成的基本功能功能包括登录、注册、录音、音乐处理、输出并保存mid、输出并保存wav、输出并保存谱面。如果情况允许，进阶功能有制作音乐游戏歌曲。

# 1.3 可行性分析

首先我们想到的是音乐识别。由于普通的音乐识别已经趋于成熟了，发现古典音乐人经常被找不到想要的曲目所困扰，我们决定进行古典音乐识别。但是古典音乐识别出于古典音乐录音种类繁多，所听的录音也许与数据库里的乐纹数据不是一个版本的录音，不是一个可行方案。哼鸣识别是我们想做的一个方向，但是在网上没有找到开源的数据集，这使得计划流产。

在进行了识别之后，我们或许可以对音乐进行比较。识别到该音乐的相似的曲子的时候，我们可以给出他们的相似程度排名以及高度相似的部分。但因为音乐识别不能够实现，所以音乐比较也被迫终止。

其次一个思路是音乐作曲。普通的作曲软件需要较高的学习成本、上手成本和专业的i/o设备。哼鸣声作曲能够通过麦克风采集到我们的哼鸣声然后直接生成一段音乐。

作曲完之后我们可以进行混音，基于此我们找到了基于unity、基于qt的插件，如果要实现作曲混音的步骤是可行的。

和音乐相关的软件还有音乐游戏，由于普通的音乐游戏已经非常的完善，而且我们也没有自信去超越他们，基于我们基本功能能够实现音乐作曲，我们决定做一款能够自作曲的音乐游戏，提高用户的可玩性。

具体的思路见图表 1 头脑风暴与可行性。

日程表

描述已自动生成

图表 1 头脑风暴与可行性

# 1.4人员管理和项目进度管理

文档编写：邱佳荣

包含子任务：进度计划、系统设计、需求规格、实现与测试、用户反馈

前端设计：王逸

包含子任务：登录功能、注册功能、页面框架设计

算法设计：邱佳荣、王逸、申亚宁

包含子任务：录音、转换-降噪算法、转换-生成音符算法、转换-音符切割算法、生成音乐

集成测试：申亚宁

包含子任务：服务器部署、域名解析、服务器维护、内网管理端口

后续完善：邱佳荣、王逸、申亚宁

包含子任务：打谱功能、接入数据库、导入到音乐游戏、多分类

发布：王逸、申亚宁、邱佳荣

具体的人员安排和进度管理如下图表 2 甘特图所示。

在线链接：<https://gantt.mindsup.cn/share/9pnslSqIh>

图形用户界面, 应用程序, 表格, Excel

描述已自动生成

图表 2 甘特图

# 2 需求分析

# 2.1 E-R图、数据流图

系统的ER图如图图表 3 E-R图所示。系统的实体包括了普通用户、.wav音乐和.mid音乐、谱子。其中的联系如下：普通用户录入 .wav音乐，.wab音乐转化为曲谱，不仅如此，用户还能查询谱子和.mid音乐。

图示

描述已自动生成

图表 3 E-R图

系统的数据流图如图表 4 数据流图所示。用户登录动作酱登录信息通过识别查找用户信息库反馈回来，进入登录用户操作。登录后用户进行录音录入.wav信息库，此后自动从该信息库中获取信息并且转化为.pdf和.mid信息库，用户可以从该信息库中查询获取信息，然后反馈给登录的用户。

图示

描述已自动生成

图表 4 数据流图

# 2.2 用例图

系统的UML图如图表 5 UML用例图所示。系统中仅有的角色就是用户，用户登录系统，验证账号密码或者注册，然后进行录音。录音后可以选择各个音乐处理的过程，包括保存mid，保存wav，保存谱面，制作歌曲。

