同濟大學

TONGJI UNIVERSITY

《WEB 技术》

实验报告（大作业）

实验名称	在线音乐分享平台
小组成员	高曾谊（1951705）
	贾仁军（1951566）
学院（系）	电子与信息工程学院
专　　业	计算机科学与技术
任课教师	郭玉臣
日　　期	2020 年 7 月 9 日

目录

装

订

线

项目背景

我们组一开始打算做一个水利监测网站，实现覆盖泰州市全境的水利监测功能。不过，实际项目过程中，我们花费了大量的时间在 WEBGIS 的学习上(超图等)，而且依然没有得到一个很好的结果。

考虑到这是一门 web 开发课，我们不应该花费如此多的时间在学习这些软件上。“实迷途其未远”，我们决定回到原点，选择从零开始做一个简单的在线音乐分享平台，并以此为基础不断丰富完善其功能，把更多的关注点放在 web 相关技术的学习与应用上。

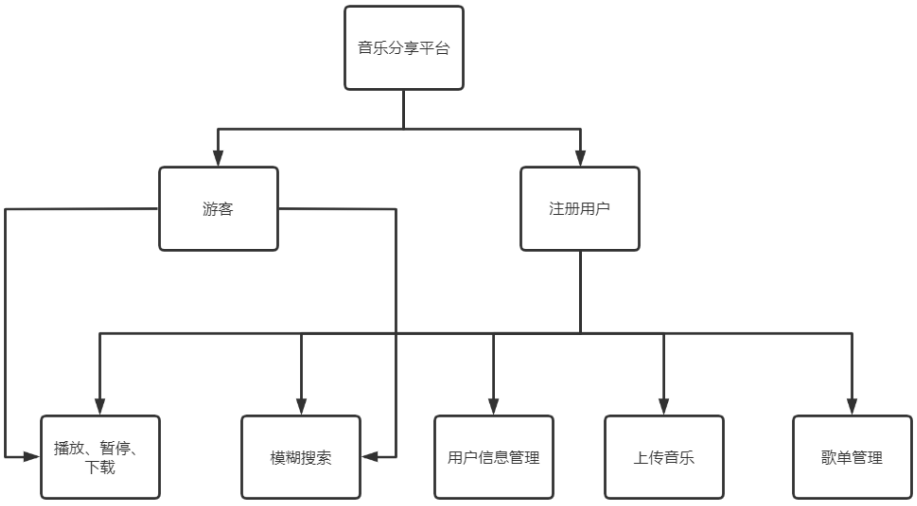
需求分析

我们观察到，当前各大音乐平台如 QQ 音乐、网易云音乐、虾米音乐等获得的版权不一致，用户要听某些歌曲时，需要在这些平台上反复切换。于是，我们希望做出一个供小众使用的在线解析网站(不涉及版权问题的自娱自乐)，以整合各大音乐平台资源，给用户提供方便。

整体设计

这一部分介绍网站的整体功能，从客户端的角度进行网页展示。

整体功能图



装 订 线

主页

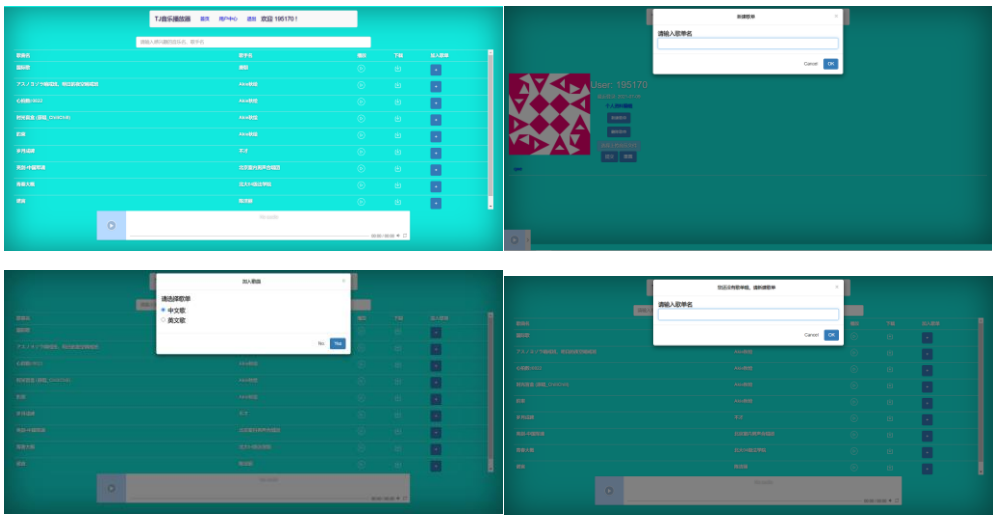
打开主页。此时您已经可以进行音乐播放和下载，并通过歌曲名、歌手名或专辑名进行模糊查询。（游客状态）

注册/登录页面

登录界面中勾选“记住我”即可在下次访问该网页时免登陆。

歌单管理

登录完成后，主页界面音乐列表中各音乐新增加入歌单按钮，可以加入歌曲到歌单，也可以在用户中心页面将音乐从对应的歌单中移除。你也可以去用户中心管理歌单（新建歌单、删除歌单、查看歌单）



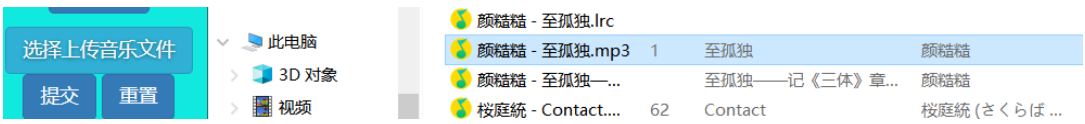
若在首页点击加入歌单时当前用户没有创建任何歌单，则会提示你新建歌单，并将该音乐加入到你新建的歌单中。

个人资料编辑页面

我们通过 python 爬虫技术访问网站 https://gravatar.zeruns.tech/avatar?d=identicon 随机获取分形图作为用户初始头像，您也可以在个人信息编辑页面自己上传头像，编辑个性签名。

分享音乐

注册登录后可以上传分享音乐，支持 mp3/flac 等常用音乐文件。



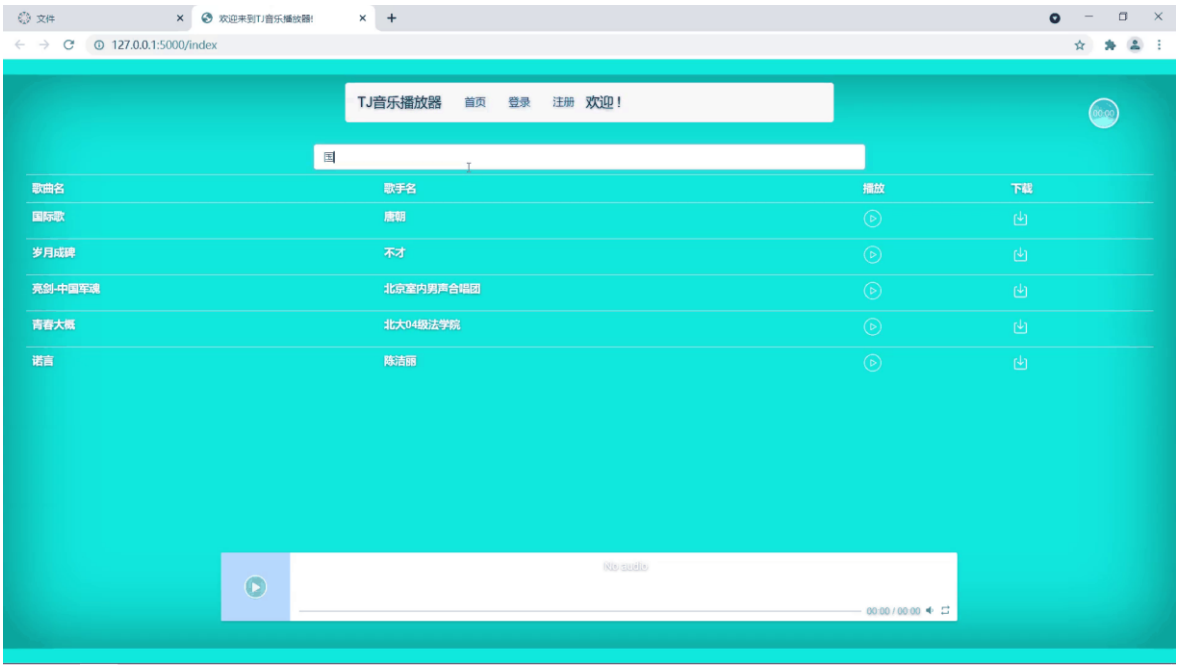
上传完成后，后端自动解析其歌曲名、歌手名、专辑名及歌词封面，同时将该音乐共享显示到首页。在这里上传文件【致孤独.mp3】，然后在主页可以看到该音乐，如图所示：

装 订 线

最终效果展示

见结果展示.mp4

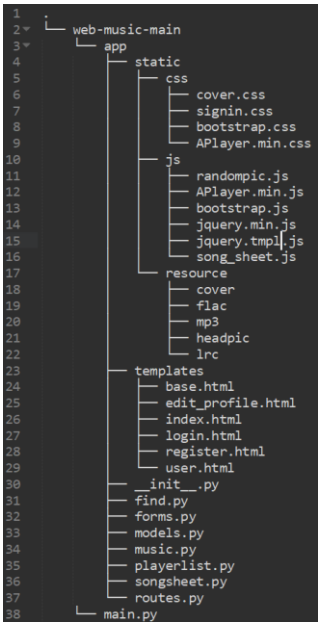
同时在这里也附上一个 GIF:



装 订 线

详细设计

文件目录



环境

后端

- Python 3.7.0
- Werkzeug 1.0.1
- Flask 1.1.2
- Flask_login 0.5.0
- Mysql　8.0.24
- Pymysql 1.0.2
- Peewee　3.14.4
- Mutagen 1.45.1

前端

- BootStrap 3.4.1
- Bootbox 5.5.2
- Aplayer.js 1.10.1
- Jquery 3.5.1

音频文件解析

使用 python 封装好的 mutagen 模块解析 mp3/flac 文件，获得歌曲名、歌手名、专辑名、专辑封面，将这些歌曲基本信息存入数据库后，重命名音频文件，依据“歌手名-歌曲名.后缀名”的格式命名，以方便访问歌曲资源。（见 music.py）

