电影售票系统详细设计报告

目录

[酷猫音乐详细设计报告 1](#_Toc10930505)

[1 引言 3](#_Toc10930506)

[1.1标识 3](#_Toc10930507)

[1.2系统概述 3](#_Toc10930508)

[1.3文档概述 4](#_Toc10930509)

[1.4功能需求 4](#_Toc10930510)

[1.5基线 4](#_Toc10930511)

[2.引用文件 5](#_Toc10930512)

[3.详细系统设计 5](#_Toc10930513)

[3.1 播放功能 5](#_Toc10930514)

[3.2 数据功能 5](#_Toc10930515)

[3.3推荐功能 6](#_Toc10930516)

[3.4 基本设计概念和处理流程 6](#_Toc10930517)

[4.数据描述 7](#_Toc10930518)

[5.程序设计说明 10](#_Toc10930519)

[5.1程序描述 10](#_Toc10930520)

[5.2功能 67](#_Toc10930521)

[5.3性能 67](#_Toc10930522)

[5.4输入项 67](#_Toc10930523)

[5.5输出项 67](#_Toc10930524)

[5.6算法 68](#_Toc10930525)

[5.7流程逻辑 70](#_Toc10930526)

[6.接口： 72](#_Toc10930527)

[7.注释设计 73](#_Toc10930528)

[8.限制条件 73](#_Toc10930529)

[9.测试计划 73](#_Toc10930530)

[10.尚未解决的问题 73](#_Toc10930531)

[11.注解 73](#_Toc10930532)

[12.总结 73](#_Toc10930533)

# 引言

现在，随着时代的发展，人们现在的生活节奏越来越快，生活工作压力也随之变大。很多人都在寻找着适合自己的减压方式，让自己的生活工作压力有所缓解。毋庸置疑的是，看电影是一种不错的解压方式，无论是缓解自己的负面情绪，还是让自己的状态更加完美，看电影都可以做到。

现在互联网的发展十分迅速，互联网的功能也十分强大。借助互联网，让我们看电影不再那么麻烦。现在网上有许多电影售票系统，可以说都是互联网崛起的功劳。许多售票系统的功能都大致相同，电影的管理，电影的搜索以及电影售票的功能。在线电影售票系统乘着互联网的东风，让我们看电影的方式越来越多样化，也让我们看电影的选择越来越多样化。这个在线电影售票系统主要就是实现电影售票的基本功能，借助互联网，实现电影的排片以及一些基本的搜索功能。当然，这个电影售票系统也能够实现选座功能。

## 1.1标识

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 文件状态：  [ ]草稿  [ √ ]正式发布  [ ]正在修改 | 报告编号 | 电影售票系统 | | |
| 当前版本 | V1.0 | | |
| 编写人 | 本小组成员 | 编写日期 | 2020-6-9 |
| 审批人 | 本小组成员 | 审批日期 | 2019-6-9 |
| 保密级别 | 3 | | |

## 1.2系统概述

在线电影售票系统主要是利用互联网的优势，结合一些现有的网上电影售票系统，提供更加人性化的电影售票操作，包括座位的显示以及选择等等。不仅如此，该系统还提供一系列的基本操作，包括影片的评价，还有评论、用户推荐这些功能。当然，这个系统还增加了搜索操作，可以根据影片名称等进行搜索。

编写在线电影售票系统详细设计说明书的目的在于，从详细设计的角度明确在线电影售票系统项目的控件、功能范围以及它的设计思路，使系统开发人员和产品管理人员明确产品功能，可以有针对性的进行系统开发、测试、验收等各方面的工作。

，

读者：项目经理、概要设计人员，编码人员，测试人员。

## 1.3文档概述

在线电影售票系统总共有两个管理系统，后台管理系统和前端服务系统。后台管理系统主要是对用户基本信息进行管理，对用户的信息进行加密等等。后台管理系统是整个系统的安全系统，这个系统会对用户的信息进行严格保护，防止用户的信息泄露。该系统还是信息收集系统，这个系统会对用户的喜好数据进行收集并且分析，然后进行歌曲推荐。这个系统是连接网络的桥梁，这个系统会向前端系统发送必要的数据。而前端系统主要功能是接受来自后台系统的数据，并且进行展示。后台系统会根据用户的操作，包括点击输入等等，将一些必要的数据发送给后台，让后台访问数据库将对应的数据返回。也就是说，前端服务系统是用户和后台的桥梁，它的存在是让用户可以更加方便的获取想要的数据，并且隔绝用户直接的访问数据，这样有效的保护了数据，让数据以及一些基本信息更加安全。

本系统结合自己看电影时的经验，让这个系统更加贴合用户，让这个系统更加人性化。

## 1.4功能需求

1. 系统中有多个电影院，系统管理员可以完成电影院的维护，系统管理员可以为每个电影院指派1个电影院管理员；

2. 电影院管理员定期维护本电影院即将上映的电影信息；

4. 网民可以根据时间、电影名称、电影院名称进行查询，查询到自己中意的电影后，注册的网民可以在网上完成订票，并进行网上支付；

5. 系统能够对指定时间、电影院、电影名字进行统计分析，以便分析出受欢迎的电影

## 1.5基线

1**.** 在线电影售票系统软件开发计划书 V1.0

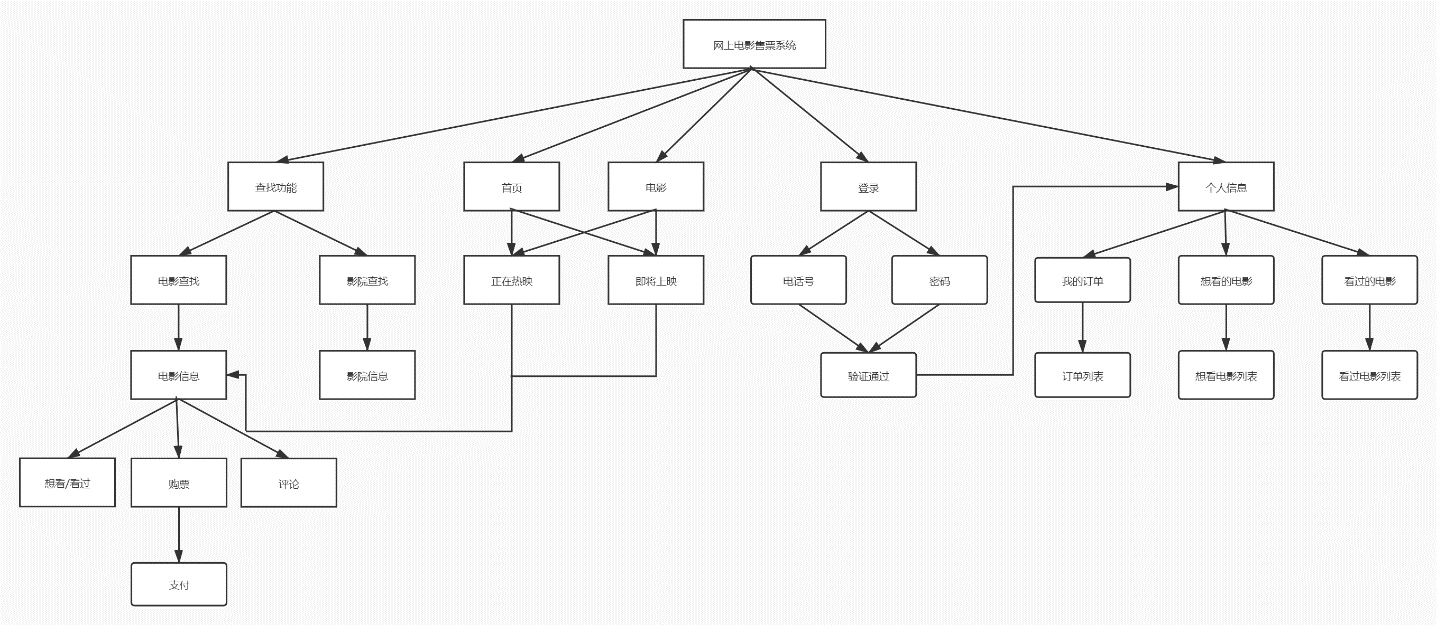
2. 在线电影售票系统需求分析报告 V1.0

3. 在线电影售票系统软件测试计划 V1.0

4. 在线电影售票系统概要设计说明书 V1.0

# 2.引用文件

# 3.详细系统设计



## 3.1基本功能

(a) 基于web的电影购票网站

(b) 注册登录用户可在线浏览热映电影信息，电影排行榜，;

(c) 注册登录用户查看附近影院，支持在线购票，实时支付等；

(d) 用户在观影结束后，可以对影院环境以及影片质量进行评价

(e) 系统UI界面简洁大方，操作简单，实时更新电影影院数据。

(f) 具有良好的平台可移植性，支持多人同时在线访问。

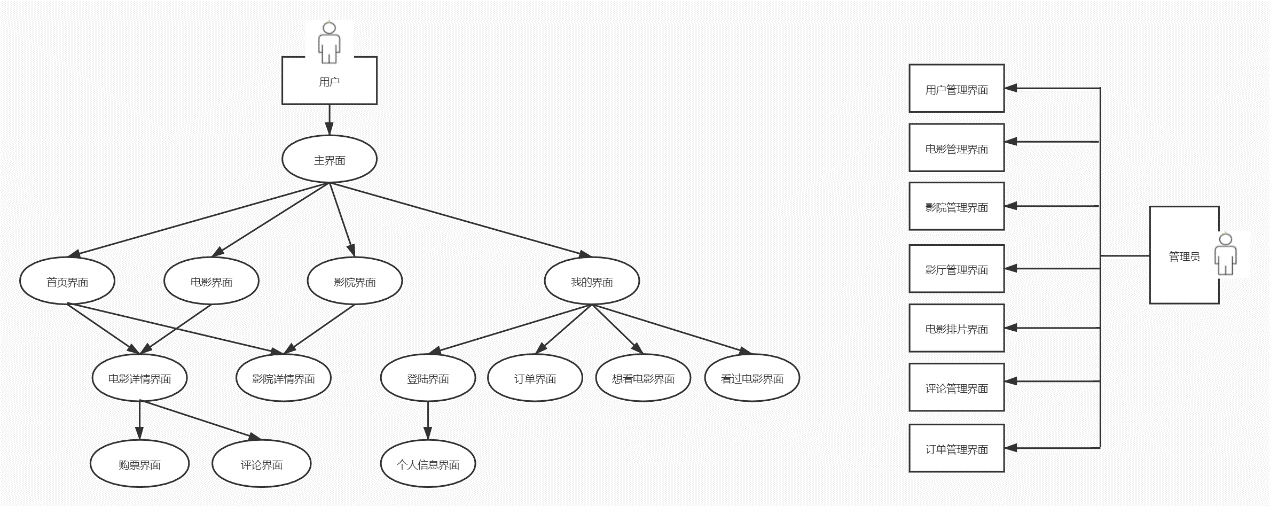
(g) 网页刷新时间等待时间不超过2s；用户能够流畅使用各项功能。

## 3.2附加功能

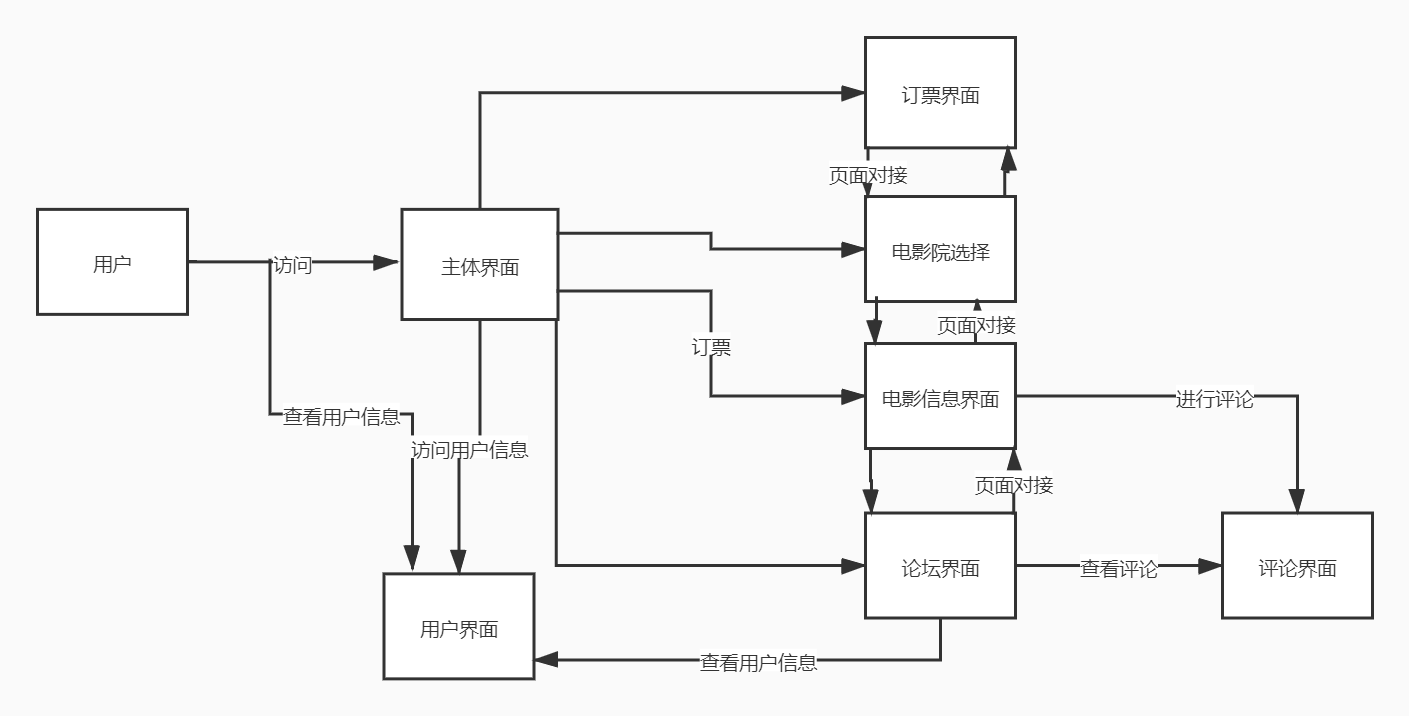
1. 安全与保密功能系统涉及用户私密信息，对于整个系统，需要完整的权限控制，每个用户分配特定的用户视图所应有的访问权限，防止某些人恶意的攻击系统，修改原始记录。设置必要的异常处理机制，比如当出现系统未知的操作时。也会弹出相应的错误提示等，以尽可能地保证用户体验。
2. 客户账号管理功能：管理人员能够列表显示用户的帐号信息、禁用或删除选定的用户帐号，在收到客户请求之后也可更改客户的账号信息。
3. 通知功能：管理人员可通过软件内消息向用户发送通知。
4. 电子邮件/短信服务：管理人员能够在客户账号信息出现异常时通过绑定的电子邮箱/手机号码向用户发送邮件/短信提醒。
5. 评论维护功能：对于违规评论，用户可向管理人员发送反馈，反馈人数达到一定程度即向管理人员显示该条消息，管理人员可以对其进行删除并发送消息通知该评论所属用户