图示

描述已自动生成

图表 5 UML用例图

# 2.3 原型系统设计

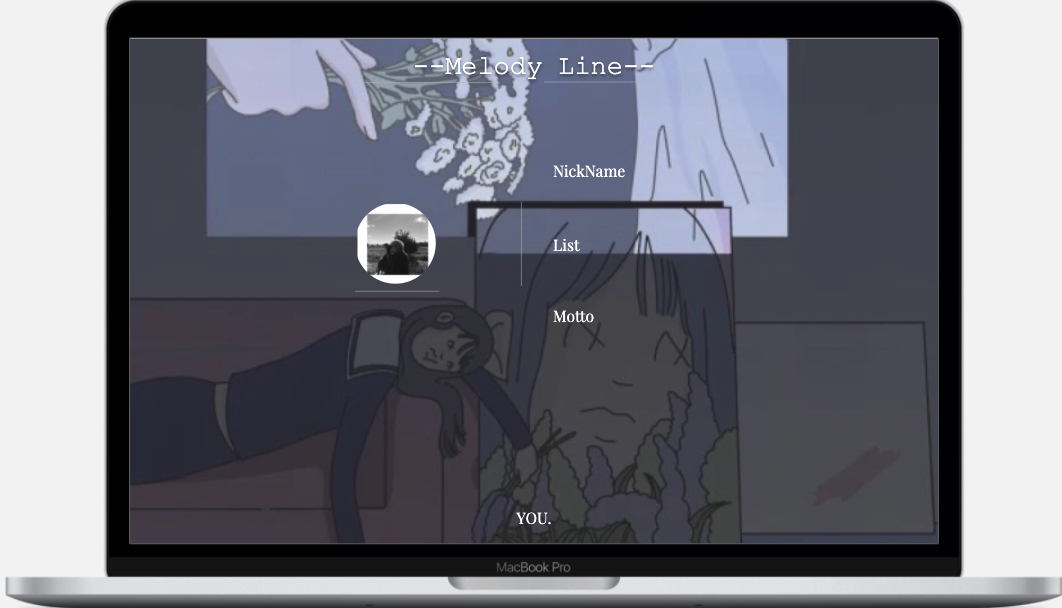
1.用户登入

点击圆框，未登录显示需要用户的用户名以及密码，以及需要一个登陆按键来支持页面的跳转，以及向后端发送post请求。其次还要有创建账号的按键提供用户进行选择，用户点击创建账号应该可以跳转到注册账号的信息，给用户提供注册功能。如果登录了，点击圆框会进入下一个界面。