下面以 flac 文件为例附上相关代码:

```
def getflacinfo():
    teplist = os.listdir(flacpath)
    for i in teplist:
        path = flacpath + i
        afile = File(path)
        author = afile.tags["Artist"][0]  # 作者
        author = formatname(author)
        title = afile.tags["Title"][0]  # 标题
        title = formatname(title)
        album = afile.tags["Album"][0]  # 专辑
        #print(album)  # 专辑
        try:
            models.Music.get(models.Music.singer_name == author,
                             models.Music.music_name == title,
                             models.Music.album_name == album)
        except:  #未找到
            models.Music(singer_name=author,
                         music_name=title,
                         music_type='flac',
                         album_name=album).save()
            try:
                artwork = afile.pictures[0].data
                with open(coverpath + author + ' - ' + title + '.jpg',
                          'wb') as img:
                    img.write(artwork)  # write artwork to n
            except:
                cover.save(coverpath + author + ' - ' + title + '.jpg')
                #可以自己添加一张默认封面

            srcFile = path
            dstFile = flacpath + author + ' - ' + title + '.flac'
            try:
                rename(srcFile, dstFile)  #歌曲重命名
            except:  #去重
                os.remove(dstFile)
                rename(srcFile, dstFile)  #歌曲重命名

            try:
                rename(lrcpath + i[:-5] + ".lrc",
                       lrcpath + author + ' - ' + title + ".lrc")  #歌词重命名
            except:
                pass
```

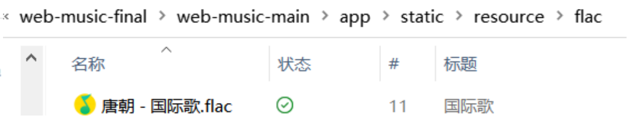
数据管理

查询相关资料可知，python 通过 pymysql 和各数据库交互；而 peewee 进一步封装了 pymysql，统一了数据库的操作与调用方式，更加方便了我们的开发。

故本项目数据选择 mysql 数据库，通过 peewee 交互。

所有歌曲文件存放在固定的文件夹中，以“歌手名–歌曲名.后缀名”的格式命名，相当于在数据库只存放地址，本地服务器存放文件。用户上传音频文件后，将对应文件放置相应文件夹里。

以 mp3 及 flac 文件为例，文件结构如图所示：



数据库设计

数据库中共三张表：

- User
- Music
- SongSheet

其具体内容如下：

User	存放用户信息
Music	存放音乐信息
SongSheet	存放用户歌单信息

User					
id	username	email	about_me	last_seen	password_hash
主键	用户名	邮箱	个性签名	上次登陆时间	密码散列值

Music				
id	music_name	siner_name	music_type	album_name
主键	歌名	歌手名	歌曲后缀名	专辑名

SongSheet			
id	userid	sheet_name	musiclist
主键	该歌单所属用户id	歌单名	歌单中包含的歌曲列表

其中，在 User 这张表中我们采取如下额外措施：

- 数据库中不存放用户密码，而是存放用户密码散列值。如图所示：

```
mysql> select * from user;
+----+----------+-------------------+----------+------------+-----------------------------------------------------
| id | username | email             | about_me | last_seen  | password_hash
+----+----------+-------------------+----------+------------+-----------------------------------------------------
|  1 | 1951705  | 1262454489@qq.com | 121      | 2021-07-02 | pbkdf2:sha256:150000$MiFw63vn$c687d5d090e6dfdd36be
+----+----------+-------------------+----------+------------+-----------------------------------------------------
```

- 以 python 封装好的 werkzeug.security 包中的 generate_password_hash 函数做为密码散列函数，将原密码的散列值存入数据库，提高安全性，保护用户隐私。
- check_password_hash 函数实现密码转换及校对，用于用户登录验证。

相关代码如下（使用 peewee 封装 pymysql 的格式，详见 models.py）：

```
class User(UserMixin, BaseModel):
    #id = pw.IntegerField(primary_key=True)  # 主键
    username = pw.CharField(max_length=64, index=True, unique=True)  # 唯一的用户名
    email = pw.CharField(max_length=120, index=True, unique=True)  # 唯一的邮箱
    about_me = pw.CharField(max_length=256)
    last_seen = pw.DateField(default=datetime.utcnow())
    password_hash = pw.CharField(max_length=128)
    def set_password(self, password):
        self.password_hash = str(generate_password_hash(password))
    def check_password(self, password):
        return check_password_hash(self.password_hash, password)
    def __repr__(self):
        return '<用户名:{}>'.format(self.username)
    def avatar(self, size):  # 根据邮箱自动获取分形图的 API
        digest = md5(self.email.lower().encode('utf-8')).hexdigest()
        return 'https://gravatar.zeruns.tech/avatar/{}?d=identicon&s={}'.format(
            digest, size)
class Music(BaseModel):  # 标准格式 继承自 BaseModel，直接关联 db，并且也继承了 Model Model 有提供增删查改的函数
    #id = pw.IntegerField(primary_key=True)
    music_name = pw.CharField(verbose_name='音乐名',
                              max_length=128,
                              null=False,
                              index=True)
    singer_name = pw.CharField(verbose_name='歌手名',
                               max_length=128,
                               null=False,
                               default='key')
    music_type = pw.CharField(verbose_name='歌曲类型',
                              max_length=16,
                              null=False,
                              default='mp3')
    album_name = pw.CharField(verbose_name='专辑名',
                              max_length=128,
                              null=False,
                              default='NO FOUND')
    def __repr__(self):
        return '<Music {}>'.format(self.music_name + ' - ' + self.singer_name)

class SongSheet(BaseModel):
    #id = pw.IntegerField(primary_key=True)
    userid = pw.IntegerField(index=True)  # 唯一的用户名,对应
    sheet_name = pw.CharField(verbose_name='歌单名',
                              max_length=32,
                              null=False,
                              index=True)
    musiclist = pw.CharField(verbose_name='歌曲列表', max_length=1024, default='')
    #歌曲 id 之间以'-'分割
    def __repr__(self):
        return '<SongSheet {}{}>'.format(self.user_id, self.sheet_name)
```


装 订 线

Peewee——数据库增删改查

Peewee 是一个简单小巧的 Python ORM，借助 peewee 库对 pymysql 的进一步封装，我们得以更加简单、整洁地完成对数据库增删改查，具体方法如下：

1、 增

```
user = User(username=form.username.data, email=form.email.data).save()
```

2、删

```
SongSheet.delete().where(SongSheet.userid == userid).execute()
```

3、改

```
current_user.username = form.username.data
current_user.about_me = form.about_me.data
current_user.save()
```

4、查

单条查询

```
tepuser = User.get(User.username == current_user.username)
```

多条查询

```
posts = SongSheet.select().where(SongSheet.userid == current_user.id)
```

以上参考官方文档 http://docs.peewee-orm.com/en/latest/

模糊查询

使用 python re 库进行正则匹配，将输入的字符串转换成如下格式：

$$E.G.\ abc \rightarrow\ a.*?b.*?c$$

将数据库中 Music 这张表里的歌手名、歌曲名、专辑名全部提取到一个列表中，查找上述列表中是否存在与对应正则表达式匹配的串，并返回给前端。

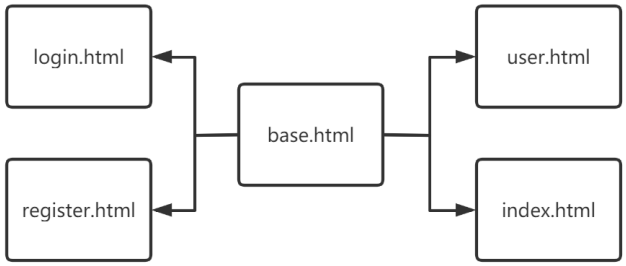
以搜索专辑名为例，搜索结果如图：（以下文件的专辑名均为 CLANNAD Original Soundtrack）

下面附上相关代码（详见 find.py）：

```
def fuzzyfinder(user_input, collection):
    suggestions = []
    pattern = '.*?'.join(user_input)  # e.g. 'abc' -> 'a.*?b.*?c'
    for item in collection:
        match = re.search(pattern, item[1], re.I)  # 歌曲名
        if match:
            suggestions.append(item)
        match = re.search(pattern, item[2], re.I)  # 歌手名
        if match:
            suggestions.append(item)
        match = re.search(pattern, item[4], re.I)  #专辑名
        if match:
            suggestions.append(item)
    teplis = list(set(suggestions))  #去重
    teplis.sort(key=lambda x: x[0])  #排序
    return teplis
```

前端框架设计

设计了 base.html 并以此为模板拓展其他 html 保持风格统一。




```
<div>
    {% block content %} {% endblock %}
</div>
```

base.html

```
{% extends 'base.html' %} {% block content %}
    ……
{% endblock %}
```

login.html/register.html/user.html/index.html

使用模板类的好处:

1、能够使用模板类的基础上对后续页面进行修改和丰富，保持界面风格统一。

2、借助模板类能够大大减少 html 的代码量。

3、避免在多个界面设计时出现 bug 而无法精确定位，减少 debug 时间。

前后端通讯

- 使用 html 设计简洁且可扩展的前端页面
- 当 html 渲染时，利用 JQuery 向后端呼起请求
- 后端收到请求后，将 JSON 数据传输给前端
- 前端根据后端传来的数据数量，使用 Flask 框架动态扩展表格长度并显示
- 代码框架如图所示（详见 songsheet.py 及 song_sheet.js）：

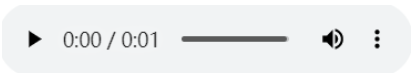
```
$.ajax({
    type: "get",
    url: "/getsheetname",
    data: {},
    async: false,
    success: function (data)
```

```
@app.route('/getsheetname/', methods=['GET', 'POST'])
def GetSheetName():
    "查询一个用户所有的表的名字"
    if request.method == 'POST':
        userid = current_user.id
        lis = []
        p = SongSheet.select().where(SongSheet.userid == userid)
        for i in p:
            lis.append({'text': i.sheet_name, "value": i.sheet_name})
        return make_response(jsonify(lis))
```

其他细节设计

歌曲播放组件

Aplayer.js 替换原生的 audio 组件，拥有更丰富的表现力和更多的交互组件，同时兼容原生 audio 的大部分接口。将列表中的播放键与 Aplayer 的播放键关联，优化使用体验。



原生 audio 组件

Aplayer.js 组件

播放逻辑

1、 点击歌曲旁边的播放按钮时同步开启 Aplayer.js 的播放事件

使用 DOM 和 js 结合的方式对 html 组件进行交互控制

```
<input
id="{{music.music_id}}"
name="play"
type="image"
src="../static/css/play.svg"
height="25"
width="25"
value="0"
onclick="player({{music.music_id}})"
/>
```