## 3.3 基本设计概念和处理流程

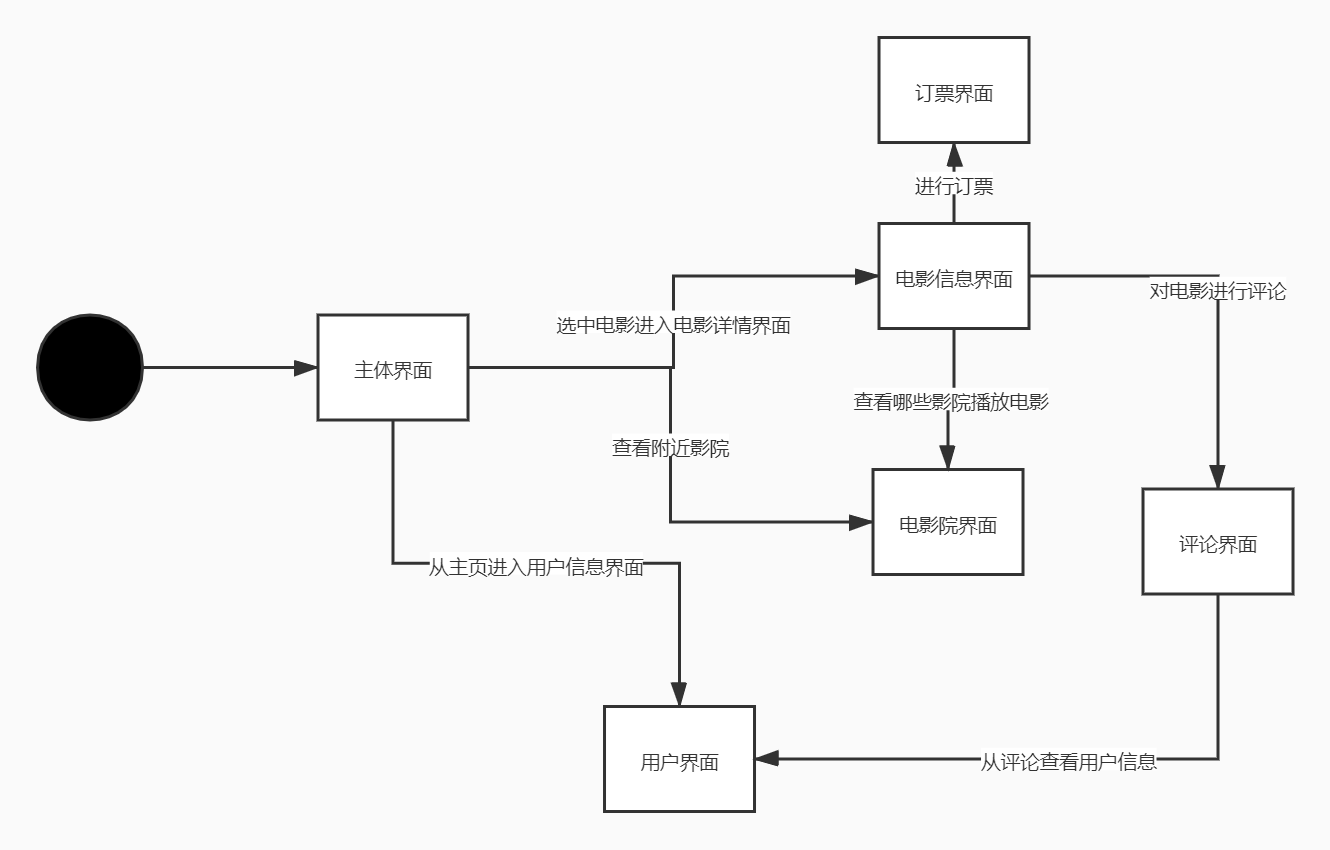
该网络售票系统对应着一个Web类型的解决方案。包括以下几大模块：



# 4.数据描述



界面分工：



后台数据表情况：

-- MySQL dump 10.13 Distrib 8.0.23, for Win64 (x86\_64)

--

-- Host: localhost Database: db\_film

-- ------------------------------------------------------

-- Server version 8.0.23

--

-- Table structure for table `t\_admin`

--

DROP TABLE IF EXISTS `t\_admin`;

CREATE TABLE `t\_admin` (

`admin\_id` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '管理员id',

`name` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '管理员姓名',

`password` varchar(100) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '管理员密码',

`sex` varchar(10) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '管理员性别',

`phone` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '管理员联系电话',

`birthday` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '出生年月日',

`avatar` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '管理员头像',

PRIMARY KEY (`admin\_id`) USING BTREE

) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci ROW\_FORMAT=DYNAMIC;

--

-- Table structure for table `t\_cinema`

--

DROP TABLE IF EXISTS `t\_cinema`;

CREATE TABLE `t\_cinema` (

`cinema\_id` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '影院id',

`cinema\_name` varchar(100) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '影院名称',

`cinema\_phone` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '影院电话',

`province` varchar(50) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '影院所在省份',

`city` varchar(50) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '影院所在市',

`county` varchar(50) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '影院所在区县',

`specified\_address` varchar(255) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '影院详细地址',

PRIMARY KEY (`cinema\_id`) USING BTREE

) ENGINE=InnoDB AUTO\_INCREMENT=15 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci ROW\_FORMAT=DYNAMIC;

--

-- Table structure for table `t\_comment`

--

DROP TABLE IF EXISTS `t\_comment`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

CREATE TABLE `t\_comment` (

`comment\_id` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '评论id',

`user\_id` int unsigned DEFAULT NULL COMMENT '用户id',

`movie\_id` int unsigned DEFAULT NULL COMMENT '电影id',

`user\_score` int unsigned DEFAULT NULL COMMENT '用户评分',

`comment\_content` text CHARACTER SET utf8 COLLATE utf8\_unicode\_ci COMMENT '用户评论内容',

`comment\_date` datetime(6) DEFAULT NULL COMMENT '评论日期',

`support\_num` int unsigned DEFAULT NULL COMMENT '点赞数',

`is\_pass` int DEFAULT '0' COMMENT '评论是否通过审核',

`support\_user` text CHARACTER SET utf8 COLLATE utf8\_unicode\_ci COMMENT '点赞用户数组',

PRIMARY KEY (`comment\_id`) USING BTREE,

KEY `t\_comment\_ibfk\_1` (`user\_id`) USING BTREE,

KEY `t\_comment\_ibfk\_2` (`movie\_id`) USING BTREE,

CONSTRAINT `t\_comment\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `t\_user` (`user\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `t\_comment\_ibfk\_2` FOREIGN KEY (`movie\_id`) REFERENCES `t\_movie` (`movie\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB AUTO\_INCREMENT=59 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci ROW\_FORMAT=DYNAMIC;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `t\_hall`

--

DROP TABLE IF EXISTS `t\_hall`;

CREATE TABLE `t\_hall` (

`hall\_id` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '影厅id',

`cinema\_id` int unsigned DEFAULT NULL COMMENT '影院id',

`name` varchar(50) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '影厅名称',

PRIMARY KEY (`hall\_id`) USING BTREE,

KEY `t\_hall\_ibfk\_1` (`cinema\_id`) USING BTREE,

CONSTRAINT `t\_hall\_ibfk\_1` FOREIGN KEY (`cinema\_id`) REFERENCES `t\_cinema` (`cinema\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB AUTO\_INCREMENT=41 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci ROW\_FORMAT=DYNAMIC;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `t\_movie`

--

DROP TABLE IF EXISTS `t\_movie`;

CREATE TABLE `t\_movie` (

`movie\_id` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '电影id',

`name` varchar(50) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '电影名称',

`poster` varchar(100) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '电影海报',

`director` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '导演',

`actor` varchar(100) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '主演',

`movie\_long` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '电影片长',

`language` varchar(50) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '电影语言版本',

`intro` text CHARACTER SET utf8 COLLATE utf8\_unicode\_ci COMMENT '电影简介',

`type` varchar(50) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '电影类型',

`public\_date` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '电影上映时间',

`wish\_num` int unsigned DEFAULT '0' COMMENT '想看人数',

`score` decimal(4,1) DEFAULT NULL COMMENT '电影综合评分',

PRIMARY KEY (`movie\_id`) USING BTREE

) ENGINE=InnoDB AUTO\_INCREMENT=39 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci ROW\_FORMAT=DYNAMIC;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `t\_order`

--

DROP TABLE IF EXISTS `t\_order`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!50503 SET character\_set\_client = utf8mb4 \*/;

CREATE TABLE `t\_order` (

`order\_id` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '订单id',

`user\_id` int unsigned DEFAULT NULL COMMENT '用户id',

`schedule\_id` int unsigned DEFAULT NULL COMMENT '影院id',

`order\_phone` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '下单手机',

`order\_date` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户下单时间',

`ticket\_num` int unsigned DEFAULT NULL COMMENT '购买电影票数',

`ticket\_total\_price` decimal(10,2) unsigned DEFAULT NULL COMMENT '电影票单价',

`order\_seat\_info` varchar(100) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '座位信息',

`pay\_type` varchar(50) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '支付方式',

`phone\_code` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '手机取票码',

PRIMARY KEY (`order\_id`) USING BTREE,

KEY `t\_order\_ibfk\_1` (`user\_id`) USING BTREE,

KEY `t\_order\_ibfk\_2` (`schedule\_id`) USING BTREE,

CONSTRAINT `t\_order\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `t\_user` (`user\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `t\_order\_ibfk\_2` FOREIGN KEY (`schedule\_id`) REFERENCES `t\_schedule` (`schedule\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB AUTO\_INCREMENT=102 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci ROW\_FORMAT=DYNAMIC;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `t\_schedule`

--

DROP TABLE IF EXISTS `t\_schedule`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!50503 SET character\_set\_client = utf8mb4 \*/;

CREATE TABLE `t\_schedule` (

`schedule\_id` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '排片id',

`movie\_id` int unsigned DEFAULT NULL COMMENT '电影id',

`cinema\_id` int unsigned DEFAULT NULL COMMENT '影院id',

`hall\_name` varchar(50) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '影厅名字',

`show\_date` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '放映日期',

`show\_time` varchar(10) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '放映时间',

`price` decimal(10,2) DEFAULT NULL COMMENT '电影售价',

`seat\_info` text CHARACTER SET utf8 COLLATE utf8\_unicode\_ci COMMENT '座位信息',

PRIMARY KEY (`schedule\_id`) USING BTREE,

KEY `hall\_id` (`hall\_name`) USING BTREE,

KEY `t\_schedule\_ibfk\_1` (`movie\_id`) USING BTREE,

KEY `t\_schedule\_ibfk\_2` (`cinema\_id`) USING BTREE,

CONSTRAINT `t\_schedule\_ibfk\_1` FOREIGN KEY (`movie\_id`) REFERENCES `t\_movie` (`movie\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `t\_schedule\_ibfk\_2` FOREIGN KEY (`cinema\_id`) REFERENCES `t\_cinema` (`cinema\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB AUTO\_INCREMENT=102 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci ROW\_FORMAT=DYNAMIC;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `t\_user`

--

DROP TABLE IF EXISTS `t\_user`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!50503 SET character\_set\_client = utf8mb4 \*/;

CREATE TABLE `t\_user` (

`user\_id` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '用户id',

`user\_name` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户名',

`avatar` varchar(100) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户头像',

`password` varchar(100) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户密码',

`phone` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户手机号码',

`sex` varchar(10) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户性别',

`birthday` varchar(20) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户出生年月日',

`sign` varchar(100) CHARACTER SET utf8 COLLATE utf8\_unicode\_ci DEFAULT NULL COMMENT '用户签名',

PRIMARY KEY (`user\_id`) USING BTREE

) ENGINE=InnoDB AUTO\_INCREMENT=55 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci ROW\_FORMAT=DYNAMIC;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `t\_wishmovie`

--

DROP TABLE IF EXISTS `t\_wishmovie`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!50503 SET character\_set\_client = utf8mb4 \*/;

CREATE TABLE `t\_wishmovie` (

`wishMovie\_id` int unsigned NOT NULL AUTO\_INCREMENT COMMENT '想看电影id',

`user\_id` int unsigned DEFAULT NULL COMMENT '用户id',

`movie\_id` int unsigned DEFAULT NULL COMMENT '电影id',

PRIMARY KEY (`wishMovie\_id`) USING BTREE,

KEY `t\_wishmovie\_ibfk\_1` (`user\_id`) USING BTREE,

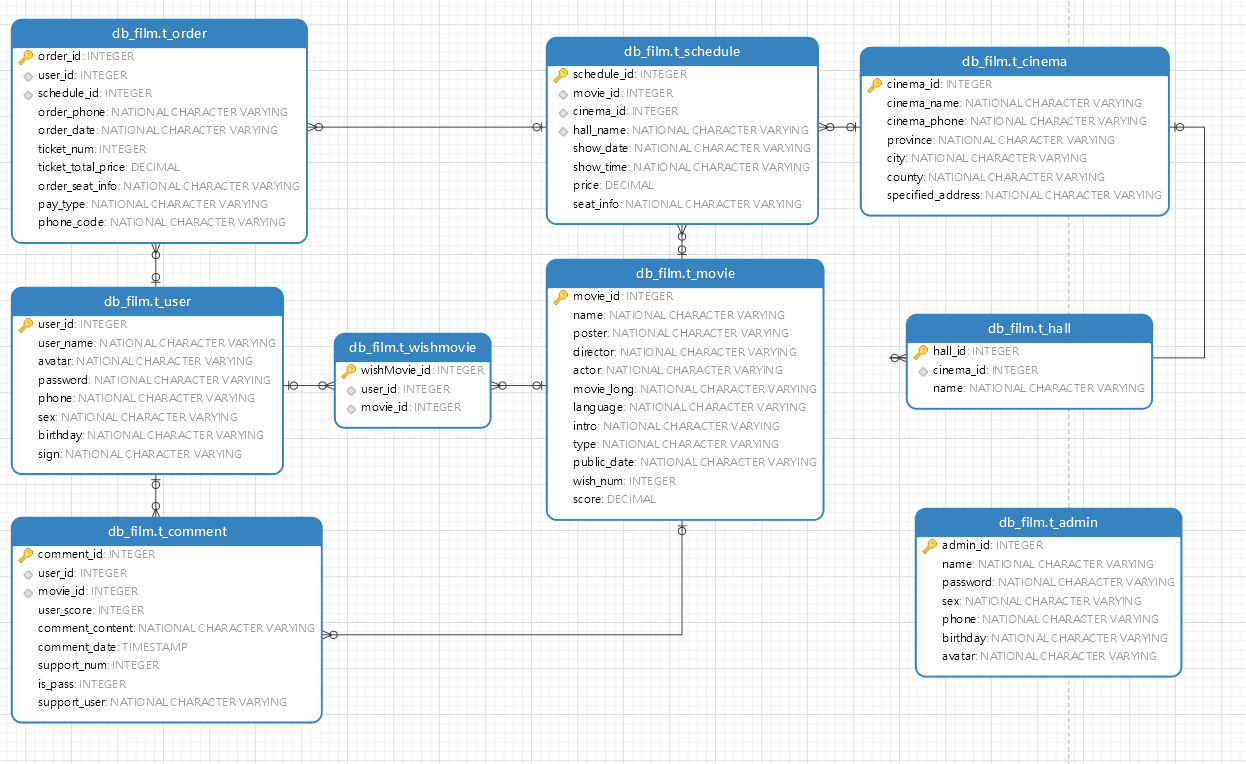
KEY `t\_wishmovie\_ibfk\_2` (`movie\_id`) USING BTREE,