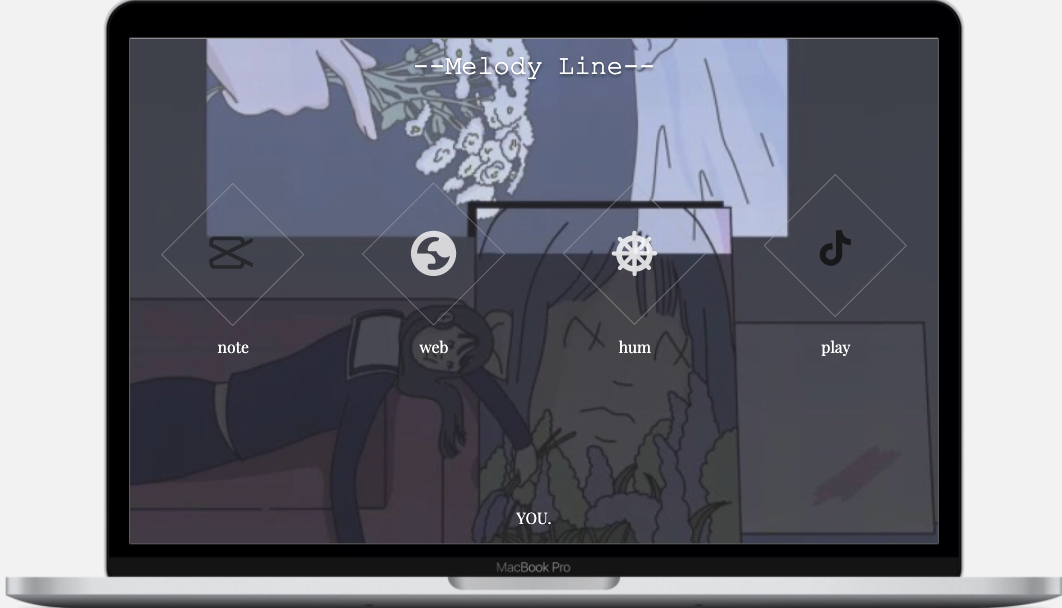
2.个人页面

个人界面展示我们用户名，头像，以及个人生成的歌曲、谱子。



3. 使用说明

点击web查看网页端用法，点击下载可以下载相应的客户端。



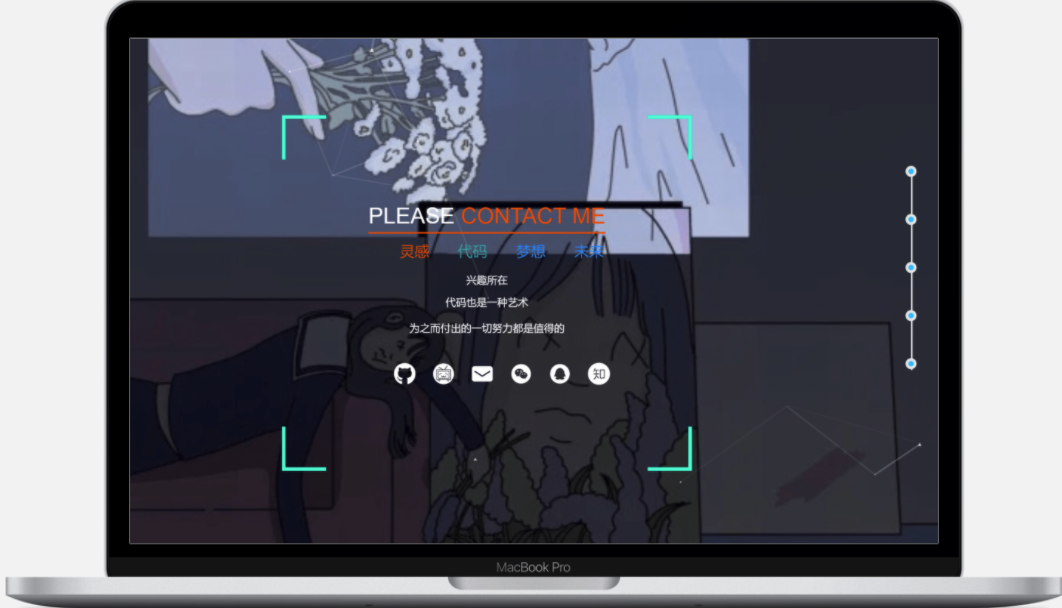
4. 音乐处理

此处有四个模块。点击rec开始录音，点击pause暂停，然后点击compose转化，点击play播放。



5.页尾

拖动到网页最后，显示页尾，有源代码、开发者等各种信息。



# 3 概要设计和详细设计

# 3.1 系统结构

# 3.1.1 功能说明

系统架构图如图表 6 架构图所示，用户从web端登录或者注册，进行记录，数据可以录入.wav信息库，用户进行转写，数据可以录入.pdf信息库，用户进行转写，数据可以录入.mid信息库。

**图示

描述已自动生成**

图表 6 架构图

# 3.1.2 代码结构、接口设计

一 文件结构：代码

1. /sound\_recorder

录音，生成wav文件保存在/record\_wav

2. /wav\_note（很多）

提取音符，生成乐谱pdf保存在/note\_pdf

3. /compose

音符转乐曲，生成mid文件保存在compose\_mid

4. handle.py

rec = record.Recorder()录音，全局变量

transcriber = music\_transcriber.MusicTranscriber(file\_name)提取音符

composer = music.Music()音符转乐器音，全局变量

二 文件结构：文件

1. /record\_wav

录音保存的音频文件

2. /note\_pdf

乐谱pdf

3. /compose\_mid

音符转乐曲的结果（项目最终结果）

三 后端接口

1、设置乐曲名

handle.set\_music\_name("乐曲名")

2、录音

handle.start\_record()调用这个函数开始录音

handle.stop\_record()录音结束后调用这个函数，生成音频 乐曲名.wav，保存在record\_wav文件夹

3、提取音符

melody = handle.get\_notes()，生成五线谱 乐曲名.pdf 保存在note\_pdf文件夹

4、音符转乐器音（打括号的是非必要函数）

（（1）可以设置乐器）

handle.choose\_program(id)

（2）音符生成乐器音

handle.create\_melody(melody)，用第3步得到的melody生成乐器音 乐曲名.mid 保存在compose\_mid文件夹

（（3）播放）

handle.play\_music()

（（4）mid转wav，需要java环境）

handle.mid2wav()

# 3.2 类图和UML时序图

UML时序图如图表 7 UML时序图所示。用户记录声音片段，进入Recorder类。保存后读取.wav文件进入Transcriber类进行转写；此后获得.wav文件，进入Spilter类分离；在分离前，还要进入Reducer类降噪，降噪完以后在spliter进行音乐分解，分解完返回，进入Transcriber中，使用Plotter进行打谱。此时用户就能够得到.pdf文件。进入Comporser中，可以播放或者保存为midi。

图示

描述已自动生成

图表 7 UML时序图

UML类图如图表 8 UML类图所示。用户类包含了乐曲名、乐曲路径、用户名、密码，可以进行记录、转化、作曲。用户可以记录乐曲，Recoder类中包括了乐曲路径，可以进行录音和保存。用户也可以进行转化操作，Transcriber类中包括了音乐、输出路径，可以进行打点（Plotter）和降噪（Enhancer）并且转化操作。用户还可以进行作曲，Composer类中包括了音乐、乐曲名、乐曲路径、乐器、节拍数，可以进行设置乐器、播放、保存为.mid文件，转化为.wav文件的操作。

图示

描述已自动生成

图表 8 UML类图

# 3.3关键数据结构定义（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

这部分要写的：（1）首先描述系统中要处理那些数据，每种类型的数据包括哪些数据项，每个数据项的数据类型；（2) 描述这多种数据在系统中如何关联，可通过图直观的说明这多种数据间的关联。