2、 点击下方播放栏的状态按钮时，列表中播放状态实时更新

为 Aplayer.js 组件添加点击事件：

在点击播放时如果正在播放，则转入暂停状态

在点击播放时如果正在播放，则销毁原有组件，重新实例化 Aplayer 组件，或者向播放列表中加入新的歌曲，为了保持显示风格的完整性，我们选择了前一种实现方式。

```
function player(id)
    {
      var M={{Musics|safe}};
      var t=document.getElementById(id);
      if (t.value==1){
          var a=document.getElementById("aplayer");
          a.value=id;
          t.value=0;
          t.src="../static/css/play.svg"
          ap.pause();
        }
      else{
        var s=document.getElementsByName("play");
        for (var i=0;i<s.length;i++)
        {
          s[i].value=0;
          s[i].src="../static/css/play.svg";
        }
        t.value=1;
        var a=document.getElementById("aplayer");
        a.value=id;
        TEP=getrealindex(M,id);
        ap.destroy();
        ap=new APlayer({
          container: document.getElementById('aplayer'),
          lrcType: 3,
          audio: {
              name: TEP.music_name,
              artist: TEP.singer_name,
              url: "../static/resource/"+TEP.music_type+"/"+TEP.singer_name+" - "+TEP.music_n
ame+"."+TEP.music_type,
              cover: "../static/resource/cover/"+TEP.singer_name+" - "+TEP.music_name+".jpg",
              lrc: "../static/resource/lrc/"+TEP.singer_name+" - "+TEP.music_name+".lrc"
          }
      });
        ap.on('pause',function(){
        var s = document.getElementsByName("play");
          for (var i = 0; i < s.length; i++) {
            s[i].value = 0;
            s[i].src = "../static/css/play.svg";
          }
        });
        ap.on('play',function(){
```

```
        var a = document.getElementById("aplayer");
        var t = document.getElementById(a.value + "");
        t.src = "../static/css/pause.svg";
        });
        ap.play();
        t.src="../static/css/pause.svg";
    }
    }
```

3、 可将多首歌曲加入播放列表，使用 Aplayer.js 的歌曲轮播效果

Aplayer 提供了播放列表组件，可以将歌曲加入播放列表并逐一控制

4、 使用下方音量控制键、字幕显示键实现对显示样式的控制的调整

Aplayer 组件自带的样式显示模块，有着远超原生 audio 组件的交互性

遇到过的问题及解决方法

文件重命名

解析音频文件并将其重命名时 python 报错，原因如下：

文件名不能包含下列任何字符:
\ / : * ? " < > |

解决方案：若歌手名/歌曲名中含有导致文件命名非法的字符，将他们全部替换为下换线字符'_':

```
def formatname(string):
    list = ['\\', '/', ':', '*', '?', '"', '<', '>', '|']
    for i in list:
        string = string.replace(i, '_')
    return string
```

Html 调用 JavaScript 函数时传参错误

原代码如下：

```
onclick="delete_song({{posts[i][0]}},{{music.music_id}})"
```

原因：posts[i][0]为字符串，假设 posts[i][0]为'123abc'，music.music_id 为 1，若使用原来的代码传参，相当于在 html 上写

```
onclick="delete_song(123abc,1)"
```

显然无法正确调用 delete_song 函数

故需要在 posts[i][0]外面加上引号。

改进后的代码如下：

```
onclick="delete_song("{{posts[i][0]}} ",{{music.music_id}})"
```

使用 js 函数前端通信时同步出错

这一个问题并没有体现在我们最后提交的代码中，而是我在测试 Aplayer.js 的组件应用时所发现的。

最初我设想的是在前端应用针对 IDv3 协议的 api 组件对 mp3 文件进行解析从而得到相关的音乐信息，最终实现在前端的 Aplayer 样式显示。但在测试过程中发现有关音乐信息不能在 js 的函数之间有效传递。各个函数之间的同名变量并不能很好地同步起来。

最后我们选择了在后端对 mp3 文件的信息进行批量提取，简化了这一操作。

在完成之后我们发现可以在 Aplayer 的回调函数处使用异步返回的方式对数据进行传递和载入，但 js 处理 IDv3 的 api 使用起来太过繁琐，逻辑和代码都需要进行大量修改，因此保持原有方案不变。

测试数据

健壮性及网站测试

1、 重复添加同一首歌到同一个歌单会弹出错误提示框。

2、 服务器忙导致连接故障时会弹出提示框。

3、 无歌单时将一首歌加入歌单会改为创建歌单。同时将该歌曲加入到新建歌单中。

4、 密码错误/用户名错误/邮箱错误等等均会弹出提示框。

5、 用户上传的音乐文件若与服务器中已有的文件相同，会进行去重，只保留其一（防止重命名错误）。

6、 (注册登录等)对输入字符串长度进行限制，不允许空串，不允许过长串。

7、 限定允许读取的音频文件的格式（mp3/flac）和图片文件格式（png/jpg/bmp）与大小，防止注入。

8、 多用户同时访问时，不会产生冲突。

小组分工

本项目中，由我们自己编写的文件目录如下（在以下文件列表中我们隐藏了并非我们编写的文件组件，包括 bootbox/bootstrap/APlayer/Jquery 组件）：

```
.
└── web-music-main
    └── app
        ├── static
        │   ├── css
        │   │   ├── cover.css
        │   │   └──signin.css
        │   ├── js
        │   │   ├── randompic.js
        │   │   └── song_sheet.js
        │   └── resource
        │       ├── cover
        │       ├── flac
        │       ├── mp3
        │       ├── headpic
        │       └── lrc
        ├── templates
        │   ├── base.html
        │   ├── edit_profile.html
        │   ├── index.html
        │   ├── login.html
        │   ├── register.html
        │   └── user.html
        ├── __init__.py
        ├── find.py
        ├── forms.py
        ├── models.py
        ├── music.py
        ├── playerlist.py
        ├── songsheet.py
        └── routes.py
    └── main.py
```

高曾谊：

后端+前后端通讯

后端数据结构、数据库设计

后端功能实现：模糊查询；数据库增删改查；文件解析、维护

前端数据显示

网站测试、报告撰写(以上所有部分)、PPT 制作与答辩

编写及维护所有 py 文件及 js 文件，协助维护所有 html 文件及 css 文件

贾仁军：

前端框架设计、样式修改

播放逻辑、交互逻辑实现

歌曲、图片文件上传下载

网站测试、报告撰写(以上有关前端框架设计样式修改的部分，心得体会， web 技术展望及课程建议部分)

编写维护所有 html/css 文件，协助维护 forms.py, randompic.js 文件

心得体会

在此前的小作业中，我们接触到了 bootstrap.js 这一类模板 js 和 flask 的路由通讯原理。借由此前各小作业中应用 flask 的基础，我们尝试着应用这些学习过的技术从零开始来完成一个完整的 web 项目。在前端方面采用了原生的 html+css+js 的基础三件套。

多网页的设计采用 flask 路由控制视图函数的方式来完成。在选定题目之后对各个页面的样式设计使用了 html 的模板类方法，保证各个页面的风格尽可能统一。

数据库方面在参考之前所学的 pymysql 的基础上，借助 peewee 库对 pymysql 的进一步封装，更加简单、整洁地完成了对数据库增删改查的处理。

前端的使用体验我们希望是交互式+多平台的，所以没有使用基础写定的 css 样式，而是参考了 bootstrap 中的模板样式，自行设计出一套可交互的模板类。在音乐播放按钮的使用上，我们前期采用了 html5 自带的 audio 组件，但因其缺乏互动性，最终改用了 Aplayer.js 框架以实现更好的交互体验。（在使用 Aplayer.js 框架前我们甚至自己写 js 以解析 lrc 文件）

音乐上传方面，为了防止不同用户对相同歌曲的不同命名导致的歌曲重复上传以及上传格式的不统一，我们使用了 mutagen 来解析音频文件附带的文件信息以解决该问题。下载方面我们并没有使用前后端交互的方式完成，而是纯前端返回下载地址的形式，方便用户直接下载也减轻后端的负载压力。

由于时间因素，这个项目的最终完成情况基本达到我们我们的预期，完成度可以说是差强人意，但有待改进的部分仍有很多：比如说个人主页下信息提交页面逻辑仍不完善，歌曲评论区仍未开放等等问题，希望在后续的 web 开发学习中能够学会解决这些问题。

暑期我们也希望能自学 vue 等更新时代的技术，去更好地完善我们的项目。

web 技术展望及课程建议

这次作业我们使用的 web 技术还远远称不上新颖，使用的是传统的前端三件套+flask 的后端平台+mysql 数据库的组成方式，最多是在 js 方面使用了一些贴近现代的 api 以及在 css 方面参照了 bootstrap 的模板类，在后端和数据库方面的技术也只有 peewee 称得上紧跟时代。

但现在最主要的 web 项目开发中，前端技术使用的都是 vue.js、React、Angular 等次世代的新技术，学校里教学的这些古董完全不能满足当前 web 开发的前端需求。

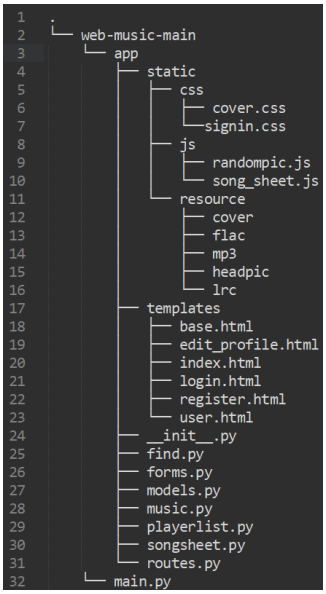
但这些新技术所需要的技术栈却也更为复杂，以 Vue.js 举例，完成一个完整的现代化页面，需要在传统的前端三件套之外使用 Axios 来完成前端的通信，使用 vue-router 管理页面的转移，使用 vuex 完成对网页状态的管理，使用 ElementUI 构建 UI 框架。可以说，现在实现一个 web 项目所需要的技术成本已经远远不是十年前所能比拟的了。

但在学校的 web 教学中，这些新兴技术的占比仍旧少的可怜，而我们在大量的从零开始的自学和漫长的 debug 过程中并不能对前端技术建立起完整的框架和结构，只是深入地学习到了 JavaScript 等语言的语法知识。对此，我们希望以后的 web 课程中能够增加这一类技术的支持和教学，使用小作业的形式让没有基础的同学们接触到更新颖的技术，而不是忍受漫长 debug 的痛苦，守着古老的前端三件套写出 00 年代风格的 web 应用。

源代码

详见项目文件夹 web-music-main.