CONSTRAINT `t\_wishmovie\_ibfk\_1` FOREIGN KEY (`user\_id`) REFERENCES `t\_user` (`user\_id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `t\_wishmovie\_ibfk\_2` FOREIGN KEY (`movie\_id`) REFERENCES `t\_movie` (`movie\_id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB AUTO\_INCREMENT=119 DEFAULT CHARSET=utf8 COLLATE=utf8\_unicode\_ci ROW\_FORMAT=DYNAMIC;

-- Dump completed on 2021-05-26 19:37:39



# 5.程序设计说明

## 5.1程序描述

提供了一个音乐门户网站，用户可以登录后选择歌单，播放歌曲，浏览专辑信息，歌手信息，以及其他用户资料等活动在网站上进行活动。

1. 调用网易云接口模块代码

//用来存储音乐信息的对象数组

var musicInfos = [];

//对象属性如下

//

// {

// songListName: ""; 仅需要在第一个对象中存储歌单名称

// songID:"";

// songName: "",

// artist: "",

// albumPic: "",

// totalTime: "",

// mp3Url: "", 该url的格式是http://music.163.com/song/media/outer/url?id=3986241

// mp3Url2: "", 该url的格式是"https://m7.music.126.net/20181016104636/965ff036084dc30ec291460d1f4f85e3/ymusic/8827/447f/9ef4/3f399b1fd6d919555b55691e6632366d.mp3"

// 上面两个是不同的链接格式，第一个链接学校的网络设置了禁止访问网易云音乐163网站，所以无法播放，但一般的网络都可以访问。

// 第二个链接则是通过网易云api接口获取到的，学校的网络没有禁止，可以访问，当然一般的网络也可以访问。

// 两个链接都同时添加到audio下的source下，一个链接失效了，另一个链接可以使用。

// connectTimes: "",

// index: "" //存储是第几首歌

// };

//存储默认歌单

var defaultSongList = [];

//存储用户创建的歌单列表

var userSongList = [];

//对象属性如下

//

//{

// userName: ""; 仅需要在第一个对象中存储用户名称

// listID: "";

// listName: "";

//}

//存储用户网络搜索的歌曲

//对象属性比musicInfos多了一个isSearch:true,

//用于跟musicInfos区别

//其他属性跟musicInfos一样

var songSearchResults = [];

//存储本地歌曲

//对象属性比musicInfos多了一个isLocal:true,

//用于跟musicInfos区别

//其他属性跟musicInfos一样

var localMusicFiles = [];

//指向正在播放的音乐列表，

//比如musicInfos、songSearchResults、localMusicFiles其中一个

var currentPlayList;

//当前正在播放的音乐列表标识，

//0为musicInfos，1位songSearchResults,2位localMusicFiles

var currentPlayListIndex = 0;

//正在播放的歌曲

//属性跟musicInfos或者songSearchResults一样

var currentPlaySong;

//用来存储解析出来的歌词,格式如下[[time,lyricString],[time,lyricString],.....]

var lyricResult = [];

//歌词中歌手的信息，用于全屏歌词显示

var lyricArtistMessage = [];

//保存$("#lyricDiv ul li")的叠加高度值

var lyricHeight = [];

//网易云获取歌曲下载地址接口

//接口详细信息参考CeuiLiSA的知乎文章链接：https://zhuanlan.zhihu.com/p/30246788

//其中type=song指定该接口时获取歌曲信息

//id是歌曲的id，可以在网易云分享歌曲链接中看到

//br为歌曲波特率，可以为128k，或者320k

var songUrl = "https://api.imjad.cn/cloudmusic/?type=song&id=862101001&br=128000";

//其中id为某个歌单的id，比如“我的喜欢”歌单，同样在分享链接中可以找到

var playlistUrl = "https://api.imjad.cn/cloudmusic/?type=playlist&id=";

//var playListID = "379133594"; //欧美经典|总有一首歌直击你心

//var playListID = "75182300"; //评论过万的英语歌与潜力股

var playListID = "2484734225"; //万次评论只为这个旋律

//获取用户名歌单的接口

var songList = "https://api.imjad.cn/cloudmusic/?type=search&search\_type=1000&s=";

//默认用户名是‘默认歌单’，我创建的一个网易云用户

var songListName = "DefaultSongList";

var isGetList = false;

//根据歌名搜索歌曲的接口

var songSearch = "https://api.imjad.cn/cloudmusic/?type=search&search\_type=1&s=";

var songSearchName = "一百万个可能";

//根据歌曲ID获取歌词的接口

var lyricApi = "https://api.imjad.cn/cloudmusic/?type=lyric&id=";

//进度条的颜色

var progressColors = ["#00ffff", "#ffff00", "#efff66", "#66ff71"];

//获取歌单的所有歌曲信息

function initSongs(url) {

var connectUrl = "";

if(url) {

connectUrl = url;

} else {

connectUrl = playlistUrl + playListID;

}

//通过ajax同步请求数据，如果网络异常则给出提示

$.ajax({

url: connectUrl,

type: "post",

dataType: 'JSON',

async: false,

cache: true,

success: getSongId,

error: function(xhr, status, error) {

//alert("抱歉，服务器故障了。\n你可以播放默认歌曲和本地音乐。");

//如果网络问题或者服务器挂了，则加载预先下载好的歌单信息

getSongId(playListTest);

}

});

}

//根据网易云接口返回的信息初始化歌曲列表

function getSongId(data) {

var tracks = data.playlist.tracks;

//刷新歌曲之前，先把之前的歌曲置空

musicInfos = [];

for(var i = 0; i < tracks.length; i++) {

var musicInfo = new Object();

//只在第一个对象中存储歌单名称

if(i == 0) {

musicInfo.songListName = data.playlist.name;

}

musicInfo.songID = tracks[i].id;

musicInfo.songName = tracks[i].name;

//为了简洁，只获取第一个歌手的名

musicInfo.artist = tracks[i].ar[0].name;

musicInfo.albumPic = tracks[i].al.picUrl + '?param=270y270';

musicInfo.totalTime = tracks[i].dt;

musicInfo.mp3Url = "http://music.163.com/song/media/outer/url?id=" + tracks[i].id + ".mp3";

musicInfo.mp3Url2 = ""; //第二个链接先置空，后面利用空闲时间异步获取，保证应用性能

//connetTimes记录当前歌曲从接口获取mp3Url连接的次数，

//超过5次则停止获取，

//避免歌曲ID失效，网易云接口传过来的是空url，造成浪费资源多次获取

musicInfo.connectTimes = 0;

musicInfo.index = i;

musicInfos.push(musicInfo);

}

//初始完成之后主要把赋值给正在播放的列表

currentPlayList = musicInfos;

currentPlayListIndex = 0;

//现在有了新的解决mp3Url的方法，就是获取到歌曲ID后直接

//在该链接后面加上ID的值http://music.163.com/song/media/outer/url?id= + id.mp3，

//这为获取url链接提供了极大的方便，

//既不用担心url链接失效，也不用担心获取url的接口失效或者IP被禁，

//也避免了多次耗费资源循环地更新url链接

//先同步获取第一首歌曲的第二个mp3Url2链接，

//后面才异步获取当前播放歌曲的前10首和后10首。

//由于网易云接口获取的url有时间限制，超过大概半个小时后url链接就会失效，

//所以用户点击上一首或下一首共10次之后或者半个小时之后就会更新链接

getMp3Url(musicInfos, 0, 1, 0, false);

}

//var errorCount = 0;

//initList当前需要获取url链接的数组，为musicInfos或者songSearchResults

function getMp3Url(initList, startindex, endIndex, goalIndex, isAsync, isCached) {

for(var i = startindex; i < endIndex; i++) {

var songUrl = "";

if(goalIndex) {

songUrl = "https://api.imjad.cn/cloudmusic/?type=song&id=" + initList[goalIndex].songID + "&br=128000";

} else {

songUrl = "https://api.imjad.cn/cloudmusic/?type=song&id=" + initList[i].songID + "&br=128000";

}

$.ajax({

url: songUrl,

type: "get",

dataType: 'JSON',

crossDomain: true,

async: isAsync,

cache: isCached ? isCached : isAsync,

success: function(data) {

if(isAsync) {

var times = 0;

while(times < initList.length) {

//为防止异步加载数据顺序错乱，只有songID对应时才添加mp3Url链接

if(initList[startindex].songID != data.data[0].id) {

startindex = (startindex + 1) % initList.length;

times++;

} else {

initList[startindex].mp3Url2 = data.data[0].url;

startindex = (startindex + 1) % initList.length;

break;

}

}

} else {

initList[goalIndex].mp3Url2 = data.data[0].url;

}

},

error: function(xhr, status, error) {

console.log(xhr.status);

//errorCount++;

}

});

}

}

function initDefaultSongList() {

//通过ajax同步请求数据，如果网络异常则给出提示

$.ajax({

url: songList + songListName,

type: "post",

dataType: 'JSON',

async: true,

cache: true,

success: initSongListData,

error: function(xhr, status, error) {

//alert("获取歌曲信息失败，请确认网络是否正常。");

}

});

}

function initSongListData(data) {

if(data.result.playlistCount > 0) {

var playlists = data.result.playlists;

for(var i = 0; i < playlists.length; i++) {

var list = new Object();

if(i == 0) {

list.userName = playlists[i].creator.nickname;

}

list.listID = playlists[i].id;

list.listName = playlists[i].name;

defaultSongList.push(list);

}

//将歌单信息传到songList.js中

setTimeout(function() {

window.frames[1].postMessage(defaultSongList, "/");

}, 3000);

//再把默认用户存入localStorage中，方便用户切换

localStorage.setItem("userName0", "DefaultSongList");

}

}

//当前售票系统状态

var playStatus = {

currentTrackLen: 0,

currentTrackIndex: 0,

currentTime: 0,

currentTotalTime: 0,

playTimes: 0, //点击上一首下一首的次数

//使用下划线前缀表示这是受保护的对象，即protected

\_playStatus: false,

};

function initPlayStatus() {

if(currentPlayList) {

playStatus.currentTrackLen = currentPlayList.length;

} else {

playStatus.currentTrackLen = 0;

}

playStatus.currentTrackIndex = 0;

playStatus.currentTime = 0;

//因为timgTask会不断读取音乐的currentTime，

//当点击本地文件夹音乐列表调用initPlayStatus()时,timgTask还会继续读取，

//导致currentTime > currentTotalTime,误以为当前歌曲已经播放完成，

//从而不断地播放一下首，导致bug

playStatus.currentTotalTime = 10000000;

playStatus.playTimes = 0;

playStatus.\_playStatus = false;

}

var timeOut;

//启动定时任务，加载时间进度条和显示加载动画

function timingTask() {

//因为interval会有累积效应，比如alert一个窗口，用户很久都还没点击，

//这时线程因为alert而堵塞，但interval仍然会计算时间，

//将到时间但还没执行的操作添加到队列中。

//当用户点击后，累积的interval就会一次性按顺序执行，

//此时一个clearInterval便无法把所有的interval停止了。

//因此会造成性能问题。

//因此时间间隔比较小的尽量使用内嵌setTimeOut来代替。

//注意：setTimeOut也会累积

playStatus.currentTime = $('#audio')[0].currentTime;

playerControls.playTime();

if(playStatus.currentTime >= playStatus.currentTotalTime) {

$('.player .controls .next').click();

}

//根据网络状态显示加载动画

//readyState == 0 表示have-nothing，还没开始加载资源

//readyState == 2 表示已经有当前数据，但是还没有下面播放的数据

//readyState == 4 表示已经有足够的数据

//networkState == 2 表示请求网络资源正在加载中

//networkState == 1 表示已经请求到网络资源，可以播放了

if(($("#audio")[0].readyState != 4) &&

$("#audio")[0].networkState != 1) {

if($(".loader:first").css("display") == "none") {

$(".loader:first").css("display", "block");

}

} else {

if($(".loader:first").css("display") == "block") {

$(".loader:first").css("display", "none");

}

}

//如果playStatus为true则循环调用自己，

//否则将自己停止

if(playStatus.playStatus) {

timeOut = setTimeout(timingTask, 300);

} else {

clearTimeout(timeOut);

//如果加载动画还在加载，则停止加载

if($(".loader:first").css("display") == "block") {

$(".loader:first").css("display", "none");

}

//刷新播放按钮状态

$('.player .controls .play i').attr('class', 'icon-' + (playStatus.playStatus ? 'pause' : 'play'));

}

}

//使用definedProperty方法监听playStatus属性，当为true的时候才进行定时任务

Object.defineProperty(playStatus, 'playStatus', {

get: function() {

return this.\_playStatus;

},

set: function(value) {

this.\_playStatus = value;

if(value == true) {

//开启定时任务

timingTask();

} else {

//按了停止键，则停止定时任务

clearTimeout(timeOut);

//如果加载动画还在加载，则停止加载

if($(".loader:first").css("display") == "block") {

$(".loader:first").css("display", "none");

}

//刷新播放按钮状态

$('.player .controls .play i').attr('class', 'icon-' + (playStatus.playStatus ? 'pause' : 'play'));

}

}

});

var playerControls;

//重载刷新歌曲的次数，超过3次则停止刷新

var loadTimes = 0;

//歌曲链接失效的次数，如果超过3次则说明大部分的歌曲url已经失效，

//缓存已经失效，需要重新获取，刷新缓存

var invalidTimes = 0;

//初始化所有资源

var init;

$().ready(function() {

//判断是否ios系统

if(/ipad|iphone|mac/i.test(navigator.userAgent)) {

$(".musicList").addClass('ios-scroll');

$(".musicList iframe").addClass('frame');

document.getElementById("view").setAttribute("content",

"width=device-width,initial-scale=0.7, minimum-scale=0.7, maximum-scale=0.7, user-scalable=no");

}else{

$(".musicList iframe").css("height","100%");

}

drawStars();

//根据localStorage初始化dataList元素，方便用户切换用户名

initDataList();

//初始化所有音乐信息

initSongs();

//初始化默认歌单

initDefaultSongList();

//售票系统控制方法

playerControls = {

//歌曲基本信息

trackInfo: function(args) {

//保存现在正在播放的歌曲

currentPlaySong = args;

//先添加audio元素

$('#audio').remove();

$('.player').append('<audio id="audio" preload="auto"><source id="source1" src=""><source id="source2" src=""></source></audio>');

$('#source1').attr('src', args.mp3Url2);

$('#source2').attr('src', args.mp3Url);

//加载audio音频

$("#audio")[0].load();

if(args.isLocal) {

//如果是本地文件夹的歌曲，则在audio音频元素加载完成时进行读取音频时长

$("#audio")[0].onloadedmetadata = function() {

//音频时长单位为秒

var totalTime = $("#audio")[0].duration;

$('.player .time .total').text(timeConvert(totalTime));

playStatus.currentTotalTime = totalTime - 1;

args.totalTime = totalTime;

}

} else {

//网易云接口返回的音频时长单位为毫秒

//显示这首歌的时长，

$('.player .time .total').text(timeConvert(args.totalTime / 1000));

//减1是为了避免计算误差，无法自动下一首

playStatus.currentTotalTime = Math.floor(args.totalTime / 1000) - 1;