# 3.4 关键算法设计（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

这部分主要描述系统中的模块实现的流程，可采用文字配合流程图的方式表示关键算法的思想及流程。

# 3.5 数据管理说明（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

说明整个系统所涉及到的数据用何种方式进行存储和访问。

# 4 实现与测试

（黑体小2加粗居中,字母、阿拉伯数字为Time New Roman小2号加粗）

# 4.1实现环境与代码管理（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

这部分可首先叙述所设计软件实现的软硬件环境，代码版本管理的签入记录需截图进行说明。码云平台签入记录截图如图4-1所示：



图4-1 码云平台代码签入示例

# 4.2 关键函数说明（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

程序代码部分在这里不需要给出来，只需要叙述清楚在系统中包括哪些函数，各函数的说明，如何利用这些函数实现系统各模块的功能，以及函数间的调用关系（可用图表示出来）。程序详见附录。

# 4.3 测试计划和测试用例（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

首先叙述一下常用的软件测试方法，在选择几个主要的功能模块（自行掌握数量，关键要体现你的水平的一些模块）描述测试过程，（1）先明确模块的功能、设计目标等。（2）分析、叙述如何选取测试数据，要求有完整的测试计划和测试用例，说明测试运行结果（这时可用截图）。

# 4.4 结果分析（黑体4号加粗,字母、阿拉伯数字为Time New Roman4号加粗）

根据测试分析运行结果、确认软件是否满足需求。

# 5 总结

# 5.1 用户反馈

用户徐锦慧（音乐初学者）：我在听说这个项目的时候就对这个软件十分的感兴趣，因为我自己也在弹吉他，很期待这样一个能够记录谱面的软件。对于我这样的吉他入门者来说，此软件的功能是十分的健全的，能够识别我弹的曲子；而且我的耳朵也不够好，谱面识别能够帮助我进行视唱练耳的学习。

但是在使用的过程中我发现扫弦是很难被识别出来的，所以觉得应该是算法还不够完善。给开发者反馈之后，收到回复得知多音的识别的算法还是十分的复杂，需要更多的时间完善。总的来说，此款软件能够满足我平时的使用需求，弥补了在这方面的缺口，功能方面对我来说基本够用，期待进一步的完善。

用户邓智维（普通的音乐爱好者）：根据使用说明下载了这个软件之后我进行了初步的使用，对于爱唱歌的我来说十分的好用。

但是对于平常使用电脑也仅仅是进行文档编辑的我，下载、安装的过程还是冗杂了一些，中间还遇到了不少的环境问题，虽然求助了开发者得到了解决，还是希望能够部署在移动端上，毕竟现在手机软件的使用更加广泛一些，而且也更适合抬手即用。

用户刘嘉骏（音乐老师）：作为一名音乐老师，我受学生所托使用了这一款软件。学生亲手帮我安装好的，所以没有遇到什么麻烦。

作为一名小提琴演奏者，软件对于识别单音音符还是十分好用的，缺憾是假设音乐时间过长，则需要等待很长的时间去获得谱子，一般人很难有耐心。但是对于一首曲子，一般时常都在10分钟之后，分段录音也不现实。还有就是因为小提琴音域跨度很大，在极高音的部分和泛音的部分该软件很容易识别低八度，希望开发者能够改善。此外希望能够对谱子进行编辑。

希望我的学生能够完善这些功能，如果真能解决，作为音乐老师我觉得还是对我们很有用处的。

# 5.1 全文总结

对我们的的工作做个总结，主要工作如下：

（1）我们详细地了解了软件工程的开发的过程，并且依据这个过程逐步实现了我们的软件。

（2）我们进行了十分紧密的团队合作，了解了分工计划的重要性。

（3）我们撰写了详细的文档，详细地记录了开发过程，也让代码更加容易读懂。

（4）我们进行了API的对接，进行了代码的规范编写以及注释的规范书写。

（5）我们进行了算法的详细研究，探讨出了一套可行的算法。

# 6 体会

邱佳荣，分工：撰写文档，设计编写后端算法。

体会：在本次软件工程的实验中我理解了文档撰写的重要性。文档撰写在一开始能够帮我们明确需求，使得我们能够有的放矢，还能够探讨出可行的方案，不至于时间精力都打了水漂。在中期我们能够很好的规划时间分配，让开发能够更加高效的进行。在后期能够帮助后加入的开发者快速读懂代码，上手维护。

# 附录 (黑体小2号加粗居中)

×××××××××××××××××××××××××××

( 宋体小4号)

……

……

……