文件目录



Python 文件

main.py——高曾谊编写维护

```
from app import app

if __name__ == '__main__':
    app.run(debug='True')
```

__init__.py——高曾谊编写维护

```
#from flask_sqlalchemy import SQLAlchemy
from flask_login import LoginManager
from flask import Flask
import peewee as pw
```


装 订 线

```
app = Flask(__name__)
app.config["SECRET_KEY"] = 'b8a0e5e48f7e4577a020b8502dcb7fc8'
login = LoginManager(app)
# py_peewee 连接的数据库名
db = pw.MySQLDatabase('flaskmusicplayer',
                      host='127.0.0.1',
                      user='root',
                      passwd='tongjigzy_02',
                      charset='utf8',
                      port=3306)


class BaseModel(pw.Model):
    class Meta:
        database = db  # 将实体与数据库进行绑定


from app import models  ##这一行必须在这里，不能移到上面去
# 连接数据库
db.connect()

# 创建数据表

models.User.create_table()

models.SongSheet.create_table()
from app import music, songsheet
from app import routes, find, playerlist
```

find.py——高曾谊编写维护

```
import re
from app.models import Music


def fuzzyfinder(user_input, collection):
    suggestions = []
    pattern = '.*?'.join(user_input)  # e.g. 'abc' -> 'a.*?b.*?c'
    for item in collection:
        match = re.search(pattern, item[1], re.I)  # 歌曲名
        if match:
            suggestions.append(item)
        match = re.search(pattern, item[2], re.I)  # 歌手名
        if match:
            suggestions.append(item)
        match = re.search(pattern, item[4], re.I)  #专辑名
        if match:
            suggestions.append(item)
    teplis = list(set(suggestions))  #去重
    teplis.sort(key=lambda x: x[0])  #排序
    return teplis


def find(inputstr):
    "找遍歌手名、歌曲名，模糊查询，返回一个四元组(id,music_name,singer_name,music_type)"
    collections = []
    Musics = Music.select().where(Music.id != '')  # 查全
    for i in Musics:
        collections.append(
            (i.id, i.music_name, i.singer_name, i.music_type, i.album_name))
```

```
    return fuzzyfinder(inputstr, collections)
```

forms.py——高曾谊编写维护，贾仁军协助维护

```
from flask_wtf import FlaskForm
from wtforms import StringField, PasswordField, BooleanField, SubmitField, TextAreaField
from wtforms.validators import DataRequired, ValidationError, Email, EqualTo, Length
from app.models import User


class LoginForm(FlaskForm):
    # DataRequired，当你在当前表格没有输入而直接到下一个表格时会提示你输入
    username = StringField('用户名', validators=[DataRequired(message='请输入用户名')])
    password = PasswordField('密码', validators=[DataRequired(message='请输入密码')])
    remember_me = BooleanField('记住我')
    submit = SubmitField('登录')


class RegistrationForm(FlaskForm):
    username = StringField('用户名', validators=[DataRequired(message='请输入用户名')])
    email = StringField('邮箱', validators=[DataRequired(
        message='请输入邮箱'), Email(message='不符合邮箱格式')])
    password = PasswordField('密码', validators=[DataRequired(message='请输入密码')])
    password2 = PasswordField(
        '重复密码', validators=[DataRequired(message='请再次输入密码
'), EqualTo('password', message='请确认密码是否相同')])
    submit = SubmitField('注册')
    # 校验用户名是否重复

    def validate_username(self, username):
        try:
            user = User.get(User.name == username)  # 查
        except:
            return  # 没查到
        raise ValidationError('用户名重复了，请您重新换一个呗!')
    # 校验邮箱是否重复

    def validate_email(self, email):
        try:
            user = User.get(User.email == email)  # 查
        except:
            return  # 没查到
        raise ValidationError('邮箱重复了，请您重新换一个呗!')


class EditProfileForm(FlaskForm):
    username = StringField('用户名', validators=[DataRequired(message='请输入用户名!')])
    about_me = TextAreaField('关于我', validators=[Length(min=0, max=140)])
    submit = SubmitField('提交')
```

models.py——高曾谊编写维护

```
from datetime import datetime
from re import VERBOSE
from werkzeug.security import generate_password_hash, check_password_hash
from hashlib import md5
from flask_login import UserMixin  # 不知道有没有用
from app import BaseModel
import peewee as pw
```

```
class User(UserMixin,
           BaseModel):  # 标准格式 继承自 BaseModel，直接关联 db，并且也继承了 Model Model 有提供增删查改的函数
    #id = pw.IntegerField(primary_key=True)  # 主键
    username = pw.CharField(max_length=64, index=True, unique=True)  # 唯一的用户名
    email = pw.CharField(max_length=120, index=True, unique=True)  # 唯一的邮箱
    about_me = pw.CharField(max_length=256)
    last_seen = pw.DateField(default=datetime.utcnow())
    password_hash = pw.CharField(max_length=128)

    def set_password(self, password):
        self.password_hash = str(generate_password_hash(password))

    def check_password(self, password):
        return check_password_hash(self.password_hash, password)

    def __repr__(self):
        return '<用户名:{}>'.format(self.username)

    def avatar(self, size):  # 根据邮箱自动获取分形图的 API
        digest = md5(self.email.lower().encode('utf-8')).hexdigest()
        return 'https://gravatar.zeruns.tech/avatar/{}?d=identicon&s={}'.format(
            digest, size)


class Music(BaseModel):  # 标准格式 继承自 BaseModel，直接关联 db，并且也继承了 Model Model 有提供增删查改的函数
    #id = pw.IntegerField(primary_key=True)
    music_name = pw.CharField(verbose_name='音乐名',
                              max_length=128,
                              null=False,
                              index=True)
    singer_name = pw.CharField(verbose_name='歌手名',
                               max_length=128,
                               null=False,
                               default='key')
    music_type = pw.CharField(verbose_name='歌曲类型',
                              max_length=16,
                              null=False,
                              default='mp3')
    album_name = pw.CharField(verbose_name='专辑名',
                              max_length=128,
                              null=False,
                              default='NO FOUND')

    def __repr__(self):
        return '<Music {}>'.format(self.music_name + ' - ' + self.singer_name)


class SongSheet(BaseModel):
    #id = pw.IntegerField(primary_key=True)
    userid = pw.IntegerField(index=True)  # 唯一的用户名,对应
    sheet_name = pw.CharField(verbose_name='歌单名',
                              max_length=32,
                              null=False,
                              index=True)
    musiclist = pw.CharField(verbose_name='歌曲列表', max_length=1024, default='')

    #由于没有列表，只好用字符串了,歌曲 id 之间以'-'分割
    def __repr__(self):
```


装 订 线

```
        return '<SongSheet {}{}>'.format(self.user_id, self.sheet_name)
```

music.py——高曾谊编写维护

```
from app import models
from mutagen import File
from PIL import Image
import os

lrcpath = "./app/static/resource/lrc/"
flacpath = "./app/static/resource/flac/"
mp3path = "./app/static/resource/mp3/"
coverpath = "./app/static/resource/cover/"
cover = Image.open("./app/static/resource/默认封面.jpg")


def rename(srcFile, dstFile):
    #print(srcFile, "------", dstFile)
    if srcFile == dstFile:
        return
    os.rename(srcFile, dstFile)
    '''
    try:
        os.rename(srcFile, dstFile)
    except Exception as e:
        print(str(e), repr(e), sep='\n')
        print('rename file fail\r\n')
    else:
        print('rename file success\r\n')
    '''


def formatname(string):
    list = ['\\', '/', ':', '*', '?', '"', '<', '>', '|']
    for i in list:
        string = string.replace(i, '_')
    return string


def getflacinfo():
    teplist = os.listdir(flacpath)
    for i in teplist:
        path = flacpath + i
        afile = File(path)
        author = afile.tags["Artist"][0]  # 作者
        author = formatname(author)
        title = afile.tags["Title"][0]  # 标题
        title = formatname(title)
        album = afile.tags["Album"][0]  # 专辑
        #print(album)  # 专辑
        try:
            models.Music.get(models.Music.singer_name == author,
                             models.Music.music_name == title,
                             models.Music.album_name == album)
        except:  #未找到
            models.Music(singer_name=author,
                         music_name=title,
                         music_type='flac',
                         album_name=album).save()
            try:
                artwork = afile.pictures[0].data
                with open(coverpath + author + ' - ' + title + '.jpg',
```

```
                        'wb') as img:
                    img.write(artwork)  # write artwork to n
            except:
                cover.save(coverpath + author + ' - ' + title + '.jpg')
                #可以自己添加一张默认封面

            srcFile = path
            dstFile = flacpath + author + ' - ' + title + '.flac'
            try:
                rename(srcFile, dstFile)  #歌曲重命名
            except:  #去重
                os.remove(dstFile)
                rename(srcFile, dstFile)  #歌曲重命名

            try:
                rename(lrcpath + i[:-5] + ".lrc",
                       lrcpath + author + ' - ' + title + ".lrc")  #歌词重命名
            except:
                pass


def getmp3info():
    teplist = os.listdir(mp3path)
    for i in teplist:
        path = mp3path + i
        afile = File(path)
        author = afile.tags["TPE1"].text[0]  # 作者
        author = formatname(author)
        title = afile.tags["TIT2"].text[0]  # 标题
        title = formatname(title)
        album = afile.tags["TALB"].text[0]  # 专辑
        #print(album)  # 专辑
        try:
            models.Music.get(models.Music.singer_name == author,
                             models.Music.music_name == title,
                             models.Music.album_name == album)
        except:  #未找到
            models.Music(singer_name=author,
                         music_name=title,
                         music_type='mp3',
                         album_name=album).save()
            try:
                artwork = afile.tags["APIC:"].data
                with open(coverpath + author + ' - ' + title + '.jpg',
                          'wb') as img:
                    img.write(artwork)  # write artwork to n
            except:
                cover.save(coverpath + author + ' - ' + title +
                           '.jpg')  #可以自己添加一张默认封面

            srcFile = path
            dstFile = mp3path + author + ' - ' + title + '.mp3'  # 命名规则： 歌手名 - 歌曲名
            try:
                rename(srcFile, dstFile)  #歌曲重命名
            except:  #去重
                os.remove(dstFile)
                rename(srcFile, dstFile)  #歌曲重命名
            try:
                rename(lrcpath + i[:-4] + ".lrc",
                       lrcpath + author + ' - ' + title + ".lrc")  #歌词重命名
            except:
                pass
```


