实现文档

一、任务分工

2213912 罗劲

- 负责Vue前端代码
- 前后端连接的实现
- 后端api控制器的框架编写
- 部分后端api函数的调整
- Yii2后端的构建
- 独自完成Yii2中models的编写，实现表项的传入
- 负责PPT制作

2210636 李天翊

- 负责Vue前端代码
- 为每个表项进行主键设置与外键连接
- 实现表与表之间联动的触发器
- 数据验证与处理
- 负责网页数据内容的查找
- 独自数据库在后端的CRUD功能

2213741 廖远菁

- 负责Vue前端代码
- 前端主页的设计
- 后端与Mysql数据库的连接
- 完成了大部分后端api函数
- 独自完成数据库的设计与构建
- 部分后端api函数的调整
- 完成了每个文档的大部分内容

二、文件目录展示

前端部分文件目录展示：

```
music-project
    ├─node_modules
    │  └─...
    ├─public
    │    ├─assets
    │    │  ├─audio
    │    │  │  └─...   //存放音频资源
    │    │  ├─images
    │    │  │  └─...   //存放图片资源（歌单封面，视频封面）
    │    │  ├─video
    │    │  │  └─...   //存放视频资源（mv）
```

```
    │   │  └─logo
    │   ├─favicon.ico  //网站图标
    │   └─index.html
    ├─src
    │   ├─assets
    │   │  ├─css
    │   │  │  └─main.css
    │   │  └─images
    │   ├─components   //页面组件渲染
    │   │  ├─LikeButton.vue
    │   │  ├─MusicPlayer.vue
    │   │  └─NavBar.vue
    │   ├─router
    │   │   └─index.js
    │   ├─views       //页面资源渲染
    │   │  ├─AboutUsView.vue
    │   │  ├─homepage.vue
    │   │  ├─LoginView.vue
    │   │  ├─songListView.vue
    │   │  ├─UserPageView.vue
    │   │  ├─VideoDetailView.vue
    │   │  └─VideoListView.vue
    │   ├─APP.vue
    │   ├─config.js
    │   ├─evenBus.js
    │   └─main.js
    └─...
```

后端部分文件目录展示:

```
music_backend
    ├─controllers
    │   ├─AdminsController.php
    │   ├─ApiController.php   //处理前端发送数据
    │   ├─ArtistController.php
    │   ├─MvlikesController.php
    │   └─...
    ├─models     //借助yii辅助生成
    │   ├─Admins.php
    │   ├─Artists.php
    │   ├─ArtistsSearch.php
    │   ├─MvlikesSearch.php
    │   └─...
    ├─views
    │   ├─admin     //借助yii辅助生成
    │   │   ├─_form_.php
    │   │   ├─_search.php
    │   │   ├─creat.php
    │   │   ├─index.php
    │   │   ├─update.php
    │   │   └─view.php
    │   ├─artists  //结构类似
```

```
│   │   └─...
│   ├─mvlikes
│   │   └─...
│   ├─playlist-songs
│   │   └─...
│   └─...
└─...
```

三、功能设计与实现

1. 前端功能设计

- 主页设计 主页的设计包括用户展示、歌曲列表、视频展示等。主要功能点包括：

  a. 用户可以通过主页面导航到其他功能页面（如歌曲列表、视频详情、用户个人页面等）。 b. 主页展示歌曲和视频的推荐列表，用户可以根据这些推荐播放音乐或视频。 c. 使用 Vue.js 渲染动态内容，并通过 Vue Router 实现页面之间的导航。

- 用户登录与注册 用户可以在登录页面输入账号与密码进行登录。 提供注册功能，允许用户通过输入必要信息进行账号注册。 登录与注册通过后端 API 进行验证。

- 音乐播放功能 通过 MusicPlayer.vue 组件实现播放功能。 支持播放、暂停、音量调节等操作。 音乐资源通过音频文件存放在 public/assets/audio 目录中。

- 歌单管理 用户可以通过点击歌单进入歌单详情页，查看歌单中的歌曲。

- 视频播放功能 通过 VideoListView.vue 和 VideoDetailView.vue 组件分别展示视频列表和视频详细信息。用户点击视频封面可以跳转到视频详情页面进行播放。

2. 后端功能设计

- API 控制器 后端使用 Yii2 框架实现所有与前端交互的 API。主要控制器包括：
  - ApiController.php：处理前端发送的请求，返回音乐、视频等相关数据。
  - AdminsController.php：处理管理员相关的请求，如添加或删除歌曲、视频等。
  - ArtistController.php：处理歌手相关的请求，展示歌手的信息，管理歌手列表。
  - MvlikesController.php：处理视频（MV）的喜好、点赞等操作。

- 数据库模型 使用 Yii2 框架自带的 Gii 工具生成数据库模型。 Admins.php：管理员表模型。 Artists.php：歌手信息表模型。 Mvlikes.php：MV 点赞信息表模型。 Playlist-songs.php：歌单与歌曲的多对多关系表。

- 数据库设计与管理 数据库的设计涉及多个表，包括用户表、歌曲表、视频表、歌手表、歌单表等。表之间通过外键进行关联，支持多对多的关系。

四、技术架构与选择

1. 前端技术栈

Vue.js：用于构建用户界面，使用 Vue Router 进行页面的导航和 Vuex 进行状态管理。 Axios：用于向后端发送 HTTP 请求。 Element UI：一个基于 Vue 的 UI 组件库，用于构建美观的页面组件。

2. 后端技术栈 Yii2：作为后端框架，提供高效的数据库操作、控制器和视图的生成工具。 MySQL：关系型数据库，用于存储用户、歌曲、视频等信息。 PHP：编写后端逻辑，处理前端请求和数据交互。

3. 数据库设计 表设计：包括用户表、歌曲表、视频表、歌单表、歌手表、MV 点赞表等。 外键关联：确保表与表之间的数据一致性，避免出现孤立数据。

五、总结与展望

本项目实现了一个功能丰富的音乐播放平台，前端使用 Vue.js 和 Element UI，后端使用 Yii2 框架和 MySQL 数据库。 目前已完成了所有基础功能的实现，后期可以加入更多个性化功能（如推荐系统、用户评论、歌单分享等）。 项目中还有许多优化空间，例如提升 API 的性能、前端的响应速度等。