}

//根据歌名长度设置字体大小，避免歌名太长超出边框

$('.player .trackInfo .name p').text(args.songName);

if(args.songName.length >= 40) {

$('.player .trackInfo .name').css("font-size", "18px");

} else if(args.songName.length > 30) {

$('.player .trackInfo .name').css("font-size", "22px");

} else {

$('.player .trackInfo .name').css("font-size", "26px");

}

//歌手名称

$('.player .trackInfo .artist').text(args.artist);

//歌曲图片

if(args.isLocal) {

$('.player .albumPic').css('background', 'url(images/artist.jpg)');

} else {

$('.player .albumPic').css('background', 'url(' + args.albumPic + ')');

}

//获取歌词

getLyric();

},

//播放、暂停状态处理

playStatus: function() {

$('.player .controls .play i').attr('class', 'icon-' + (playStatus.playStatus ? 'pause' : 'play'));

if(playStatus.playStatus) {

//networkState = 3则说明音乐mp3URl链接失效，找不到资源，需要重新获取歌曲url链接

//networkState = 0则说明该音乐的mp3Url为空

if($("#audio")[0].networkState != 3 && $("#audio")[0].networkState != 0) {

console.log(currentPlaySong.songName + " , " +

currentPlaySong.songID + " , " +

',' + currentPlaySong.mp3Url2);

console.log("readyState:" + $("#audio")[0].readyState);

console.log("networkState:" + $("#audio")[0].networkState);

console.log("error:" + $("#audio")[0].error);

$("#audio")[0].play();

//如果5秒后还是没有任何资源(reayState=0)而且网络仍然在加载(networkState=2)

//则重新加载刷新歌曲，重新加载次数不超过3次，仍然失败则是网络问题

setTimeout(function() {

console.log("readyState:" + $("#audio")[0].readyState);

console.log("networkState:" + $("#audio")[0].networkState);

console.log("error:" + $("#audio")[0].error);

if($("#audio")[0].readyState == 0 &&

$("#audio")[0].networkState == 2) {

//重新刷新的歌曲信息

playerControls.trackInfo(currentPlaySong);

$("#audio")[0].load();

loadTimes++;

//playStatus.playStatus = false;

//先停顿0.1秒，让程序先加载音乐资源,否则networkState属性会为3，意为找不到资源

if(loadTimes <= 4) {

setTimeout(function() {

playerControls.playStatus();

}, 300);

} else {

loadTimes = 0;

playStatus.playStatus = false;

alert("加载歌曲失败，请检查网络。");

}

} else if($("#audio")[0].networkState == 3) {

//如果5秒后该音乐的网络状态为3，则说明请求资源失效

loadTimes = 0;

invalidTimes++; //歌曲失效次数增加，超过3次刷新url链接

playerControls.trackInfo(currentPlaySong);

$("#audio")[0].load();

//调用播放方法去刷新链接

playerControls.playStatus();

}

}, 5000);

//当点击下一首上一首的次数超过8次，对url进行更新

//因为指定了ajax使用cached，已经加载url的不用重新连接网易云接口

if(playStatus.playTimes >= 8 && currentPlayListIndex != 2) {

init20Songs();

playStatus.playTimes = 0;

}

//显示第一句歌词

if(lyricResult.length <= 0) {

$("#lyric").text("纯音乐，请欣赏。");

} else {

$("#lyric").text(lyricResult[0][1]);

}

} else {

//先判断是否已经获取了3次

if(currentPlaySong.connectTimes < 3) {

//同步获取当前失效歌曲的url链接

if(currentPlayListIndex == 1) {

//如果是网络搜索结果则更新songSearchResults

getMp3Url(songSearchResults, 0, 1, playStatus.currentTrackIndex, false);

songSearchResults[currentPlaySong.index].connectTimes++;

playStatus.playStatus = false;

//重新加载失效的歌曲信息

playerControls.trackInfo(songSearchResults[currentPlaySong.index]);

} else if(currentPlayListIndex == 0) {

//否则更新musicInfos

getMp3Url(musicInfos, 0, 1, playStatus.currentTrackIndex, false);

musicInfos[playStatus.currentTrackIndex].connectTimes++;

playStatus.playStatus = false;

//重新加载失效的歌曲信息

playerControls.trackInfo(musicInfos[playStatus.currentTrackIndex]);

}

$("#audio")[0].load();

//先停顿0.1秒，让程序先加载音乐资源,否则networkState属性会为3，意为找不到资源

setTimeout(function() {

playerControls.playStatus();

}, 300);

alert("歌曲链接失效，请重新点击播放键。");

} else {

playStatus.playStatus = false;

alert("抱歉，该歌曲获取失败，请换下一首歌。");

}

//失效次数超过3次则说明缓存的url链接几乎都失效了，需要重新获取，刷新缓存

invalidTimes++;

if(invalidTimes >= 3 && currentPlayListIndex != 2) {

init20Songs(false, false);

invalidTimes = 0;

}

}

} else {

if($("#audio")[0].played) {

$('#audio')[0].pause();

//恢复歌词字样

if(lyricResult.length <= 0){

if(playStatus.currentTrackIndex == 0){

$("#lyric").text("歌词");

}else{

$("#lyric").text("纯音乐，请欣赏。");

}

}else{

$("#lyric").text("歌词");

}

}

}

},

//当前时间和进度处理

playTime: function() {

$('.player .time .current').text(timeConvert(playStatus.currentTime));

$('.player .progress').css('width', playStatus.currentTime / playStatus.currentTotalTime \* 100 + '%');

}

};

var timeConvert = function(timestamp) {

var minutes = Math.floor(timestamp / 60);

var seconds = Math.floor(timestamp - (minutes \* 60));

if(seconds < 10) {

seconds = '0' + seconds;

}

timestamp = minutes + ':' + seconds;

return timestamp;

};

init = function() {

//初始化播放状态

initPlayStatus();

if(musicInfos.length > 0) {

playerControls.trackInfo(currentPlayList[playStatus.currentTrackIndex]);

playerControls.playStatus();

}

$('.player .controls .play').unbind("click").click(function() {

playStatus.playStatus = !playStatus.playStatus;

playerControls.playStatus();

});

$('.player .controls .previous').unbind("click").click(function() {

if(playStatus.currentTrackIndex - 1 < 0) {

alert('已经没有上一首了');

playStatus.currentTrackIndex = 0;

} else {

playStatus.currentTrackIndex--;

}

if(currentPlayListIndex != 2) {

playerControls.trackInfo(currentPlayList[playStatus.currentTrackIndex]);

//先停顿0.1秒，让程序先加载音乐资源,否则networkState属性会为3，意为找不到资源

setTimeout(function() {

playerControls.playStatus();

}, 300);

$(".player .progress").css("background-image", "-webkit-linear-gradient(top, rgba(255, 255, 255, 0), " +

progressColors[playStatus.currentTrackIndex % 5] + ")");

//点击下一首次数加1

playStatus.playTimes++;

} else {

playLocalMusic(playStatus.currentTrackIndex);

}

});

$('.player .controls .next').unbind("click").click(function() {

if(playStatus.currentTrackIndex + 1 >= playStatus.currentTrackLen) {

alert('已经没有下一首了');

playStatus.currentTrackIndex = 0;

} else {

playStatus.currentTrackIndex++;

}

if(currentPlayListIndex != 2) {

playerControls.trackInfo(currentPlayList[playStatus.currentTrackIndex]);

//先停顿0.1秒，让程序先加载音乐资源,否则networkState属性会为3，意为找不到资源

setTimeout(function() {

playerControls.playStatus();

}, 300);

$(".player .progress").css("background-image", "-webkit-linear-gradient(top, rgba(255, 255, 255, 0), " +

progressColors[playStatus.currentTrackIndex % 5] + ")");

//点击上一首次数加1

playStatus.playTimes++;

} else {

playLocalMusic(playStatus.currentTrackIndex);

}

});

};

init();

//延迟5秒初始化20首歌链接，避免一开始加载时让用户觉得卡

setTimeout(init20Songs, 5000);

//然后再定时每30分钟更新一次url链接

setInterval(init20Songs, 30 \* 60 \* 1000);

});

var frontLength = 10;

var behindLength = 10;

var startIndex = 0;

var endIndex = 0;

var time = null;

//异步加载当前歌曲的前10首后10首歌曲的链接，

//因为如果一次性循环访问网易云接口太多次会被503，

//或者会发生异域访问安全性问题，抛出No 'Access-Control-Allow-Origin' header 异常

//所以分两次进行异步加载

function init20Songs(secondFlag, isCached) {

var currentIndex = playStatus.currentTrackIndex;

var initList;

switch(currentPlayListIndex) {

//如果是初始的音乐歌曲列表

case 0:

initList = musicInfos;

break;

//如果是网络搜索结果列表

case 1:

initList = songSearchResults;

break;

}

if(!secondFlag) {

startIndex = (currentIndex - 10) >= 0 ? currentIndex - 10 : 0;

getMp3Url(initList, startIndex, currentIndex, null, true, isCached);

} else {

endIndex = (currentIndex + 10) >= musicInfos.length ? musicInfos.length : currentIndex + 10;

getMp3Url(initList, currentIndex, endIndex, null, true, isCached);

}

time = setTimeout(function() {

init20Songs(true);

}, 5000);

if(time && secondFlag) {

clearTimeout(time);

setTimeout(checkMusicInfo, 5000);

}

}

//检查20首歌曲的链接是否全都加载成功

function checkMusicInfo() {

var urlNullCount = 0;

var nullIndexs = [];

for(var i = startIndex; i < endIndex; i++) {

if(musicInfos[i].mp3Url2.length <= 0) {

urlNullCount++;

nullIndexs.push(i);

// console.log("numIndex:"+i+",nullID: "+musicInfos[i].songID+","+musicInfos[i].songName);

}

}

//console.log("urlNullCount:"+urlNullCount);

//console.log("errorCount:"+errorCount);

if(nullIndexs.length > 0) {

getNullIDUrl(nullIndexs);

}

}

//对空的url重新获取，避免死循环，最后只能获取3次

var currentTime = 0;

function getNullIDUrl(nullIndexs) {

currentTime++;

//判断是否进行了连接获取数据，为true才循环进行检查音乐信息

var hasConnected = false;

for(var i = 0; i < nullIndexs.length; i++) {

//先判断该歌曲是否获取超过了5次

var goalIndex = nullIndexs[i];

if(musicInfos[goalIndex].connectTimes < 5) {

//是以同步加载的方式获取

getMp3Url(musicInfos, 0, 1, goalIndex, false);

musicInfos[goalIndex].connectTimes++;

hasConnected = true;

}

}

if(currentTime < 3 && hasConnected) {

setTimeout(checkMusicInfo, 5000);

} else {

//重置为0，下一轮循环也是3次

currentTime = 0;

}

}

function showList() {

//通过iframe的postMessage方法将歌曲的信息musicInfos传送到music.js中进行显示

//"/"表示传送信息的页面和接收的页面是同源的意思

window.frames[0].postMessage(musicInfos, "/");

listButtonShow(true);

$("#songList").css("display", "none");

$("#searchMusicList").css("display", "none");

}

//为了避免代码重复而整合的函数,

//当切换到歌曲列表页面，则把其他的元素隐藏，否则相反

function listButtonShow(isListShow) {

var none = "";

var block = "";

if(isListShow) {

none = "none";

block = "block";

} else {

block = "none";

none = "block";

}

$(".albumPic").css("display", none);

$("#musicList").css("display", block);

$("#musicBtn").css("display", none);

$("#back").css("display", block);

$("#switch").css("display", none);

$("#local").css("display", block);

$("#title").css("display", none);

$("#search").css("display", block);

}

function goBack() {

if($("#searchMusicList").css("display") == "block") {

if(currentPlayListIndex == 0) {

$(".musicList").css("display", "none");

$("#musicList").css("display", "block");

} else if(currentPlayListIndex == 2) {

$(".musicList").css("display", "none");

$("#localMusicList").css("display", "block");

}

} else if($("#netSearchList").css("display") == "block") {

$(".musicList").css("display", "none");

$("#musicList").css("display", "block");

} else {

listButtonShow(false);

$("#songList").css("display", "none");

$("#localMusicList").css("display", "none");

$("#lyricDiv").css("display", "none");

}

}

//切换歌单

function getSongList(userName) {

if(userName.length > 0) {

//显示加载动画

$(".loader2").css("display", "block");

//使用正则表达式和字符串的replace方法去掉前后的空格

userName = userName.replace(/^\s+|\s+$/g, "");

//console.log(userName.length);

//如果输入的用户名为字符串，则说明是搜索该用户创建的所有歌单的

//如果是数字，则说明输入的是歌单ID

if(isNaN(userName)) {

//通过ajax同步请求歌单数据，如果网络异常则给出提示

$.ajax({

url: songList + userName,

type: "post",

dataType: 'JSON',

async: false,

cache: true,

success: showSongList,

error: function(xhr, status, error) {

$(".loader2").css("display", "none");

alert("抱歉，服务器故障了，获取用户歌单失败。");

}

});

} else {

isGetList = true;

showMusicList(userName, true);

}

}

}

//显示歌单列表

function showSongList(songList) {

if(songList.result.playlistCount > 0) {

isGetList = true;

var playlists = songList.result.playlists;

for(var i = 0; i < playlists.length; i++) {

var list = new Object();

if(i == 0) {

list.userName = playlists[i].creator.nickname;

}

list.listID = playlists[i].id;

list.listName = playlists[i].name;

userSongList.push(list);

}

//加载完毕，关闭加载动画

$(".loader2").css("display", "none");

//将歌单信息传到songList.js中

window.frames[1].postMessage(userSongList, "/");

listButtonShow(true);

$(".musicList").css("display", "none");

$("#songList").css("display", "block");

$("#title").css("display", "block");

$("#search").css("display", "none");

$("#local").css("display", "none");

} else {

isGetList = false;

$(".loader2").css("display", "none");

alert("抱歉，服务器故障了，获取用户歌单失败。");

}

}

//自定义的弹出div窗口

function myPrompt(placeHolderName) {

if(isGetList == false) {

//jquery获取的元素是jquery对象,即使没有该元素也不会返回null,

//所以以length是否为0判断div弹窗元素是否已经存在

if($("#msg").length != 0) {

$("#msg").fadeIn(200);

} else {

$("body").append('<div id="msg">' +

'<div id="msg\_top">请输入您的网易云用户名：' +

'<div class="cross"><span class="icon-cross"></span></div>' +

'<br /> <span style="font-size:14px">（或者网易云歌单ID）</span>' +

'</div>' +

'<div id="msg\_cont"><input id="userName" class="nameInput" type="text" list="nameLists" placeholder="' + placeHolderName + '" /></div>' +

'<div class="msg\_confirm" id="confirm"><span class="icon-checkmark"></span></div>' +

'</div>');

}

//如果使用jquery绑定事件的函数会被重复执行,则绑定的次数会叠加,

//需要先解绑再添加,否则会重复执行多次

$("#userName").unbind("keypress").keypress(function(e) {

if(e.keyCode == 13) {

$("#confirm").click();

}

})

$(".icon-cross").unbind("click").click(function() {

$("#msg").fadeOut(200);

});