装 订 线

```
if models.Music.table_exists():
    models.Music.drop_table()  #先删表
models.Music.create_table()
getflacinfo()
getmp3info()

from app.playerlist import initplayerlist


def initdatabase():
    if models.Music.table_exists():
        models.Music.drop_table()  #先删表
    models.Music.create_table()
    getflacinfo()
    getmp3info()
    initplayerlist()


'''
def GetMusicInfo(s):
    type = s.split('.')[-1]
    u = s.rstrip('.' + type).split(' - ')
    return (u[0], u[1], type)
# 命名规则： 歌手名 - 歌曲名
musicstr = os.listdir(mp3path) + os.listdir(flacpath)  # 获取歌曲/歌手名字
for i in musicstr:
    tep = GetMusicInfo(i)
    models.Music(singer_name=tep[0], music_name=tep[1],
                 music_type=tep[2]).save()
'''
```

playerlist.py——高曾谊编写维护

```
from app.models import Music

playerlist = []  #播放列表


def initplayerlist():
    global playerlist
    playerlist.clear()
    Musics = Music.select().where(Music.id != '')  # 查全
    for i in Musics:
        dic = {}
        dic['music_id'] = i.id
        dic['singer_name'] = i.singer_name
        dic['music_name'] = i.music_name
        dic['music_type'] = i.music_type
        dic['album_name'] = i.album_name
        dic["path"] = i.singer_name + ' - ' + i.music_name + '.' + i.music_type
        playerlist.append(dic)


initplayerlist()
```

songsheet.py——高曾谊编写维护

```
from app import app
```

```
from app.models import Music, User, SongSheet
from flask_login import current_user
from flask import Flask, make_response, request, redirect, render_template, url_for, jsonify


@app.route('/getsheetname/', methods=['GET', 'POST'])
def GetSheetName():
    "查询一个用户所有的表的名字"
    if request.method == 'POST':
        userid = current_user.id
        #lis = [{'text': "Choose one...", "value": "Choose one..."}]
        lis = []
        p = SongSheet.select().where(SongSheet.userid == userid)
        for i in p:
            lis.append({'text': i.sheet_name, "value": i.sheet_name})
        return make_response(jsonify(lis))


@app.route('/getsheet/', methods=['GET', 'POST'])
def GetSheet():
    "初始化用户首页时，前端向传递表单名，后端向前端传递歌单信息"
    if request.method == 'POST':
        userid = current_user.id
        sheet_name = request.form.get("sheet_name")
        lis = []
        try:
            p = SongSheet.get(SongSheet.userid == userid,
                              SongSheet.sheet_name == sheet_name)  # 查
        except:  #不存在该歌单，或是用户未登录
            pass
        if p.musiclist == "":
            pass
        else:
            teplist = p.musiclist.split('-')
            for i in teplist:
                p = Music.get(id=int(i))
                dic = {}
                dic['music_id'] = i.id
                dic['singer_name'] = i.singer_name
                dic['music_name'] = i.music_name
                dic['music_type'] = i.music_type
                dic['album_name'] = i.album_name
                dic["path"] = i.singer_name + ' - ' + i.music_name + '.' + i.music_type
                lis.append(dic)
        resp = make_response(jsonify(lis))
        print(resp)
        return resp


@app.route('/addsheet/', methods=['POST'])
def AddSheet():
    "某用户创建歌单，需传入歌单名(认为同一用户不能创建两个同名歌单)"
    if request.method == 'POST':
        userid = current_user.id
        sheet_name = request.form.get("sheet_name")
        #print("-----------------------")
        #print(username, "----", sheet_name)
        dic = {}
        try:  #同一个用户不能有同样名字的歌单
            SongSheet.get(SongSheet.userid == userid,
                          SongSheet.sheet_name == sheet_name)  # 查
        except:  #没查到,可以建立
            SongSheet(userid=userid, sheet_name=sheet_name).save()
```


装 订 线

```
            dic['isok'] = 1
        else:  #查到了,不能建立
            dic['isok'] = 0
        return make_response(jsonify(dic))


@app.route('/deletesheet/', methods=['POST'])
def DeleteSheet():
    "某用户删除歌单，需传入歌单名"
    if request.method == 'POST':
        userid = current_user.id
        sheet_name = request.form.get("sheet_name")
        dic = {}
        try:  #同一个用户不能有同样名字的歌单
            p = SongSheet.get(SongSheet.userid == userid,
                              SongSheet.sheet_name == sheet_name)  # 查
        except:  #没查到,没法删
            dic['isok'] = 0
        else:
            #用户名和表名必须同时对应，不然会误删
            SongSheet.delete().where(
                SongSheet.userid == userid,
                SongSheet.sheet_name == sheet_name).execute()
            dic['isok'] = 1
        return make_response(jsonify(dic))


@app.route('/addsong/', methods=['POST'])
def AddSong():
    "某用户在某首歌单中加入一首歌,需传入歌单名及歌曲 id"
    if request.method == 'POST':
        userid = current_user.id
        sheet_name = request.form.get("sheet_name")
        music_id = request.form.get("music_id")
        dic = {}
        try:
            p = SongSheet.get(SongSheet.userid == userid,
                              SongSheet.sheet_name == sheet_name)  # 查
            Music.get(id=music_id)
        except:  #没找到对应歌单/歌曲
            dic['isok'] = 0
        else:
            dic['isok'] = 1
            if p.musiclist == '':  #空
                SongSheet.update({
                    SongSheet.musiclist: str(music_id)
                }).where(SongSheet.userid == userid,
                         SongSheet.sheet_name == sheet_name).execute()  # 改
            else:  #非空
                teplist = p.musiclist.split('-')
                if str(music_id) not in teplist:
                    teplist.append(str(music_id))
                    SongSheet.update({
                        SongSheet.musiclist: '-'.join(teplist)
                    }).where(SongSheet.userid == userid,
                             SongSheet.sheet_name == sheet_name).execute()  # 改
                else:
                    dic['isok'] = -1
        return make_response(jsonify(dic))


@app.route('/deletesong/', methods=['POST'])
def DeleteSong():
```


装 订 线

```
    "某用户在某首歌单中删除一首歌,需传入歌单名及歌曲 id"
    if request.method == 'POST':
        userid = current_user.id
        sheet_name = request.form.get("sheet_name")
        music_id = request.form.get("music_id")
        dic = {}
        try:
            p = SongSheet.get(SongSheet.userid == userid,
                              SongSheet.sheet_name == sheet_name)  # 查
            Music.get(id=music_id)
        except:  #没找到对应歌单/歌曲
            dic['isok'] = 0
        if p.musiclist == '':  #歌单中元素为空
            dic['isok'] = -1
        else:
            dic['isok'] = 1
            teplist = p.musiclist.split('-')
            try:
                teplist.remove(str(music_id))
                SongSheet.update({
                    SongSheet.musiclist: '-'.join(teplist)
                }).where(SongSheet.userid == userid,
                         SongSheet.sheet_name == sheet_name).execute()  # 改
            except:  #teplist 里面没有对应元素，没法删
                dic['isok'] = -1
        return make_response(jsonify(dic))
```

routes.py——高曾谊编写维护

```
from app.music import initdatabase
from app.find import find
from flask import render_template, flash, redirect, url_for, request
from app import app
from app.forms import LoginForm
from flask_login import current_user, login_user, logout_user, login_required
from app.models import SongSheet, User, Music
from werkzeug.urls import url_parse
from app.forms import RegistrationForm
from datetime import datetime
from app.forms import EditProfileForm
from app import login
from app.playerlist import playerlist


@login.user_loader
def load_user(id):
    try:
        user = User.get(User.id == id)  # 查
    except:
        user = None
    return user


@app.route("/", methods=['GET', 'POST'])
def rootindex():
    return redirect(url_for('index'))  # 重定向


@app.route('/index', methods=['GET', 'POST'])
# 这样，必须登录后才能访问首页了,会自动跳转至登录页
def index():
    if request.method == 'POST':
```

```
        searchname = request.form.get("search_content")
        tep = find(searchname)
        searchres = []
        for i in tep:
            dic = {}
            dic['music_id'] = i[0]
            dic['music_name'] = i[1]
            dic['singer_name'] = i[2]
            dic['music_type'] = i[3]
            dic['album_name'] = i[4]
            dic["path"] = i[2] + ' - ' + i[1] + '.' + i[3]
            searchres.append(dic)
        return render_template("index.html", Musics=searchres)
    return render_template('index.html', Musics=playerlist)


@app.route('/login', methods=['GET', 'POST'])
def login():
    # 判断当前用户是否验证，如果通过的话返回首页
    if current_user.is_authenticated:
        return redirect(url_for('index'))

    form = LoginForm()
    # 对表格数据进行验证
    if form.validate_on_submit():
        # 根据表格里的数据进行查询，如果查询到数据返回 User 对象，否则返回 None
        try:
            user = User.get(User.username == form.username.data)  # 查
        except:
            flash('无效的用户名,请检查输入或注册')
            # 然后重定向到登录页面
            return redirect(url_for('login'))
        # 查到了，判断密码
        if not user.check_password(form.password.data):
            # 如果用户不存在或者密码不正确就会闪现这条信息
            flash('密码错误')
            # 然后重定向到登录页面
            return redirect(url_for('login'))

        # 这是一个非常方便的方法，当用户名和密码都正确时来解决记住用户是否记住登录状态的问题
        login_user(user, remember=form.remember_me.data)
        # 此时的 next_page 记录的是跳转至登录页面是的地址
        next_page = request.args.get('next')
        # 如果 next_page 记录的地址不存在那么就返回首页
        if not next_page or url_parse(next_page).netloc != '':
            next_page = url_for('index')
        return redirect(next_page)
    return render_template('login.html', form=form)


import requests


@app.route('/register', methods=['GET', 'POST'])
def register():
    # 判断当前用户是否验证，如果通过的话返回首页
    if current_user.is_authenticated:
        return redirect(url_for('index'))
    form = RegistrationForm()
    if form.validate_on_submit():
        namefound = True
        emailfound = True
        try:
```


装 订 线

```
            User.get(User.username == form.username.data)
        except:
            namefound = False
        try:
            User.get(User.email == form.email.data)
        except:
            emailfound = False
        if namefound == False and emailfound == False:
            user = User(username=form.username.data, email=form.email.data)
            user.set_password(form.password.data)
            user.save()
            url = user.avatar(256)
            strhtml = requests.get(url)  #Get 方式获取网页数据
            with open("./app/static/resource/headpic/" + str(user.id) + '.jpg',
                      'wb') as fw:
                fw.write(strhtml.content)

            flash('恭喜你成为我们网站的新用户!')
            return redirect(url_for('login'))
        else:
            flash('该用户名或邮箱已被注册!')
            return render_template('register.html', form=form)
    return render_template('register.html', form=form)


# 设置允许的文件格式
ALLOWED_EXTENSIONS_PIC = set(['png', 'jpg', 'JPG', 'PNG', 'bmp', 'BMP'])
ALLOWED_EXTENSIONS_MUSIC = set(['mp3', 'flac', 'MP3', 'FLAC'])


def allowed_file(filename, type):
    return '.' in filename and filename.rsplit('.', 1)[1] in type


import os
from PIL import Image


@app.route('/user/', methods=['GET', 'POST'])
@login_required
def user():
    if request.method == 'POST':
        f = request.files['musicfile']
        if not (f and allowed_file(f.filename, ALLOWED_EXTENSIONS_MUSIC)):
            flash("请上传合适的文件! ")
        else:
            path = "./app/static/resource/" + f.filename.rsplit(
                '.', 1)[1] + "/" + f.filename
            f.save(path)
            initdatabase()

    lists = []
    tepuser = User.get(User.username == current_user.username)  # 查
    posts = SongSheet.select().where(SongSheet.userid == current_user.id)
    for i in posts:
        newdic = []
        newdic.append(i.sheet_name)
        if i.musiclist == "":
            newdic.append('')
        else:
            teplist = i.musiclist.split("-")
            searchres = []
            print(teplist)
```


装 订 线

```
            for j in teplist:
                dic = {}
                dic['music_id'] = j
                p = Music.get(id=j)
                dic['music_name'] = p.music_name
                dic['singer_name'] = p.singer_name
                dic['music_type'] = p.music_type
                dic['album_name'] = p.album_name
                dic["path"] = p.singer_name + ' - ' + p.music_name + '.' + p.music_type
                searchres.append(dic)
            newdic.append(searchres)
        lists.append(newdic)

    return render_template('user.html',
                           user=tepuser,
                           posts=lists,
                           posts_len=len(lists))


@app.route('/logout')
def logout():
    logout_user()
    return redirect(url_for('index'))


@app.before_request
def before_request():
    if current_user.is_authenticated:
        current_user.last_seen = datetime.utcnow()
        current_user.save()  # db.session.commit()


@app.route('/edit_profile', methods=['GET', 'POST'])
@login_required
def edit_profile():
    form = EditProfileForm()
    if request.method == 'POST' and form.validate_on_submit():
        try:
            User.get(User.username == form.username.data)
        except:
            current_user.username = form.username.data
            current_user.about_me = form.about_me.data
            current_user.save()
            flash('您的提交已变更.')
            return redirect(url_for('edit_profile'))
        if current_user.username == form.username.data:
            current_user.about_me = form.about_me.data
            current_user.save()
            flash('您的提交已变更.')
            return redirect(url_for('edit_profile'))
        else:
            flash("该用户名已被注册!")
    elif request.method == 'GET':
        form.username.data = current_user.username
        form.about_me.data = current_user.about_me
    return render_template('edit_profile.html', title='个人资料编辑', form=form)


@app.route('/upload_pic', methods=['GET', 'POST'])
@login_required
def upload_pic():
    form = EditProfileForm()
    if request.method == 'POST':
```


装 订 线

```
        print(request.files)
        f = request.files['picfile']
        if not (f and allowed_file(f.filename, ALLOWED_EXTENSIONS_PIC)):
            flash("请上传合适的文件！")
        else:
            path = "./app/static/resource/headpic/" + str(
                current_user.id) + '.' + f.filename.rsplit('.', 1)[1]
            f.save(path)
            im = Image.open(path).resize((256, 256)).convert('RGB')
            os.remove(path)
            im.save("./app/static/resource/headpic/" + str(current_user.id) +
                    '.jpg')  #以用户 id.jpg 保存在 headpic 文件夹中
            flash('您的提交已变更.')
    return redirect(url_for('edit_profile'))
```

HTML 文件

base.html——贾仁军编写维护，高曾谊协助维护

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    {% if title %}
    <title>{{ title }} - 音乐播放器</title>
    {% else %}
    <title>欢迎来到 TJ 音乐播放器!</title>
    {% endif %}
    <link href="../static/css/bootstrap.min.css" rel="stylesheet" />
    <link href="../static/css/cover.css" rel="stylesheet" />
    <link href="../static/css/signin.css" rel="stylesheet" />
    <script type="text/javascript" src="../static/js/booxbox.js"></script>
    <script type="text/javascript" src="../static/js/jquery.min.js"></script>
    <script type="text/javascript" src="../static/js/bootstrap.min.js"></script>
    <script type="text/javascript" src="../static/js/bootbox.min.js"></script>
    <script type="text/javascript" src="../static/js/bootbox.js"></script>
    <script type="text/javascript" src="../static/js/bootstrap.js"></script>
    <script type="text/javascript" src="../static/js/jquery.tmpl.js"></script>
    <style>
      .bootbox-body {
        color: black !important;
        font-size: large !important;
      }
      .navbar-brand {
        color: rgb(28, 13, 83) !important;
      }
      a {
        color: blue !important;
      }
      .bootbox-body {
        color: black !important;
        font-size: large !important;
        text-align: left !important;
      }
      .modal-header {
        color: black !important;
      }
    </style>
  </head>
  <body>
    <div class="site-wrapper">
      <div class="site-wrapper-inner">
        <div class="cover-container">
```

```
          <div class="masthead clearfix">
            <div class="inner">
              <nav class="navbar navbar-default">
                <div class="container-fluid">
                  <div class="navbar-header">
                    <a class="navbar-brand">TJ 音乐播放器</a>
                  </div>
                  <div id="navbar" class="navbar-collapse collapse">
                    <ul class="nav navbar-nav">
                      <li><a href="{{ url_for('index') }}">首页</a></li>
                      {% if current_user.is_anonymous %}
                      <li><a href="{{ url_for('login') }}">登录</a></li>
                      <li><a href="{{ url_for('register') }}">注册</a></li>
                      {% else %}
                      <li>
                        <a
                          href="{{ url_for('user', username=current_user.username) }}"
                          >用户中心</a
                        >
                      </li>
                      <li><a href="{{ url_for('logout') }}">退出</a></li>
                      {% endif %}
                    </ul>
                    <div class="navbar-header">
                      <a class="navbar-brand"
                        >欢迎 {{ current_user.username }} !</a
                      >
                    </div>
                  </div>
                </div>
              </nav>
            </div>
          </div>
          <div id="inner" class="inner cover">
            {% with messages = get_flashed_messages() %} {% if messages %}
            <script>
              var messages = {{messages|safe}};
              var tep='';
              for (var i=0; i<messages.length; i++) {
                  tep+=messages[i];
              }
              bootbox.alert({
              //title: "提示",
              size:"big",
              message:tep,
              })
            </script>
            {% endif %} {% endwith %}
          </div>
        </div>
        {% block content %} {% endblock %}
      </div>
    </div>
  </body>
</html>
```

edit_profile.html——贾仁军编写维护，高曾谊协助维护

```
{% extends "base.html" %} {% block content %}
<h1>个人资料编辑</h1>

<form action="/upload_pic" method="post" enctype="multipart/form-data"  onreset="myreset()">
  <label id="realBtn" class="btn btn-info">
```


装 订 线

```
    <input
    id="uploadBtn"
    type="file"
    class="btn btn-primary"
    name="picfile"
    required="required"
    style="left:-9999px;position:absolute;"
    onchange="change(this)"
    />
    <span id="text">选择上传头像文件</span>
  </label>
    <input class="btn btn-primary" type="submit" value="提交" />
    <input class="btn btn-primary" type="reset" value="重置" />
</form>


<form action="" method="post" >
  {{ form.hidden_tag() }}
  <br><br>
  <div class="col-md-12">
    <div class="col-md-5"></div>
    <div class="col-md-2">
    {{ form.username.label }}<br />
    {{ form.username(size=32,class="form-control") }}<br />
    {% for error in form.username.errors %}
    <span style="color: red">[{{ error }}]</span>
    {% endfor %}
    </div>
  </div>
  <br>
  <div class="col-md-12">
    <div class="col-md-4"></div>
    <div class="col-md-4">
    {{ form.about_me.label }}<br />
    {{ form.about_me(cols=50, rows=4,class="form-control") }}<br />
    {% for error in form.about_me.errors %}
    <span style="color: red">[{{ error }}]</span>
    {% endfor %}
    <input class="btn btn-primary" type="submit" value="提交" />
    <input class="btn btn-primary" type="reset" value="重置" />
  </div>
  </div>
</form>

<script>
  function change(obj)
    {
      var t=document.getElementById('text');
      t.innerHTML=(obj.value).slice(12);
    }
    function myreset()
    {
      var t=document.getElementById('text');
      t.innerHTML="选择上传头像文件";
    }
</script>
{% endblock %}
```

index.html——贾仁军编写维护，高曾谊协助维护

```
{% extends 'base.html' %} {% block content %}
<script src="../static/js/song_sheet.js"></script>
<link rel="stylesheet" href="../static/css/APlayer.min.css" />
```

```
<br /><br /><br />
<div class="col-md-12">
  <div class="col-md-3"></div>
  <div class="col-md-6">
    <form
      action=""
      method="post"
      name="submit"
      class="text-center"
      novalidate="novalidate"
    >
      <input
        type="text"
        class="form-control"
        placeholder="请输入感兴趣的音乐名、歌手名"
        name="search_content"
        maxlength="128"
      />
    </form>
  </div>
</div>
<br /><br />
<div class="col-md-12">
  <!--audio 标签控制：https://www.w3school.com.cn/tags/tag_audio.asp-->
  <div class="col-md-12">
    <div style="height: 490px; overflow: auto">
      <table class="table" id="music_list">
        <tr>
          <th>歌曲名</th>
          <th>歌手名</th>
          <th>播放</th>
          <th>下载</th>
          {% if not current_user.is_anonymous %}
          <th>加入歌单</th>
          {% endif %}
        </tr>
        {% for music in Musics %}
        <tr>
          <th>{{music.music_name}}</th>
          <th>{{music.singer_name}}</th>
          <th>
            <input
              id="{{music.music_id}}"
              name="play"
              type="image"
              src="../static/css/play.svg"
              height="25"
              width="25"
              value="0"
              onclick="player({{music.music_id}})"
            />
          </th>
          <th>
            <a href="../static/resource/{{music.music_type}}/{{music.singer_name}} - {{music.musi
c_name}}.{{music.music_type}}"
            download="{{music.singer_name}} - {{music.music_name}}.{{music.music_type}}"
            >
          <img src="../static/css/download.svg"
          height="25"
          width="25">
          </a>
          </th>
          {% if not current_user.is_anonymous %}
          <th>
```


装 订 线

```
            <button
              class="btn btn-primary"
              onclick="add_song({{music.music_id}})"
            >
              +
            </button>
          </th>
          {% endif %}
        </tr>
        {% endfor %}
      </table>
    </div>
  </div>
</div>
<div class="col-md-12">
  <div class="col-md-2"></div>
  <div class="col-md-8">
    <div id="aplayer" class="aplayer" value="0"></div>
  </div>
</div>
<script src="../static/js/APlayer.min.js"></script>
<script>
  var ap = new APlayer({
      container: document.getElementById('aplayer'),
      lrcType: 3,
  });
  ap.on('pause',function(){
    var s = document.getElementsByName("play");
      for (var i = 0; i < s.length; i++) {
        s[i].value = 0;
        s[i].src = "../static/css/play.svg";
      }
  });
  ap.on('play',function(){
    var a = document.getElementById("aplayer");
    var t = document.getElementById(a.value + "");
    t.src = "../static/css/pause.svg";
  });

    function getrealindex(M,id) {
      for(var i=0;i<M.length;i++)
      {
        if(M[i].music_id==id)return M[i];
      }
      return M[0];
    }
    function player(id)
    {
      var M={{Musics|safe}};
      var t=document.getElementById(id);
      if (t.value==1){
          var a=document.getElementById("aplayer");
          a.value=id;
          t.value=0;
          t.src="../static/css/play.svg"
          ap.pause();
        }
      else{
        var s=document.getElementsByName("play");
        for (var i=0;i<s.length;i++)
        {
          s[i].value=0;
          s[i].src="../static/css/play.svg";
        }
        t.value=1;
        var a=document.getElementById("aplayer");
```