$("#confirm").unbind("click").click(function() {

if($("#userName").val().length > 0) {

$("#msg").css("display", "none");

getSongList($("#userName").val());

//将用户名永久保存到浏览器中的localStorage中,方便后面输入

if(nameLists.search($("#userName").val()) == -1) {

var key = "userName" + (localStorage.length + 1);

localStorage.setItem(key, $("#userName").val());

}

} else {

alert("请输入您的网易云用户名");

}

});

} else {

listButtonShow(true);

$(".musicList").css("display", "none");

$("#songList").css("display", "block");

$("#title").css("display", "block");

$("#search").css("display", "none");

$("#local").css("display", "none");

}

}

//点击歌单或者输入歌单ID显示歌曲列表

function showMusicList(listID, isUserInput) {

$(".loader2").css("display", "block");

var url = playlistUrl + listID;

initSongs(url);

//init();

showList();

setTimeout(init20Songs, 5000);

$(".loader2").css("display", "none");

//如果是用户直接输入歌单ID的,还需要把歌单的信息传送到songList页面中

if(isUserInput) {

//将歌单信息传到songList.js中

var inputSongList = [];

var songList = new Object();

songList.userName = "Admin";

songList.listID = listID;

songList.listName = musicInfos[0].songListName;

inputSongList.push(songList);

window.frames[1].postMessage(inputSongList, "/");

}

}

//添加消息监听,监听iframe页面传送过来的信息,即用户点的是第几首歌曲

window.addEventListener("message", function(e) {

//通过信息的来源判断是哪个frame传过来的

if(e.source == window.frames[0]) {

//alert("这是从音乐列表传过来的信息");

if(e.data == "switchSongList") {

listButtonShow(true);

$(".musicList").css("display", "none");

$("#songList").css("display", "block");

$("#title").css("display", "block");

$("#search").css("display", "none");

} else {

//将当前播放列表指向musicInfos

currentPlayList = musicInfos;

//当前播放列表的表示符为0

currentPlayListIndex = 0;

playCurrentMusic(e.data)

}

} else if(e.source == window.frames[1]) {

//alert("这是从歌单列表传过来的信息");

if(e.data == "switchUser") {

isGetList = false;

userSongList = [];

myPrompt("UUZUUZ");

} else {

var listID = e.data;

showMusicList(listID);

}

} else if(e.source == window.frames[2]) {

//alert("这是从本地搜索列表传过来的信息");

if(currentPlayListIndex != 2) {

playCurrentMusic(e.data);

} else {

playLocalMusic(e.data);

}

} else if(e.source == window.frames[3]) {

//alert("这是从网络搜索列表传过来的信息");

//将当前播放列表指向songSearchResults

currentPlayList = songSearchResults;

//当前播放列表的标识符为1

currentPlayListIndex = 1;

playCurrentMusic(e.data);

} else if(e.source == window.frames[4]) {

//alert("这是从本地文件夹音乐列表传过来的信息")

if(e.data == "switchLocalList") {

getLocalFiles();

} else {

currentPlayList = localMusicFiles;

currentPlayListIndex = 2;

playLocalMusic(e.data);

}

}

}, false);

//本地电影售票还需要额外处理,将本地音乐文件转化成可以播放的二进制音频

function playLocalMusic(currentTrackIndex) {

//初始化一下播放状态,指向当前播放列表

initPlayStatus();

if(localMusicFiles[currentTrackIndex].mp3Url.length <= 0) {

var localMusic = localMusicFiles[currentTrackIndex];

var musicFiles = localMusic.musicData;

var fileReader = new FileReader();

//监听本地音乐读取事件

fileReader.onload = function(e) {

localMusic.mp3Url = e.target.result;

//如果读取完成则进行播放

playStatus.currentTrackIndex = currentTrackIndex;

playerControls.trackInfo(currentPlayList[playStatus.currentTrackIndex]);

playStatus.playStatus = true;

setTimeout(function() {

playerControls.playStatus();

}, 1000);

}

//将本地音乐文件以url的二进制音频形式进行读取

fileReader.readAsDataURL(musicFiles);

} else {

//如果已经保存了mp3Url信息,则直接播放

playStatus.currentTrackIndex = currentTrackIndex;

playerControls.trackInfo(currentPlayList[playStatus.currentTrackIndex]);

playStatus.playStatus = true;

setTimeout(function() {

playerControls.playStatus();

}, 500);

}

}

function playCurrentMusic(currentTrackIndex) {

//初始化一下播放状态,指向当前播放列表

initPlayStatus();

playStatus.currentTrackIndex = currentTrackIndex;

playerControls.trackInfo(currentPlayList[playStatus.currentTrackIndex]);

playStatus.playStatus = true;

//如果点击的音乐url链接为空,则进行获取

if(currentPlayList[playStatus.currentTrackIndex].mp3Url2.length <= 0) {

getMp3Url(currentPlayList, 0, 1, playStatus.currentTrackIndex, false, false);

//如果前后的音乐url链接都是为空则获取前后10音乐的链接

if(currentPlayList[(playStatus.currentTrackIndex + 1) % currentPlayList.length].mp3Url2.length <= 0 ||

currentPlayList[(playStatus.currentTrackIndex - 1) % currentPlayList.length].mp3Url2.length <= 0) {

setTimeout(init20Songs, 5000);

}

}

setTimeout(function() {

playerControls.playStatus();

}, 300);

}

//快捷键

$("html").unbind("keydown").keydown(function(e) {

//播放快捷键 空格

//事件会冒泡,只有nodeName为body或者html才出发,否则input框按下空格键也会触发

if((e.target.nodeName == "BODY" || e.target.nodeName == "HTML") && e.keyCode == 32) {

$('.player .controls .play').click();

}

//下一首快捷键 Ctrl + Alt + ->

if(e.ctrlKey && e.altKey && e.keyCode == 39) {

$('.player .controls .next').click();

}

//上一首快捷键,Ctrl + Alt + <-

if(e.ctrlKey && e.altKey && e.keyCode == 37) {

$('.player .controls .previous').click();

}

});

//保存所有用户名,如果用户名已存在则无需添加

var nameLists = "";

//从localStorage获取数据初始化dataList

function initDataList() {

for(var i = 0; i < localStorage.length; i++) {

var key = localStorage.key(i);

if(key.search("userName") != -1) {

var value = localStorage.getItem(key);

//只对还没有添加的用户名进行添加

if(nameLists.search(value) == -1) {

$("#nameLists").append('<option label="' + value + '" value="' + value + '"></option>');

nameLists += value;

}

}

}

}

//监听localStorage是否增加了新用户名数据

//需要注意的是同一个页面修改本地内存,也在用一个页面监控storage事件,这是没有效果的

//"storage"事件需要同源的两个不同的页面才可以触发

//为了让用一个页面可以监听,需要重写localStorage方法,抛出自定义事件

//window.addEventListener("storage", function(e) {

// $("#nameLists").append('<option label="' + e.key + '" value="' + e.key + '"></option>');

//});

(function() {

//因为要更改setItem方法,所以先把原方法保存

var originalSetItem = localStorage.setItem;

//更改setItem方法

localStorage.setItem = function(key, value) {

//生成一个自定义事件

var setItemEvent = new Event("setItemEvent");

setItemEvent.value = value;

//抛出触发事件

window.dispatchEvent(setItemEvent);

//最后再执行原来方法的操作

originalSetItem.apply(this, arguments);

}

//这样就实现了当调用localStorage.setItem()方法时,把用户名添加到datalist元素中

window.addEventListener("setItemEvent", function(e) {

if(nameLists.search(e.value) == -1) {

$("#nameLists").append('<option label="' + e.value + '" value="' + e.value + '"></option>');

nameLists += e.value;

}

})

})();

//保存搜索的歌曲名,如果歌曲名已存在则无需添加

var songNameLists = "";

//搜索歌曲

function promptSearch() {

if($("#searchDiv").length != 0) {

$("#searchDiv").fadeIn(200);

} else {

$("body").append('<div id="searchDiv">' +

'<div class="search\_top">请输入歌名：' +

'<div class="cross"><span class="icon-cross2"></span></div></div>' +

'<div class="search\_cont"><input id="songName" class="songInput" type="text" list="songLists" /></div>' +

'<div class="search\_confirm" ><button class="localSearch" id="localSearch">本地搜索</button>' +

'<button class="netSearch" id="netSearch">网络搜索</button></div>' +

'</div>');

}

//如果使用jquery绑定事件的函数会被重复执行,则绑定的次数会叠加,

//需要先解绑再添加,否则会重复执行多次

$("#songName").unbind("keypress").keypress(function(e) {

if(e.keyCode == 13) {

$("#localSearch").click();

}

})

$(".icon-cross2").unbind("click").click(function() {

$("#searchDiv").fadeOut(200);

});

$("#localSearch").unbind("click").click(function() {

if($("#songName").val().length > 0) {

$("#searchDiv").fadeOut(200);

localSearch($("#songName").val());

//将歌曲保存到浏览器中的sessionStorage中,方便后面输入

if(songNameLists.search($("#songName").val()) == -1) {

// var key = "userName" + localStorage.length;

// localStorage.setItem(key, $("#userName").val());

}

} else {

alert("请输入歌名");

}

});

$("#netSearch").unbind("click").click(function() {

if($("#songName").val().length > 0) {

$("#searchDiv").fadeOut(200);

netSearch($("#songName").val());

//将歌曲保存到浏览器中的sessionStorage中,方便后面输入

if(songNameLists.search($("#userName").val()) == -1) {

// var key = "userName" + localStorage.length;

// localStorage.setItem(key, $("#userName").val());

}

} else {

alert("请输入歌名");

}

});

}

function localSearch(songName) {

/\*快速查找JSON数组的属性值,只支持全匹配

var musicInfo = currentPlayList.find(function(musicInfo){

//musicInfo是musicInfos的一个元素

return musicInfo.songName == songName;

});

if(musicInfo != null){

console.log("找到歌曲:"+musicInfo.songID+","+musicInfo.songName);

}

\*/

var results = [];

for(var i = 0; i < currentPlayList.length; i++) {

songName = songName.toLowerCase().trim();

musicInfoName = currentPlayList[i].songName.toLowerCase().trim();

if(musicInfoName.indexOf(songName) >= 0) {

results.push(currentPlayList[i]);

}

}

if(results.length > 0) {

//console.log("找到歌曲:"+results[0].index+","+results[0].songID+","+results[0].songName);

//把搜索结果传送到搜索页面

window.frames[2].postMessage(results, "/");

$("#searchDiv").fadeOut(200);

$(".musicList").css("display", "none");

$("#netSearchList").css("display", "none");

$("#searchMusicList").css("display", "block");

} else {

alert("找不到本地歌曲");

$("#searchDiv").fadeIn(0);

}

}

function netSearch(songName) {

var searchUrl = songSearch + songName;

$.ajax({

url: searchUrl,

type: 'get',

dataType: 'json',

async: false,

cache: true,

success: showSearchHtml,

error: function(xhr, status, error) {

alert("抱歉,搜索失败,服务器出故障了.")

}

})

}

function showSearchHtml(results) {

var songs = results.result.songs;

//把之前的搜索结果置空

songSearchResults = [];

for(var i = 0; i < songs.length; i++) {

var song = new Object();

song.songID = songs[i].id;

song.songName = songs[i].name;

//为了简洁，只获取第一个歌手的名

song.artist = songs[i].ar[0].name;

song.albumPic = songs[i].al.picUrl + '?param=270y270';

song.totalTime = songs[i].dt;

song.mp3Url = "http://music.163.com/song/media/outer/url?id=" + songs[i].id + ".mp3";

song.mp3Url2 = "";

//connetTimes记录当前歌曲从接口获取mp3Url连接的次数，

//超过5次则停止获取，

//避免歌曲ID失效，网易云接口传过来的是空url，造成浪费资源多次获取

song.connectTimes = 0;

song.playTimes = 0;

song.index = i;

song.isSearch = true;

songSearchResults.push(song);

}

//初始化网络搜索到歌曲的链接

setTimeout(function() {

init20Songs(false, true, true);

}, 5000);

//把搜索结果传送到搜索页面

window.frames[3].postMessage(songSearchResults, "/");

$("#searchDiv").fadeOut(200);

$("#musicList").css("display", "none");

$("#searchMusicList").css("display", "none");

$("#netSearchList").css("display", "block");

}

function getLocalFiles() {

//如果本地音乐列表不存在,则进行选择音乐文件

if(localMusicFiles.length <= 0 || $("#localMusicList").css("display") == "block") {

if($("#localMusic").length != 0) {

$("#localMusic").fadeIn(200);

} else {

$("body").append('<div id="localMusic">' +

'<div id="music\_top">请选择您的音乐文件：' +

'<div class="music\_cross"><span class="icon-cross3"></span></div>' +

'<br /> <span style="font-size:14px">（可以一次性选多首音乐）</span>' +

'</div>' +

'<div id="music\_cont"><input id="musicFiles" class="musicInput" type="file" accept="audio/\*" multiple/></div>' +

'<div class="music\_confirm" id="music\_confirm"><span class="icon-checkmark"></span></div>' +

'</div>');

}

$(".icon-cross3").unbind("click").click(function() {

$("#localMusic").fadeOut(200);

});

$("#music\_confirm").unbind("click").click(function() {

var musicFiles = document.getElementById("musicFiles");

//value属性保存的是文件的绝对路径,如果为空则说明没有选到文件

if(musicFiles.value) {

$("#localMusic").fadeOut(200);

var files = musicFiles.files;

var hasNotEmpty = true;

var isMp3 = false;

for(var i = 0; i < files.length; i++) {

//如果选择的文件是音频文件才进行添加

if(/audio\/\w+/g.test(files[i].type)) {

//存在音频文件才把上次选择的本地音乐清空,再进行添加

if(hasNotEmpty) {

localMusicFiles = [];

hasNotEmpty = false;

isMp3 = true;

}

var musicInfo = new Object();

musicInfo.songID = -1;

musicInfo.songName = files[i].name;

musicInfo.artist = "本地歌曲";

musicInfo.albumPic = "";

//本地歌曲时长时间还需要获取数据处理

musicInfo.totalTime = 210000;

musicInfo.mp3Url = "";

musicInfo.mp3Url2 = "";

musicInfo.connectTimes = 0;

musicInfo.index = i;

//先将本地音乐文件的files对象保存,等到用户点击歌曲的时候才进行加载播放

musicInfo.musicData = files[i];

//标志该歌曲是本地歌曲

musicInfo.isLocal = true;

localMusicFiles.push(musicInfo);

}

}

//如果选择的所有文件都不是音频文件则进行提醒

if(!isMp3) {

alert("请选择音频文件");

$("#localMusic").fadeIn(0);

} else {

//将本地音乐数据传送到本地音乐列表页面

window.frames[4].postMessage(localMusicFiles, "/");

$("#musicList").css("display", "none");

$("#songList").css("display", "none");

$("#searchMusicList").css("display", "none");

$("#netSearchList").css("display", "none");

$("#localMusicList").css("display", "block");

}

} else {

alert("请选择音乐文件");

}

});

} else {

//如果本地音乐文件列表已经存在,则直接显示出来

listButtonShow(true);

$(".musicList").css("display", "none");

$("#title").css("display", "none");

$("#search").css("display", "block");

$("#local").css("display", "block");

$("#localMusicList").css("display", "block");