```
        a.value=id;
        TEP=getrealindex(M,id);
        ap.destroy();
        ap=new APlayer({
          container: document.getElementById('aplayer'),
          lrcType: 3,
          audio: {
              name: TEP.music_name,
              artist: TEP.singer_name,
              url: "../static/resource/"+TEP.music_type+"/"+TEP.singer_name+" - "+TEP.music_name+
"."+TEP.music_type,
              cover: "../static/resource/cover/"+TEP.singer_name+" - "+TEP.music_name+".jpg",
              lrc: "../static/resource/lrc/"+TEP.singer_name+" - "+TEP.music_name+".lrc"
          }
      });
        ap.on('pause',function(){
        var s = document.getElementsByName("play");
          for (var i = 0; i < s.length; i++) {
            s[i].value = 0;
            s[i].src = "../static/css/play.svg";
          }
        });
        ap.on('play',function(){
        var a = document.getElementById("aplayer");
        var t = document.getElementById(a.value + "");
        t.src = "../static/css/pause.svg";
        });
        ap.play();
        t.src="../static/css/pause.svg";
    }
    }
</script>
{% endblock %}
```

login.html——贾仁军编写维护，高曾谊协助维护

```
{% extends 'base.html' %} {% block content %}

<h2 class="form-signin-heading">登 录</h2>
<p>
  还没注册? <a href="{{ url_for('register') }}"><u>点击一下就可以注册哦!</u></a>
</p>
<form class="form-signin" action="" method="post" novalidate="novalidate">
  {{ form.hidden_tag() }} {{ form.username.label(class="sr-only") }}<br />
  {{
  form.username(size=32,class="form-control",placeholder="请输入用户名...")}}<br />
  {% for error in form.username.errors %}
  <span style="color: red">{{ error }}</span>
  {% endfor %} {{ form.password.label(class="sr-only")}}<br />
  {{
  form.password(size=32,class="form-control",placeholder="请输入密码...")}}<br />
  {% for error in form.password.errors %}
  <span style="color: red">{{ error }}</span>
  {% endfor %}
  <div>{{ form.remember_me() }} {{ form.remember_me.label }}</div>
  {{ form.submit(class="btn btn-lg btn-primary") }}
</form>
{% endblock %}
```

register.html——贾仁军编写维护，高曾谊协助维护

```
{% extends "base.html" %}
```

```
{% block content %}
    <h2 class="form-signin-heading">注 册</h2>
    <form action="" method="post" class="form-signin" novalidate="novalidate">
        {{ form.hidden_tag() }}
        <p>
            {{ form.username.label(class="sr-only") }}<br>
            {{ form.username(size=32,class="form-control",placeholder="请输入用户名...")}}<br>
            {% for error in form.username.errors %}
            <span style="color: red;">{{ error }}</span>
            {% endfor %}
        </p>
        <p>
            {{ form.email.label(class="sr-only") }}<br>
            {{ form.email(size=64,class="form-control",placeholder="请输入邮箱...")}}<br>
            {% for error in form.email.errors %}
            <span style="color: red;">{{ error }}</span>
            {% endfor %}
        </p>
        <p>
            {{ form.password.label(class="sr-only") }}<br>
            {{ form.password(size=32,class="form-control",placeholder="请输入密码...")}}<br>
            {% for error in form.password.errors %}
            <span style="color: red;">{{ error }}</span>
            {% endfor %}
        </p>
        <p>
            {{ form.password2.label(class="sr-only") }}<br>
            {{ form.password2(size=32,class="form-control",placeholder="请再次输入密码...")}}<br>
            {% for error in form.password2.errors %}
            <span style="color: red;">{{ error }}</span>
            {% endfor %}
        </p>
        <p>{{ form.submit(class="btn btn-lg btn-primary") }}</p>
    </form>
{% endblock %}
```

user.html——贾仁军编写维护，高曾谊协助维护

```
{% extends "base.html" %} {% block content %}
<link rel="stylesheet" href="../static/css/APlayer.min.css" />
<script src="../static/js/song_sheet.js"></script>
<script src="../static/js/randompic.js"></script>
<script>
  document.getElementById("uploadBtn").onchange = function () {
    document.getElementById("uploadFile").value = this.files[0].name;
    document.getElementById("uploadFile").style.color = "white";
  };
</script>
<div class="col-md-12">
<table>
  <tr valign="top">
    <td>
      <img id="headpic" src="../static/resource/headpic/{{user.id}}.jpg" />
    </td>
    <td>
      <h2>User: {{ user.username }}</h2>
      {% if user.about_me %}
      <p>{{ user.about_me }}</p>
      {% endif %} {% if user.last_seen %}
      <p>最近登录: {{ user.last_seen }}</p>
      {% endif %} {% if user == current_user %}
      <p><a href="{{ url_for('edit_profile') }}">个人资料编辑</a></p>
```

```
      {% endif %}
      <p>
        <button
          type="button"
          class="btn btn-primary"
          onclick="add_song_sheet()"
        >
          <font size="1">新建歌单</font>
        </button>
      </p>
      <p>
        <button
          type="button"
          class="btn btn-primary"
          onclick="delete_song_sheet()"
        >
          <font size="1">删除歌单</font>
        </button>
      </p>
      <!--<p>
        <button type="button" class="btn btn-primary" onclick="randompic()">
          <font size="1">随机头像</font>
        </button>
      </p>-->
      <form action="" enctype="multipart/form-data" method="POST" onreset="myreset()">
        <div>
        <label id="realBtn" class="btn btn-info">
          <input
          id="uploadBtn"
          type="file"
          class="btn btn-primary"
          name="musicfile"
          required="required"
          style="left:-9999px;position:absolute;"
          onchange="change(this)"
          />
          <span id="text">选择上传音乐文件</span>
        </label>
        </div>
        <div>
        <input class="btn btn-primary" type="submit" value="提交" />
        <input class="btn btn-primary" type="reset" value="重置"/>
        </div>
      </form>
    </td>
  </tr>
</table>
<div class="tabbable" id="tabs-779775">
  <ul class="nav nav-tabs">
    {% for i in range(posts_len) %}
    <li>
      <a href="#panel-{{i}}" data-toggle="tab">{{posts[i][0]}}</a>
    </li>
    {% endfor %}
  </ul>
  <div class="tab-content">
    {% for i in range(posts_len) %}
    <div class="tab-pane" id="panel-{{i}}">
      <table class="table" id="music_list-{{i}}">
        <thead>
          <tr>
            <th>歌曲名</th>
            <th>歌手名</th>
            <th>播放</th>
            {% if not current_user.is_anonymous %}
```


装 订 线

```
            <th>加入歌单</th>
            <th>移出歌单</th>
            {% endif %}
          </tr>
        </thead>
        <tbody>
          {% for music in posts[i][1] %}
          <tr>
            <th>{{music.music_name}}</th>
            <th>{{music.singer_name}}</th>
            <th>
              <input
                id="{{music.music_id}}"
                name="play"
                type="image"
                src="../static/css/play.svg"
                height="25"
                width="25"
                value="0"
                onclick="player({{i}},{{music.music_id}})"
              />
            </th>
            <th>
              <button
                class="btn btn-primary"
                onclick="add_song({{music.music_id}})"
              >
                +
              </button>
            </th>
            <th>
              <button
                class="btn btn-primary"
                onclick="delete_song('{{posts[i][0]}}',{{music.music_id}})"
              >
                -
              </button>
            </th>
          </tr>
          {% endfor %}
        </tbody>
      </table>
    </div>
    {% endfor %}
  </div>
</div>

<div class="col-md-12">
  <div class="col-md-7">
    <div id="aplayer" class="aplayer" value="0"></div>
  </div>
</div>
</div>
<script src="../static/js/APlayer.min.js"></script>
<script>
  var ap = new APlayer({
      container: document.getElementById('aplayer'),
      fixed: true,
      lrcType: 3,
  });
  ap.on('pause',function(){
    var s = document.getElementsByName("play");
      for (var i = 0; i < s.length; i++) {
        s[i].value = 0;
        s[i].src = "../static/css/play.svg";
```

```
      }
  });
  ap.on('play',function(){
    var a = document.getElementById("aplayer");
    var t = document.getElementById(a.value + "");
    t.src = "../static/css/pause.svg";
  });

    function getrealindex(M,id) {
      for(var i=0;i<M.length;i++)
      {
        if(M[i].music_id==id)return M[i];
      }
      return M[0];
    }

    var posts={{posts|safe}};
    function player(i,id)
    {
      var M=posts[i][1];
      console.log(M);
      var t=document.getElementById(id);
      if (t.value==1){
          var a=document.getElementById("aplayer");
          a.value=id;
          t.value=0;
          t.src="../static/css/play.svg"
          ap.pause();
        }
      else{
        var s=document.getElementsByName("play");
        for (var i=0;i<s.length;i++)
        {
          s[i].value=0;
          s[i].src="../static/css/play.svg";
        }
        t.value=1;
        var a=document.getElementById("aplayer");
        a.value=id;
        TEP=getrealindex(M,id);
        ap.destroy();
        ap=new APlayer({
          container: document.getElementById('aplayer'),
          lrcType: 3,
          audio: {
              name: TEP.music_name,
              fixed: true,
              artist: TEP.singer_name,
              url: "../static/resource/"+TEP.music_type+"/"+TEP.singer_name+" - "+TEP.music_name+
"."+TEP.music_type,
              cover: "../static/resource/cover/"+TEP.singer_name+" - "+TEP.music_name+".jpg",
              lrc: "../static/resource/lrc/"+TEP.singer_name+" - "+TEP.music_name+".lrc"
          }
      });
        ap.on('pause',function(){
        var s = document.getElementsByName("play");
          for (var i = 0; i < s.length; i++) {
            s[i].value = 0;
            s[i].src = "../static/css/play.svg";
          }
        });
        ap.on('play',function(){
        var a = document.getElementById("aplayer");
        var t = document.getElementById(a.value + "");
        t.src = "../static/css/pause.svg";
        });
```

```
        ap.play();
        t.src="../static/css/pause.svg";
    }
  }

    function change(obj)
    {
      var t=document.getElementById('text');
      t.innerHTML=(obj.value).slice(12);
    }
    function myreset()
    {
      var t=document.getElementById('text');
      t.innerHTML="选择上传音乐文件";
    }
</script>

{% endblock %}
```

CSS 文件

cover.css——贾仁军编写维护，高曾谊协助维护

```
a,
a:focus,
a:hover {
  color: #fff;
}

.btn-default,
.btn-default:hover,
.btn-default:focus {
  color: #333;
  text-shadow: none;
  background-color: #fff;
  border: 1px solid #fff;
}

html,
body {
  height: 100%;
  background-color: rgb(18, 233, 222);
}
body {
  color: #fff;
  text-align: center;
  text-shadow: 0 1px 3px rgba(0, 0, 0, 0.5);
}

.site-wrapper {
  display: table;
  width: 100%;
  height: 100%;
  min-height: 100%;
  -webkit-box-shadow: inset 0 0 100px rgba(0, 0, 0, 0.5);
  box-shadow: inset 0 0 100px rgba(0, 0, 0, 0.5);
}
.site-wrapper-inner {
  display: table-cell;
  vertical-align: top;
}
.cover-container {
  margin-right: auto;
  margin-left: auto;
```