}

}

//调用trackInfo方法加载音乐信息的时候,也进行加载歌词,并进行解析

function getLyric() {

if(currentPlaySong.isLocal) {

//如果是本地音乐,则把歌词隐藏

$(".lyric").css("display", "none");

} else {

//如果是网络音乐,则把歌词显示

$(".lyric").css("display", "block");

//通过ajax异步获取歌词信息，如果网络异常则给出提示

$.ajax({

url: lyricApi + currentPlaySong.songID,

type: "get",

dataType: 'JSON',

async: true,

cache: true,

success: processLyric,

error: function(xhr, status, error) {

alert("抱歉,获取歌词失败，服务器出故障了。");

}

});

}

}

//解析歌词

function processLyric(lyricData) {

//正则表达式,匹配[00:59.40]这是时间格式

var pattern = /\[\d{2}:\d{2}.\d+\]/g;

//如果标志没有歌词或者歌词内容不匹配时间格式,则归为纯音乐

if(lyricData.nolyric == true || !pattern.test(lyricData.lrc.lyric)) {

lyricResult = [];

lyricArtistMessage = [];

$("#lyricDiv").empty();

$("#lyricDiv").append("<ul><li>纯音乐,请欣赏。</li></ul>")

$("#lyric").text("纯音乐，请欣赏。");

return;

}

var lyricInfo = lyricData.lrc.lyric;

var lines = lyricInfo.split("\n");

//用来存储解析出来的歌词,格式如下[[time,lyricString],[time,lyricString],.....]

lyricResult = [];

lyricArtistMessage = [];

//有一些歌词前面时歌手的介绍,没有歌词的,需要把它去掉

//直到匹配到有时间的歌词才会停止循环

var artistPattern = /\[\w+:/g;

//需要注意,正则表达式使用g模式的话,下一次匹配会从lastIndex开始匹配,

//因为上面使用了pattern进行了匹配,有可能lastIndex不为0,所以需要重置为0

pattern.lastIndex = 0;

while(!pattern.test(lines[0])) {

//先把歌手信息保存下来,因为全屏歌词的时候需要显示

var lyricMessage = lines[0].replace(artistPattern, "").slice(0, -1);

lyricMessage.length > 0 ? lyricArtistMessage.push(lyricMessage) : 0;

//去掉第一个元素并返回剩下元素

lines = lines.splice(1);

}

//lines最后一行是空的话把它去除掉

lines[lines.length - 1].length == 0 && lines.pop();

lines.forEach(function(value, index, array) {

//返回匹配的时间

var time = value.match(pattern);

//将时间置换成空格,返回歌词字符串内容

var lyricString = value.replace(pattern, "");

//由于一行歌词会有多个时间, 如[03:33.65][03:35.39],所以需要进一步分离

time.forEach(function(value2, index2, array2) {

//去掉前后的[]

var t = value2.slice(1, -1).split(":");

//用秒数表示当前歌词的时间

var seconds = parseInt(t[0]) \* 60 + parseFloat(t[1]);

//将时间和歌词以数组的形式压进lyriclyricResult中

lyricResult.push([seconds, lyricString]);

});

});

//将结果按照时间排序,保证歌词正确有序输出

lyricResult.sort(function(a, b) {

//如果想要a在b前面,则返回一个负数,否则a想排在b后面,则返回一个正数

//所以想要元素按照升序排序,则返回a与b的差值就行

return a[0] - b[0];

});

addLyric();

//把歌词位置置为初始化

lyricIndex = 1;

showLyric();

}

function addLyric() {

$("#lyricDiv").empty();

var ul = $("<ul></ul>");

//歌手信息

for(var i = 0; i < lyricArtistMessage.length; i++) {

ul.append("<li>" + lyricArtistMessage[i] + "</li>")

}

//歌词信息

for(var i = 0; i < lyricResult.length; i++) {

ul.append("<li>" + lyricResult[i][1] + "</li>")

}

$("#lyricDiv").append(ul);

}

//监听audio的timeUpdate事件,

//第一句歌词会在点击播放按钮时显示出来,

//如果当前时间已经大于第一句歌词的时间,则说明第一句唱完了,

//则把第一句隐藏,更改歌词内容,显示第二句歌词.

//再把当前时间跟第三句歌词时间进行比较,依次循环.

var lyricIndex = 1; //记录当前是第几条歌词

function showLyric() {

//因为歌词头部还有歌手信息,因此高亮的歌词从j开始

var j = lyricArtistMessage.length;

$("#lyricDiv ul li").get(j).style.color = "#ff6666";

//检测第第三句歌词,如果是中文,则歌词大小改为18px

if(lyricResult.length > 3) {

for(var i = 0; i < lyricResult[2][1].length; i++) {

//因为歌词英文的分号为’,所以也要排除这个

if(lyricResult[2][1].charCodeAt(i) > 127 && lyricResult[2][1].charAt(i) != '’') {

$("#lyricDiv ul").css("font-size", "19px");

$("#lyric").css("font-size", "19px");

}

}

}

$("#audio")[0].ontimeupdate = function(e) {

if(lyricResult.length > 0 && this.currentTime > lyricResult[lyricIndex][0]) {

//如果大屏歌词没有打开,则小框进行显示

if($("#lyricDiv").css("display") == "none") {

$("#lyric").fadeOut(0);

$("#lyric").text(lyricResult[lyricIndex][1]);

$("#lyric").fadeIn();

} else {

$("#lyric").text("歌词");

//把上一条歌词颜色进行恢复

$("#lyricDiv ul li").css("color", "#cbc7c7");

//把现在这条歌词高亮显示

$("#lyricDiv ul li").get(lyricIndex + j).style.color = "#ff6666";

//初始歌词的高度是80,所以将80-叠加的高度则得出歌词需要滑动的高度

$("#lyricDiv ul").animate({

"top": 80 - lyricHeight[lyricIndex - 1] + "px"

}, 1000);

}

//把歌词的索引移到下一条

lyricIndex++;

}

}

}

function showFullLyric() {

$(".header button").css("display", "none");

$("#back").css("display", "block");

$(".musicList").css("display", "none");

$(".albumPic").css("display", "none");

$("#lyricDiv").css("display", "block");

//把所有歌词的高度保存,因为设为none之后该值也会丢失

//为了方面后面设置高度,保存的值是叠加高度的值

lyricHeight = [];

for(var i = 0; i < $("#lyricDiv ul li").length; i++) {

var last = i == 0 ? 0 : lyricHeight[i - 1];

lyricHeight.push(last + $("#lyricDiv ul li")[i].offsetHeight);

}

$("#lyric").text("歌词");

//迅速滑到当前播放的歌词

//初始歌词的高度是80,所以将80-叠加的高度则得出歌词需要滑动的高度

if(lyricIndex >= 2){

$("#lyricDiv ul").animate({

"top": 80 - lyricHeight[lyricIndex - 1] + "px"

}, 1000);

}

}

function lyricTip(t, e) {

var lyric = document.getElementsByClassName("lyric")[0];

var tooltipTop = lyric.offsetTop;

if(e.target.id == "lyric") {

var tooltipHtml = "<div id='tooltip' class='tooltip'>点击查看全部歌词 </div>";

$(t).append(tooltipHtml); //添加到页面中

$("#tooltip").css({

"top": tooltipTop + 30 + "px",

"left": "210px",

}).show("fast"); //设置提示框的坐标，并显示

}

}

function tipOut() {

$("#tooltip").remove();

}

function showShortcut(t, e) {

var title = document.getElementById("title");

var tooltipTop = title.offsetTop;

if(e.target.id == "title") {

var tooltipHtml = "<div id='shortCutTooltip' class='tooltip' style='color:#ff6666;'>播放:空格键<br/>下一首:Ctrl+Alt+ -><br/>上一首:Ctrl+Alt+ <-</div>";

$(t).append(tooltipHtml); //添加到页面中

$("#shortCutTooltip").css({

"top": tooltipTop + 40 + "px",

"left": "120px",

}).show("fast"); //设置提示框的坐标，并显示

}

}

function showShortcutOut() {

$("#shortCutTooltip").remove();

}

//画星空背景

function drawStars() {

var canvas = document.getElementById('canvas'),

ctx = canvas.getContext('2d'),

w = canvas.width = window.innerWidth,

h = canvas.height = window.innerHeight,

hue = 217, //色调色彩

stars = [], //保存所有星星

count = 0, //用于计算星星

maxStars = 1300; //星星数量

//canvas2是用来创建星星的源图像，即母版，

//根据星星自身属性的大小来设置

var canvas2 = document.createElement('canvas'),

ctx2 = canvas2.getContext('2d');

canvas2.width = 100;

canvas2.height = 100;

//创建径向渐变，从坐标(half，half)半径为0的圆开始，

//到坐标为(half,half)半径为half的圆结束

var half = canvas2.width / 2,

gradient2 = ctx2.createRadialGradient(half, half, 0, half, half, half);

gradient2.addColorStop(0.025, '#CCC');

//hsl是另一种颜色的表示方式，

//h->hue,代表色调色彩，0为red，120为green，240为blue

//s->saturation，代表饱和度，0%-100%

//l->lightness，代表亮度，0%为black，100%位white

gradient2.addColorStop(0.1, 'hsl(' + hue + ', 61%, 33%)');

gradient2.addColorStop(0.25, 'hsl(' + hue + ', 64%, 6%)');

gradient2.addColorStop(1, 'transparent');

ctx2.fillStyle = gradient2;

ctx2.beginPath();

ctx2.arc(half, half, half, 0, Math.PI \* 2);

ctx2.fill();

// End cache

function random(min, max) {

if(arguments.length < 2) {

max = min;

min = 0;

}

if(min > max) {

var hold = max;

max = min;

min = hold;

}

//返回min和max之间的一个随机值

return Math.floor(Math.random() \* (max - min + 1)) + min;

}

1. 用户登录界面代码

<!DOCTYPE html>

<html lang="zh-cn">

<head>

<meta charset="UTF-8">

<title>账户注册</title>

<style type="text/css">

body,input{

margin: 0;

padding: 0;

background: pink;

}

input{

display: inline-block;

background: #fff;

}

.xiao-container{

width: 100%;

}

.xiao-register-box{

position: relative;

height: 800px;

width: 800px;

top: 50px;

margin: 0 auto;

z-index: 99999;

background:#ffffff;

border: 7px solid #ccc;

}

.xiao-title-box{

position: absolute;

width: 300px;

height: 50px;

margin-left: 250px;

margin-top: 5px;

text-align: center;

font-size: 28px;

font-weight: 800;

color: #ff5000;

line-height: 50px;

}

.xiao-username-box{

position: absolute;

width: 420px;

height: 40px;

line-height: 40px;

margin-top:100px;

margin-left:80px;

font-weight: 700;

}

.xiao-username-input{

display: inline-block;

margin-left: 45px;

/\*background: green;\*/

}

#username{

height: 35px;

width: 290px;

border: 2px solid #ccc;

border-radius: 5px;

}

.xiao-userPassword-box{

position: absolute;

width: 420px;

height: 40px;

line-height: 40px;

margin-top:180px;

margin-left:80px;

font-weight: 700;

}

.xiao-userPassword-input{

display: inline-block;

margin-left: 61px;

}

#userPassword{

height: 35px;

width: 290px;

border: 2px solid #ccc;

border-radius: 5px;

}

.xiao-userRePassword-box{

position: absolute;

width: 420px;

height: 40px;

line-height: 40px;

margin-top:260px;

margin-left:80px;

font-weight: 700;

}

.xiao-userRePassword-input{

display: inline-block;

margin-left: 28px;

}

#userRePassword{

height: 35px;

width: 290px;

border: 2px solid #ccc;

border-radius: 5px;

}

.xiao-userPhone-box{

position: absolute;

width: 420px;

height: 40px;

line-height: 40px;

margin-top:340px;

margin-left:80px;

font-weight: 700;

}

.xiao-userPhone-input{

display: inline-block;

margin-left: 28px;

}

#userPhone{

height: 35px;

width: 290px;

border: 2px solid #ccc;

border-radius: 5px;

}

.xiao-userEmail-box{

position: absolute;

width: 420px;

height: 40px;

line-height: 40px;

margin-top:420px;

margin-left:80px;

font-weight: 700;

}

.xiao-userEmail-input{

display: inline-block;

margin-left: 28px;

}

#userEmail{

height: 35px;

width: 290px;

border: 2px solid #ccc;

border-radius: 5px;

}

.xiao-userGender-box{

position: absolute;

width: 420px;

height: 40px;

line-height: 40px;

margin-top:500px;

margin-left:82px;

font-weight: 700;

}

.xiao-userGender-input{

display: inline-block;

margin-left: 62px;

}

.xiao-require{

color: red;

}

.xiao-submit-box{

position:absolute;

width: 80px;

height: 40px;

line-height: 40px;

margin-top: 580px;

margin-left:200px;

border-radius: 5px;

background: grey;

}

#xiao-submit-button{

display: inline-block;

width: 80px;

height: 40px;

border-radius: 5px;

background: red;

}

.xiao-goLogin-box{

position:absolute;

width: 150px;

height: 20px;

margin-top: 600px;

margin-left:320px;

}

</style>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>茜色音乐-登陆</title>

<!-- Bootstrap -->

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="css/nav.css" rel="stylesheet">

<link href="css/css.css" rel="stylesheet">

<style>

.slider{max-width:1920px;text-align:center;}

</style>

</head>

<body>

<div class="heads">

<div class="top">

<span class="headerRight"><img src="images/BIAOTI.png" width="273" height="19" /></span>

</div><!--top/-->

</div><!--heads/-->

<h1 class="xlogo">

<a href="index.html"><img src="images/logo.png" width="210" height="77" text-align:left;/></a>

</h1>

<a href="zhuce.html"><p class="denglu">没有账号？去注册</p></a>

<nav class="nav-collapse">

<ul>

<li class="navCur"><a href="index.html">首页</a></li>

<li><a href="#">推荐歌曲</a>

<div class="chilNav">

<div class="row chilNavCar">

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tu1.jpg" alt="...">

<div class="caption">

<p>がくっぽいど</p>

<p><a href="#" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tua.jpg" alt="...">

<div class="caption">

<p>歌爱ユキ AHS ”</p>

<p><a href="#" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tub.jpg" alt="...">

<div class="caption">

<p>氷山キヨテル AHS</p>

<p><a href="#" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tuc.jpg" alt="...">

<div class="caption">

<p>あの”</p>

<p><a href="#" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tud.jpg" alt="...">

<div class="caption">

<p>琪露诺</p>

<p><a href="#" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

</div><!--row/-->

</div><!--chilNav/-->

</li>

<li><a href="#">在线歌曲</a></li>

<li><a href="#">我的歌单</a>

<div class="chilNav" id="chilNav">

<div class="chiNava"><a href="#">我创建的</a></div>

<div class="chiNava"><a href="#">我收藏的</a></div>

</div><!--chilNav/-->

</li>

<li><a href="#">茜色电台</a></li>

<li><a href="#">联系我们</a></li>

</ul>

</nav>

<div class="heights">&nbsp;</div>

<form action="denglu.php" method="post">

<center>

<div class="xiao-username-box">

<span class="xiao-require">\*</span>

<label for="username">用户名</label>

<div class="xiao-username-input">

<input type="text" id="username" name="username" placeholder="请输入用户名 长度:6-12个字符" />