```
}

.inner {
  padding: 30px;
}

.masthead-brand {
  margin-top: 10px;
  margin-bottom: 10px;
}

.masthead-nav > li {
  display: inline-block;
}
.masthead-nav > li + li {
  margin-left: 20px;
}
.masthead-nav > li > a {
  padding-right: 0;
  padding-left: 0;
  font-size: 16px;
  font-weight: 700;
  color: #fff;
  color: rgba(255, 255, 255, 0.75);
  border-bottom: 2px solid transparent;
}
.masthead-nav > li > a:hover,
.masthead-nav > li > a:focus {
  background-color: transparent;
  border-bottom-color: #a9a9a9;
  border-bottom-color: rgba(255, 255, 255, 0.25);
}
.masthead-nav > .active > a,
.masthead-nav > .active > a:hover,
.masthead-nav > .active > a:focus {
  color: #fff;
  border-bottom-color: #fff;
}

@media (min-width: 768px) {
  .masthead-brand {
    float: left;
  }
  .masthead-nav {
    float: right;
  }
}

.cover {
  padding: 0 20px;
}
.cover .btn-lg {
  padding: 10px 20px;
  font-weight: 700;
}

.mastfoot {
  color: #999;
  color: rgba(255, 255, 255, 0.5);
}

@media (min-width: 768px) {
  .masthead {
    color: #fff;
    position: fixed;
    top: 0;
```

```
  }
  .mastfoot {
    position: fixed;
    bottom: 0;
  }
  .site-wrapper-inner {
    vertical-align: middle;
  }
  .masthead,
  .mastfoot,
  .cover-container {
    width: 100%;
  }
}

@media (min-width: 992px) {
  .masthead,
  .mastfoot,
  .cover-container {
    width: 700px;
  }
}
body {
  padding-top: 20px;
  padding-bottom: 20px;
}

.navbar {
  margin-bottom: 20px;
```

signin.css——贾仁军编写维护，高曾谊协助维护

```
.form-signin {
  max-width: 330px;
  padding: 15px;
  margin: 0 auto;
}
.form-signin .form-signin-heading,
.form-signin .checkbox {
  margin-bottom: 10px;
}
.form-signin .checkbox {
  font-weight: 400;
}
.form-signin .form-control {
  position: relative;
  -webkit-box-sizing: border-box;
  -moz-box-sizing: border-box;
  box-sizing: border-box;
  height: auto;
  padding: 10px;
  font-size: 16px;
}
.form-signin .form-control:focus {
  z-index: 2;
}
.form-signin input[type="email"] {
  margin-bottom: -1px;
  border-bottom-right-radius: 0;
  border-bottom-left-radius: 0;
}
.form-signin input[type="password"] {
  margin-bottom: 10px;
```

```
  border-top-left-radius: 0;
  border-top-right-radius: 0;
}
```

JAVASCRIPT 文件

randompic.js——高曾谊编写维护，贾仁军协助维护

```
//生成从 minNum 到 maxNum 的随机数
function randomNum(minNum, maxNum) {
  switch (arguments.length) {
    case 1:
      return parseInt(Math.random() * minNum + 1, 10);
      break;
    case 2:
      return parseInt(Math.random() * (maxNum - minNum + 1) + minNum, 10);
      break;
    default:
      return 0;
      break;
  }
}
function randompic() {
  var num = randomNum(5, 1000);
  var path = "../static/resource/pic/pic (" + num + ").jpg";
  var tep = document.getElementById("headpic");
  console.log(num);
  console.log(path);
  tep["src"] = path;
}

// 上传图片
$("#picfile").change(function () {
  var filePath = $(this).val();
  let fr = new FileReader(); //创建 new FileReader()对象
  let imgObj = this.files[0]; //获取图片
  fr.readAsDataURL(imgObj); //将图片读取为 DataURL
  if (
    filePath.indexOf("jpg") != -1 ||
    filePath.indexOf("JPG") != -1 ||
    filePath.indexOf("PNG") != -1 ||
    filePath.indexOf("png") != -1
  ) {
    fr.onload = function () {
      $(".setselfmsg .userpic img").attr("src", this.result);
    };
  } else {
    confirm("您未上传文件，或者您上传文件类型有误！");
    return false;
  }
});
```

song_sheet.js——高曾谊编写维护

```
function show_song_sheet(sheet_name) {
  $.ajax({
    type: "POST",
    url: "/getsheet/",
    data: { sheet_name: sheet_name },
    async: false,
    success: function (data) {
```

```
      console.log(data);
      for (var name in data) {
        (function (name) {
          //这个是 function 里 i，即 function 的形参，也可以换成 j，换成什么变量名都无所谓
          console.log(name);
          var x = document.getElementById("user_list").insertRow();
          var cell1 = x.insertCell();
          cell1.innerHTML = data[name]["music_name"];
          var cell2 = x.insertCell();
          cell2.innerHTML = data[name]["singer_name"];
          var cell = document.createElement("audio");
          cell.controls = "controls";
          cell.preload = "auto";
          cel.src = "../static/resource/mp3/" + data[name]["path"];
          x.appendChild(cell);
        })(name); //这是循环中的 i,被作为参数传入
      }
    },
  });
}
function get_sheet_name() {
  $.ajax({
    type: "POST",
    url: "/getsheetname/",
    data: {},
    async: false,
    success: function (data) {
      console.log(data);
      return data;
    },
    error: function () {
      return null;
    },
  });
}

function add_song_sheet() {
  console.log("add sheet begin");
  bootbox.prompt({
    title: "新建歌单",
    message: "请输入歌单名",
    //"输入歌单名字 传入 result 里面"
    callback: function (result) {
      console.log(result);
      if (result != null && result != "") {
        $.ajax({
          url: "/addsheet/",
          type: "POST",
          data: { sheet_name: result },
          dataType: "json",
          success: function (res) {
            console.log(res);
            if (res["isok"] == 1) {
              //bootbox.alert("Successfully add!");
              ////location.reload();
              window.location.href = "/user/";
            } else if (res["isok"] == 0) {
              bootbox.alert("添加失败，该歌单已存在");
            } else {
              bootbox.alert("Disconnect... Please try again later");
            }
          },
          error: function (res) {
            bootbox.alert("Disconnect... Please try again later");
          },
```

```
        });
      } else;
    },
  });
}
function delete_song_sheet() {
  console.log("delete sheet begin");
  $.ajax({
    type: "POST",
    url: "/getsheetname/",
    data: {},
    async: false,
    success: function (data) {
      console.log(data);
      bootbox.prompt({
        inputType: "radio",
        inputOptions: data,
        title: "删除歌单",
        message: "请选择待删除的歌单",
        buttons: {
          confirm: {
            label: "Yes",
          },
          cancel: {
            label: "No",
          },
        },
        callback: function (result) {
          if (result != null && result != "") {
            $.ajax({
              url: "/deletesheet/",
              type: "POST",
              data: { sheet_name: result },
              dataType: "json",
              success: function (res) {
                console.log(res);
                if (res["isok"] == 1) {
                  //bootbox.alert("Successfully deleted!");
                  location.reload();
                } else if (res["isok"] == 0) {
                  bootbox.alert("Delete failed!NO FOUND!");
                } else {
                  bootbox.alert("Disconnect... Please try again later");
                }
              },
              error: function (res) {
                bootbox.alert("Disconnect... Please try again later");
              },
            });
          } else;
        },
      });
      return null;
    },
    error: function () {
      return null;
    },
  });
}
function add_song(music_id) {
  console.log("add song begin");
  $.ajax({
    type: "POST",
    url: "/getsheetname/",
    data: {},
```

```
    async: false,
    success: function (data) {
      console.log(data.length);
      if (data.length == 0) {
        bootbox.prompt({
          title: "您还没有歌单哦，请新建歌单",
          message: "请输入歌单名",
          //"输入歌单名字 传入 result 里面"
          callback: function (result) {
            console.log(result);
            $.ajax({
              url: "/addsheet/",
              type: "POST",
              data: { sheet_name: result },
              dataType: "json",
              success: function (res) {
                console.log(res);
                if (res["isok"] == 1) {
                  $.ajax({
                    url: "/addsong/",
                    type: "POST",
                    data: { sheet_name: result, music_id: music_id },
                    dataType: "json",
                    success: function (res) {
                      console.log(res);
                      if (res["isok"] == 1) {
                        //bootbox.alert("Successfully add!");
                        location.reload();
                      } else if (res["isok"] == 0) {
                        bootbox.alert("Add failed!NO FOUND!");
                      } else if (res["isok"] == -1) {
                        bootbox.alert("该歌曲已存在");
                      } else {
                        bootbox.alert("Disconnect... Please try again later");
                      }
                    },
                    error: function (res) {
                      bootbox.alert("Disconnect... Please try again later");
                    },
                  });
                } else if (res["isok"] == 0) {
                  bootbox.alert("添加失败，该歌单已存在");
                } else {
                  bootbox.alert("Disconnect... Please try again later");
                }
              },
              error: function (res) {
                bootbox.alert("Disconnect... Please try again later");
              },
            });
          },
        });
      } else {
        bootbox.prompt({
          inputType: "radio",
          inputOptions: data,
          title: "加入歌曲",
          message: "请选择歌单",
          buttons: {
            confirm: {
              label: "Yes",
            },
            cancel: {
              label: "No",
            },
```

```
          },
          callback: function (result) {
            if (result != null && result != "") {
              $.ajax({
                url: "/addsong/",
                type: "POST",
                data: { sheet_name: result, music_id: music_id },
                dataType: "json",
                success: function (res) {
                  console.log(res);
                  if (res["isok"] == 1) {
                    //bootbox.alert("Successfully add!");
                    location.reload();
                  } else if (res["isok"] == 0) {
                    bootbox.alert("Add failed!NO FOUND!");
                  } else if (res["isok"] == -1) {
                    bootbox.alert("该歌曲已存在");
                  } else {
                    bootbox.alert("Disconnect... Please try again later");
                  }
                },
                error: function (res) {
                  bootbox.alert("Disconnect... Please try again later");
                },
              });
            } else;
          },
        });
      }
    },
  });
}
function delete_song(sheet_name, music_id) {
  console.log("delete song begin");
  console.log(sheet_name);
  console.log(music_id);
  bootbox.confirm({
    message: "确定从该歌单移除这首歌曲吗？",
    buttons: {
      confirm: {
        label: "Yes",
      },
      cancel: {
        label: "No",
      },
    },

    callback: function (result) {
      if (result) {
        $.ajax({
          url: "/deletesong/",
          type: "POST",
          data: { sheet_name: sheet_name, music_id: music_id },
          dataType: "json",
          success: function (res) {
            console.log(res);
            if (res["isok"] == 1) {
              //bootbox.alert("Successfully deleted!");
              location.reload();
            } else if (res["isok"] == 0 || res["isok"] == -1) {
              bootbox.alert("Delete failed!NO FOUND!");
            } else {
              bootbox.alert("Disconnect... Please try again later");
            }
          },
```


装 订 线

```
          error: function (res) {
            bootbox.alert("Disconnect... Please try again later");
          },
        });
      } else;
    },
  });
}
```