</div>

</div>

<div class="xiao-userPassword-box">

<span class="xiao-require">\*</span>

<label for="userPassword">密码</label>

<div class="xiao-userPassword-input">

<input type="password" id="userPassword" name="userPassword" placeholder="请输入密码 长度:6-12个字符" />

</div>

</div>

<div class="xiao-submit-box">

<input id = "xiao-submit-button" type="submit" value="登陆">

</div>

<div class="xiao-goLogin-box">

<a href="zhuce.html" style="text-decoration: none;">没有账号？去注册</a>

</div>

</center>

</form>

<div class="footerfa">

<p>

<a href="index.html">首页</a>

<a href="#">在线歌曲</a>

<a href="#">推荐歌曲</a>

<a href="#">我的歌单</a>

<a href="#">茜色电台</a>

<a href="#" style="border:0;">联系我们</a>

</p>

</div><!--footer/-->

<!--客服代码 start-->

<div class="kefu">

<div class="kefuBack">

<img src="images/fuchuang.jpg" width="25" height="147" class="keLeft" />

<div class="kefuRight">

<div class="kefuRightBox">

地址：山东大学<br />

电话：18981792086<br />

客服QQ：<a href="http://wpa.qq.com/msgrd?v=3&uin=809324688&site=qq&menu=yes" target="\_blank" title="点击在线咨询">809324688</a> <br />

<span class="kefuRightClick">&nbsp;</span>

</div><!--kefuRightBox/-->

</div><!--kefuRight/-->

</div><!--kefuBack/-->

</div><!--kefu-->

<div class="pinggu">

&nbsp;

</div>

<div class="pingguClick">&nbsp;</div>

<!--客服代码 end-->

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="js/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

<script src="js/js.js"></script>

<!--响应式导航-->

<script>

var navigation = responsiveNav(".nav-collapse", {

animate: true, // Boolean: Use CSS3 transitions, true or false

transition: 284, // Integer: Speed of the transition, in milliseconds

label: "Menu", // String: Label for the navigation toggle

insert: "after", // String: Insert the toggle before or after the navigation

customToggle: "", // Selector: Specify the ID of a custom toggle

closeOnNavClick: false, // Boolean: Close the navigation when one of the links are clicked

openPos: "relative", // String: Position of the opened nav, relative or static

navClass: "nav-collapse", // String: Default CSS class. If changed, you need to edit the CSS too!

navActiveClass: "js-nav-active", // String: Class that is added to <html> element when nav is active

jsClass: "js", // String: 'JS enabled' class which is added to <html> element

init: function(){}, // Function: Init callback

open: function(){}, // Function: Open callback

close: function(){} // Function: Close callback

});

</script>

<!--banner焦点轮换-->

<script src="js/jquery.excoloSlider.js"></script>

<script>

$(function () {

$("#sliderA").excoloSlider();

});

</script>

<!--图片左右滚动-->

<script type="text/javascript" src="js/slider.js"></script>

<script type="text/javascript">

$(function(){

//默认状态下左右滚动

$("#s1").xslider({

unitdisplayed:4,

movelength:1,

unitlen:300,

autoscroll:30000

});

})

</script>

</body>

</html>

1. 主页代码

<!DOCTYPE html>

<html lang="zh-cn">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>茜色音乐</title>

<!-- Bootstrap -->

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="css/nav.css" rel="stylesheet">

<link href="css/css.css" rel="stylesheet">

<style>

.slider{max-width:1920px;text-align:center;}

</style>

</head>

<body>

<div class="heads">

<div class="top">

<span class="headerRight"><img src="images/BIAOTI.png" width="273" height="19" /></span>

</div><!--top/-->

</div><!--heads/-->

<h1 class="xlogo">

<a href="index.html"><img src="images/logo.png" width="210" height="77" text-align:left;/></a>

</h1>

<a href="denglu.html"><p class="denglu">登录</p></a>

<p class="denglu">丨</P>

<a href="zhuce.html"><p class="denglu">注册</p></a>

<nav class="nav-collapse">

<ul>

<li class="navCur"><a href="index.html">首页</a></li>

<li><a href="#.html">推荐歌曲</a>

<div class="chilNav">

<div class="row chilNavCar">

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tu1.jpg" alt="...">

<div class="caption">

<p>がくっぽいど</p>

<p><a href="gqxq.html" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tua.jpg" alt="...">

<div class="caption">

<p>歌爱ユキ AHS ”</p>

<p><a href="gqxq.html" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tub.jpg" alt="...">

<div class="caption">

<p>氷山キヨテル AHS</p>

<p><a href="gqxq.html" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tuc.jpg" alt="...">

<div class="caption">

<p>あの”</p>

<p><a href="gqxq.html" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

<div class="col-sm-6 col-md-2">

<div class="thumbnail">

<img data-src="holder.js/300x300" src="images/tud.jpg" alt="...">

<div class="caption">

<p>琪露诺</p>

<p><a href="gqxq.html" class="more">查看详情&gt;&gt;</a></p>

</div>

</div>

</div><!--col-sm-6 col-md-2/-->

</div><!--row/-->

</div><!--chilNav/-->

</li>

<li><a href="bofang.html">在线歌曲</a></li>

<li><a href="bofang.html">我的歌单</a>

<div class="chilNav" id="chilNav">

<div class="chiNava"><a href="#">我创建的</a></div>

<div class="chiNava"><a href="#">我收藏的</a></div>

</div><!--chilNav/-->

</li>

<li><a href="bofang.html">茜色电台</a></li>

<li><a href="#">联系我们</a></li>

</ul>

</nav>

<div class="heights">&nbsp;</div>

<!--banner-->

<div id="sliderA" class="slider">

<img src="images/bannera.jpg" data-plugin-slide-caption="" />

<img src="images/bannerb.jpg" data-plugin-slide-caption="" />

<img src="images/bannerc.jpg" data-plugin-slide-caption="" />

<img src="images/bannerd.jpg" data-plugin-slide-caption="" />

</div>

<!--banner-->

<div class="hengfu hfBg1">

<div class="hengfuParBox">

<div class="hengfuPar">

<h3>妄想症Paranoia</h3>

<p>洛天依言和乐正绫原创系列。 </p>

<div class="more"><a href="gqxq.html"></a></div>

</div><!--hengfuPar/-->

</div><!--hengfuParBox/-->

</div><!--hengfu/-->

<div class="hengfu hfBg2">

<div class="hengfuParBox">

<div class="hengfuPar">

<h3>透明色</h3>

<p>歌手：纯白P 时间：2014-05-08</p>

<div class="more"><a href="gqxq.html"></a></div>

</div><!--hengfuPar/-->

</div><!--hengfuParBox/-->

</div><!--hengfu/-->

<div class="hengfu hfBg3">

<div class="hengfuParBox">

<div class="hengfuPar">

<h3>故梦（翻唱）</h3>

<p>歌手：双笙 时间：2016-04-15</p>

<div class="more"><a href="gqxq.html"></a></div>

</div><!--hengfuPar/-->

</div><!--hengfuParBox/-->

</div><!--hengfu/-->

<div class="removeHengfu hfBg1">

<div class="hengfuLeft">

<img src="images/hengfua\_a.jpg" />

</div><!--hengfuLeft/-->

<div class="hengfuParBox">

<div class="hengfuPar">

<h3>妄想症Paranoia</h3>

<p>洛天依言和乐正绫原创系列。 时间：2011-06-08</p>

<div class="more"><a href="gqxq.html">查看更多</a></div>

</div><!--hengfuPar/-->

</div><!--hengfuParBox/-->

<div class="clearfix"></div>

</div><!--removeHengfu/-->

<div class="removeHengfu hfBg2">

<div class="hengfuLeft">

<img src="images/hengfua\_b.jpg" />

</div><!--hengfuLeft/-->

<div class="hengfuParBox">

<div class="hengfuPar">

<h3>透明色</h3>

<p>歌手：纯白P 时间：2014-05-08</p>

<div class="more"><a href="gqxq.html">查看更多</a></div>

</div><!--hengfuPar/-->

</div><!--hengfuParBox/-->

<div class="clearfix"></div>

</div><!--removeHengfu/-->

<div class="removeHengfu hfBg1">

<div class="hengfuLeft">

<img src="images/hengfua\_c.jpg" />

</div><!--hengfuLeft/-->

<div class="hengfuParBox">

<div class="hengfuPar">

<h3>故梦（翻唱）</h3>

<p>歌手：双笙 时间：2016-04-15</p>

<div class="more"><a href="gqxq.html">查看更多</a></div>

</div><!--hengfuPar/-->

</div><!--hengfuParBox/-->

<div class="clearfix"></div>

</div><!--removeHengfu/-->

<script src="http: //aboutplayer.com/music.js"></script>

<h4 class="h4Title">相似网站</h4>

<div class="scrolllist" id="s1">

<a class="abtn aleft" href="#left" title="左移"><img src="images/left.jpg" width="29" height="239"></a>

<div class="imglist\_w">

<ul class="imglist">

<li><img src="images/xb.jpg" />

<p>网易云音乐</p>

<p><a href="https://music.163.com/" class="more">前往&gt;&gt;</a></p>

</li>

<li><img src="images/xa.jpg" />

<p>QQ音乐”</p>

<p><a href="https://y.qq.com/" class="more">前往&gt;&gt;</a></p>

</li>

<li><img src="images/xc.jpg" />

<p>酷狗音乐”</p>

<p><a href="http://www.kugou.com/" class="more">前往&gt;&gt;</a></p>

</li>

<li><img src="images/xd.jpg" />

<p>酷我音乐”</p>

<p><a href="http://www.kuwo.cn/" class="more">前往&gt;&gt;</a></p>

</li>

<li><img src="images/xe.jpg" />

<p>千千音乐”</p>

<p><a href="http://music.taihe.com/" class="more">前往&gt;&gt;</a></p>

</li>

<li><img src="images/xf.jpg" />

<p> 咪咕音乐”</p>

<p><a href="http://www.migu.cn/" class="more">前往&gt;&gt;</a></p>

</li>

<li><img src="images/xg.jpg" />

<p>优酷音乐”</p>

<p><a href="http://music.youku.com/" class="more">前往&gt;&gt;</a></p>

</li>

<div class="clearfix"></div>

</ul><!--imglist end-->

</div>

<a class="abtn aright" href="#right" title="右移"><img src="images/right.jpg" width="29" height="239" /></a>

</div><!--scrolllist end-->

<div class="footer">

<p>

<a href="index.html">首页</a>

<a href="#">在线歌曲</a>

<a href="#">推荐歌曲</a>

<a href="#">我的歌单</a>

<a href="#">茜色电台</a>

<a href="#" style="border:0;">联系我们</a>

</p>

</div><!--footer/-->

<!--客服代码 start-->

<div class="kefu">

<div class="kefuBack">

<img src="images/fuchuang.jpg" width="25" height="147" class="keLeft" />

<div class="kefuRight">

<div class="kefuRightBox">

地址：山东大学<br />

电话：18981792086<br />

客服QQ：<a href="http://wpa.qq.com/msgrd?v=3&uin=809324688&site=qq&menu=yes" target="\_blank" title="点击在线咨询">809324688</a> <br />

<span class="kefuRightClick">&nbsp;</span>

</div><!--kefuRightBox/-->

</div><!--kefuRight/-->

</div><!--kefuBack/-->

</div><!--kefu-->

<div class="pinggu">

&nbsp;

</div>

<div class="pingguClick">&nbsp;</div>

<!--客服代码 end-->

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->

<script src="js/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

<script src="js/js.js"></script>

<!--响应式导航-->

<script>

var navigation = responsiveNav(".nav-collapse", {

animate: true, // Boolean: Use CSS3 transitions, true or false

transition: 284, // Integer: Speed of the transition, in milliseconds

label: "Menu", // String: Label for the navigation toggle

insert: "after", // String: Insert the toggle before or after the navigation

customToggle: "", // Selector: Specify the ID of a custom toggle

closeOnNavClick: false, // Boolean: Close the navigation when one of the links are clicked

openPos: "relative", // String: Position of the opened nav, relative or static

navClass: "nav-collapse", // String: Default CSS class. If changed, you need to edit the CSS too!

navActiveClass: "js-nav-active", // String: Class that is added to <html> element when nav is active

jsClass: "js", // String: 'JS enabled' class which is added to <html> element

init: function(){}, // Function: Init callback

open: function(){}, // Function: Open callback

close: function(){} // Function: Close callback

});

</script>

<!--banner焦点轮换-->

<script src="js/jquery.excoloSlider.js"></script>

<script>

$(function () {

$("#sliderA").excoloSlider();

});

</script>

<!--图片左右滚动-->

<script type="text/javascript" src="js/slider.js"></script>

<script type="text/javascript">

$(function(){

//默认状态下左右滚动

$("#s1").xslider({

unitdisplayed:4,

movelength:1,

unitlen:300,

autoscroll:30000

});

})

</script>

</body>

</html>

## 5.2功能

实现以下全部功能。

1. 注册登入功能：

在登入页面（注册页面）登入账号（注册账号）即可。有管理员和普通用户两种登入方式，管理员登入可以对电影排版，哪个电影可以上映，在哪里上映等做出修改。

1. 买票功能：

登入之后，在首页的正在热映页面可以看到如今正在热卖的电影，可能会快速找到自己喜爱的电影。

我们也可以点击首页的右方电影按钮，然后在里面可以查看如今所有在映的电影。我们也可以在影院页面查看是否有自己喜欢的电影，每个影院都有自己上映的电影的清单。

最后我们可以直接在首页上方的搜索界面搜索想要看的电影。

在上述所有操作中只要找到想要购买的电影票，都可以在当前页面购买。

1. 搜索功能：

在首页上方直接搜索，可以搜索自己想看的电影，或者自己心仪的电影院。

1. 预购功能：

在首页的即将上映页面又即将上映的电影，我们可以在里面进行预购。

1. 查看订单功能：

在右下方“我的”界面可有“我的订单”功能，在这里我们可以看到自己的所有历史订单，清楚自己的每一笔消费。

1. 想看的电影：

也是在右下方“我的”界面右边“想看的电影”可以在那里写入想看的电影。

1. 看过的电影：

同是“我的”界面，我们可以查看自己所有看过的电影，留下自己的点点滴滴。

1. 评论功能：

可以再电影下进行评论，谈谈自己对电影的感受等。

## 5.3性能

精确度较高，且及时响应程度高

## 5.4输入项

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 输入方式 | 输入媒介 | 输入来源 |
| Username | Username | Varchar | 用户输入 | 键盘 | TextBox1 |
| Password | Password | Varchar | 用户输入 | 键盘 | TextBox2 |
| Moviename | Moviename | Varchar | 参数传递 | 系统 | netSearchList.js |

## 5.5输出项

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 名称 | 标识 | 数据类型 | 输出方式 | 输出来源 |
| Username | Username | Varchar | 写入数据 | TextBox1 |
| Password | Password | Varchar | 写入数据 | TextBox2 |
| Movie | Movie | Varchar | 写入数据 | netSearchList.js |

## 5.6算法

推荐算法简单说就是在海量的用户数据（行为记录等）中对用户进行划分，对同一群体的用户推荐其他用户喜欢的电影。

这其中需要给电影分类并建立评分细则、建立用户模型、寻找相似用户，基于用户的行为数据将电影分类匹配

将电影分成三个部分：系统推荐 、个人喜好推荐 、同好推荐 。

1. 从准确性、多样性角度分析

系统推荐电影

准确性低、多样性高

多样性高能为用户带来新鲜感，如果发现了一个从未听过但特别喜欢的影片，会带来惊喜感，调动用户正面情绪。

可是由于准确性低，很可能新影片很不被用户喜欢，所以在系统推荐电影在播放界面设置“删除”、“下一个”两个按键便于用户切换电影。

个人喜好推荐

准确性高、多样性低

准确性高使得每日推荐的20个电影比较好的满足用户口味，但是存在电影类型单一化的问题，因此设置了播放列表以提供用户浏览、操作的权利，弥补曲目单一化带给用户的失望。

同好推荐

准确性中、多样性中

同好推荐 有别于其他两个个性化推荐功能，它准确性多样性的阈值不只是由算法决定的，更多的是它功能形式所决定的。先把功能的面向对象分为两类，一类是用户，一类是UGC片单，系统分别为片单和用户加标签以提高准确度，由于UGC片单是由很多用户创建，所以UGC片单就具有多样性，两者糅合从而保证了准确度和多样性共存。

2. 从操作流程上分析

三个功能从看见功能按键到最终获得推荐曲目的步骤：

看见系统推荐电影>点击系统推荐电影>获取电影

看见个人喜好推荐 >点击个人喜好推荐 >看见推荐列表>筛选喜欢曲目>点击喜欢曲目>获取电影

看见同好推荐 >点击同好推荐 >跳转片单页面>发现类型标签>筛选类型标签>点击类型标签>看见标签下的同好推荐 >筛选片单>点击片单>浏览片单列表>筛选喜欢电影>点击喜欢电影>获取电影

可以发现三种方式获取推荐电影的操作流程由简入繁。

3. 从用户使用阶段分析

三个功能对应着三种用户阶段：

系统推荐电影-新用户：

系统推荐电影位于个页黄金位置，新用户初次体验的产品功能时大概率点击这个按键，所以要简化用户使用流程，用户在快速感受产品个性化推荐的魅力后才产生继续了解其他功能的欲望。

个人喜好推荐 -普通用户：

新用户使用系统推荐电影过后需要不一样的体验来满足个性化需求。

每日20个电影推荐对用户来说是可预知的，20个上限的设定给用户物以稀为贵的感觉，会珍惜每日的推荐，而每日更新无法回看以往推荐的设定，会让用户觉得一天不看就错过了什么的紧迫感。

推荐算法设定了基于不同用户行为的权重，“想看”最高，评论、搜索、分享其次

同好推荐 -深度用户：

片单是云电影连接个性化推荐和社交的重要桥梁，同好推荐 是个性化推荐功能最后一环。

在深度体验了同好推荐 之后，用户会得到片单可被分享和推荐的认识，很可能会产生自建片单的冲动。

而片单在云电影中具有社交属性，用户可以互相收藏、评论、分享片单，而且片单在个人主页中也反映了个人电影风格，让用户能够更好地展现自己给他人。

4. 从参与元素分析

系统推荐电影：系统

个人喜好推荐 ：系统+自己

同好推荐 ：系统+自己+其他用户

欧氏距离vs余弦相似度

在线下导购时代，导购员会通过系统的话术掌握消费者的情况，来推荐商品。

类比导购员推销时的思维逻辑，我们可以得到电影推荐算法需要解决的三个核心问题：

将用户信息转化为用户类型；

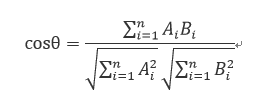
我们怎么量化两个事物之间的相似度呢？常见的方法是利用欧式距离和余弦相似度。利用欧式距离时，我们把A、B用户看做两点，用两点间距离表示二者相似度。使用余弦相似度时，则把二者看成同一坐标系下的两个向量。两个向量间夹角大小反应出他们的相似度，夹角越小则相似度越大。二维空间向量表示为r(x1,x2),多维空间向量表示为r(x1,x2…,xn) 比如，假设用户有5个维度

对喜剧的喜欢程度（1~5分），对动作片的喜欢程度（1~5分），对爱情片的喜欢程度（1~5分），对恐怖片的喜欢程度（1~5分），对科幻片的喜欢程度（1~5分）。

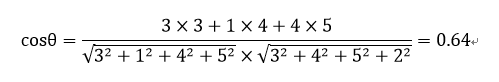
用户A：对喜剧的喜欢程度3，对动作片的喜欢程度1，对爱情片的喜欢程度4，对恐怖片的喜欢程度5，对科幻片喜欢程度0，用户A可以用向量表示为r\_A (3,1,4,5,0)

一个用户B：对喜剧的喜欢程度3，对动作片的喜欢程度4，对爱情片的喜欢程度5，对恐怖片的喜欢程度0，对科幻片的喜欢程度2，用户B可以用向量表示为r\_B (3,4,5,0,2)

对于向量A和B而言，他们的在多维空间的夹角可以用向量余弦公式计算：

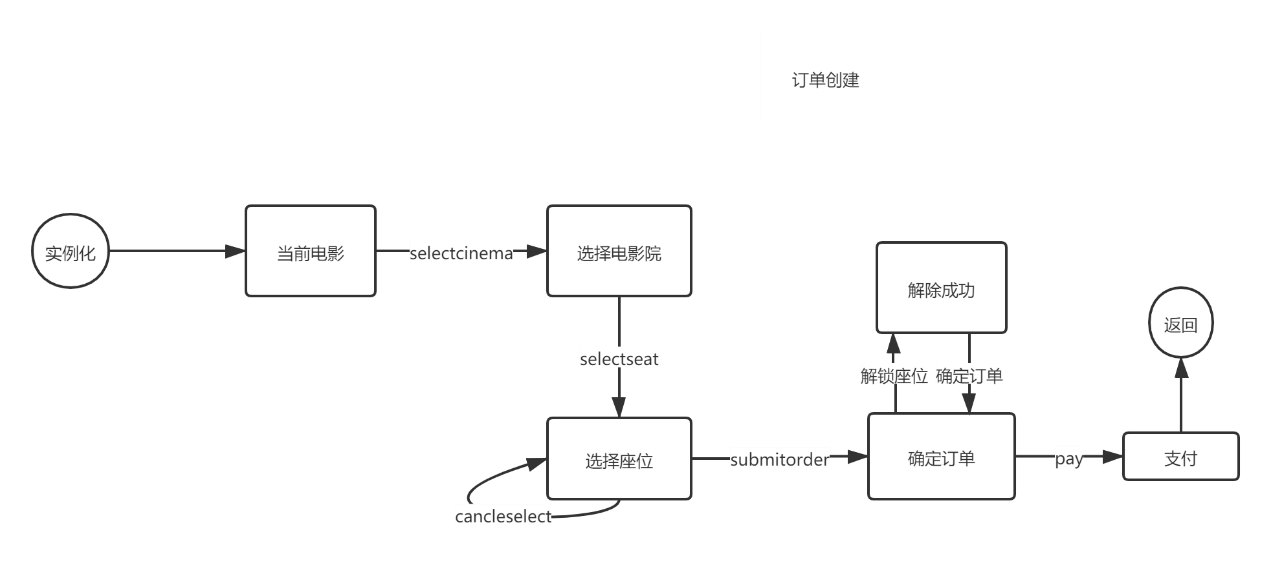


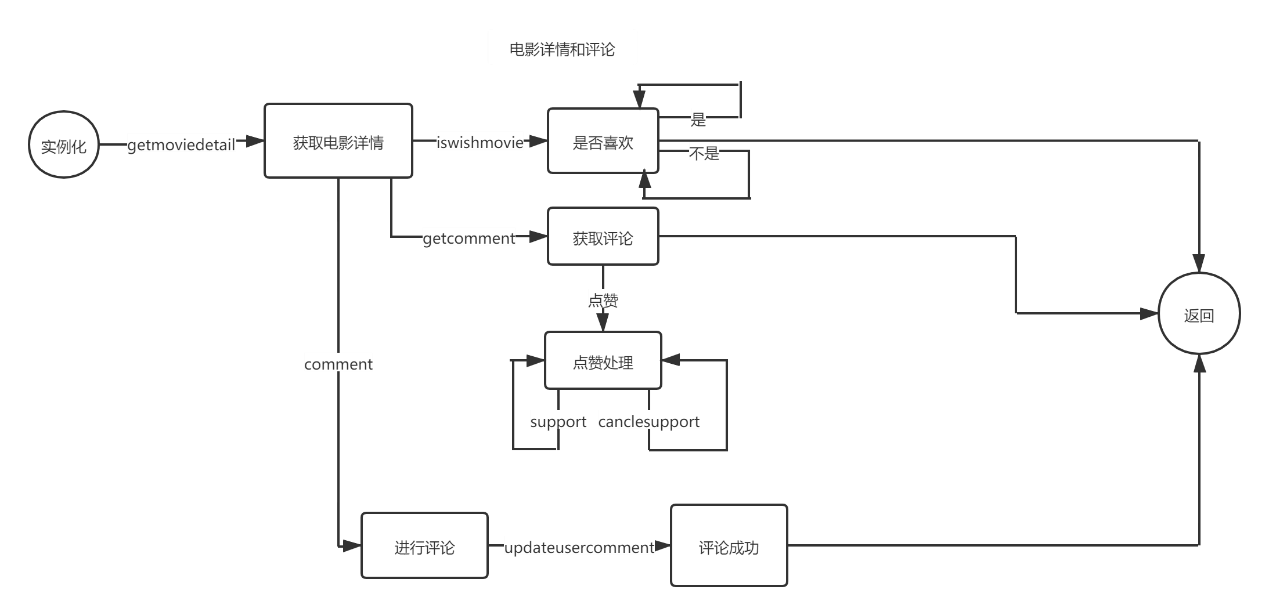
余弦相似度取值在0到1之间，0代表完全正交，1代表完全一致。那么用户A和B的相似度计算：

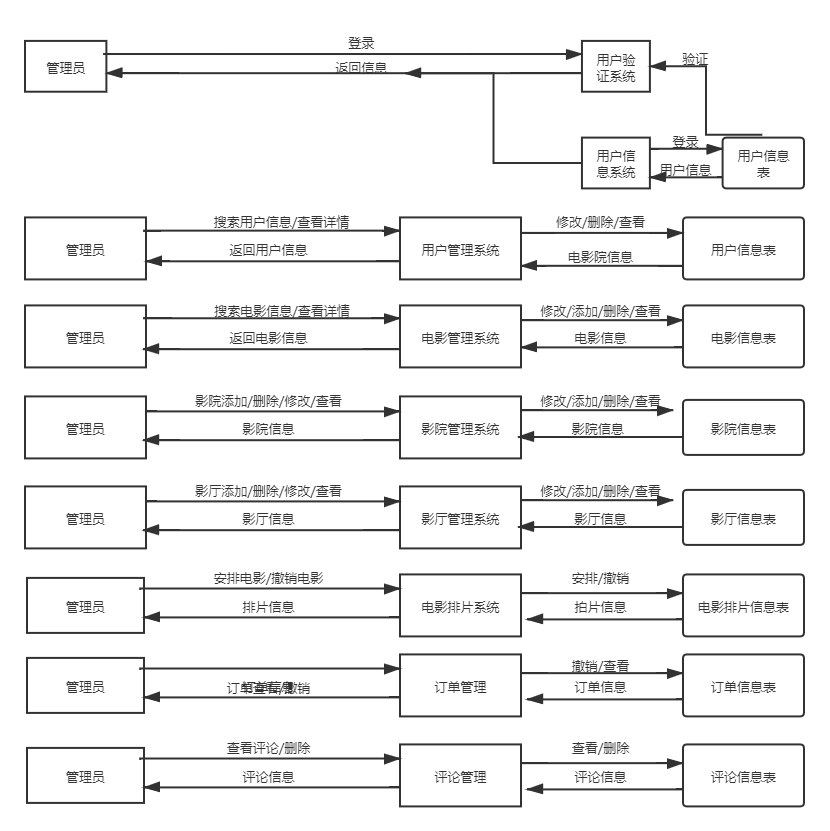


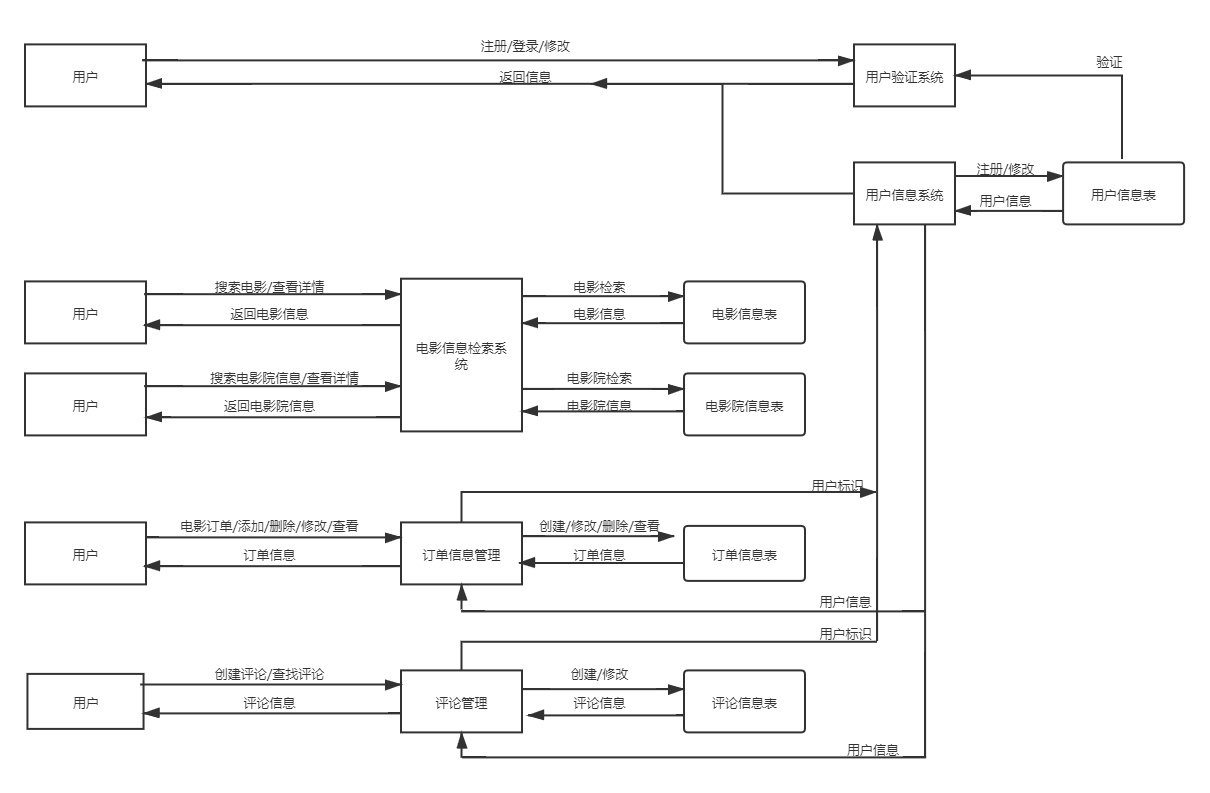
即代表了两个用户电影偏好的相似程度。

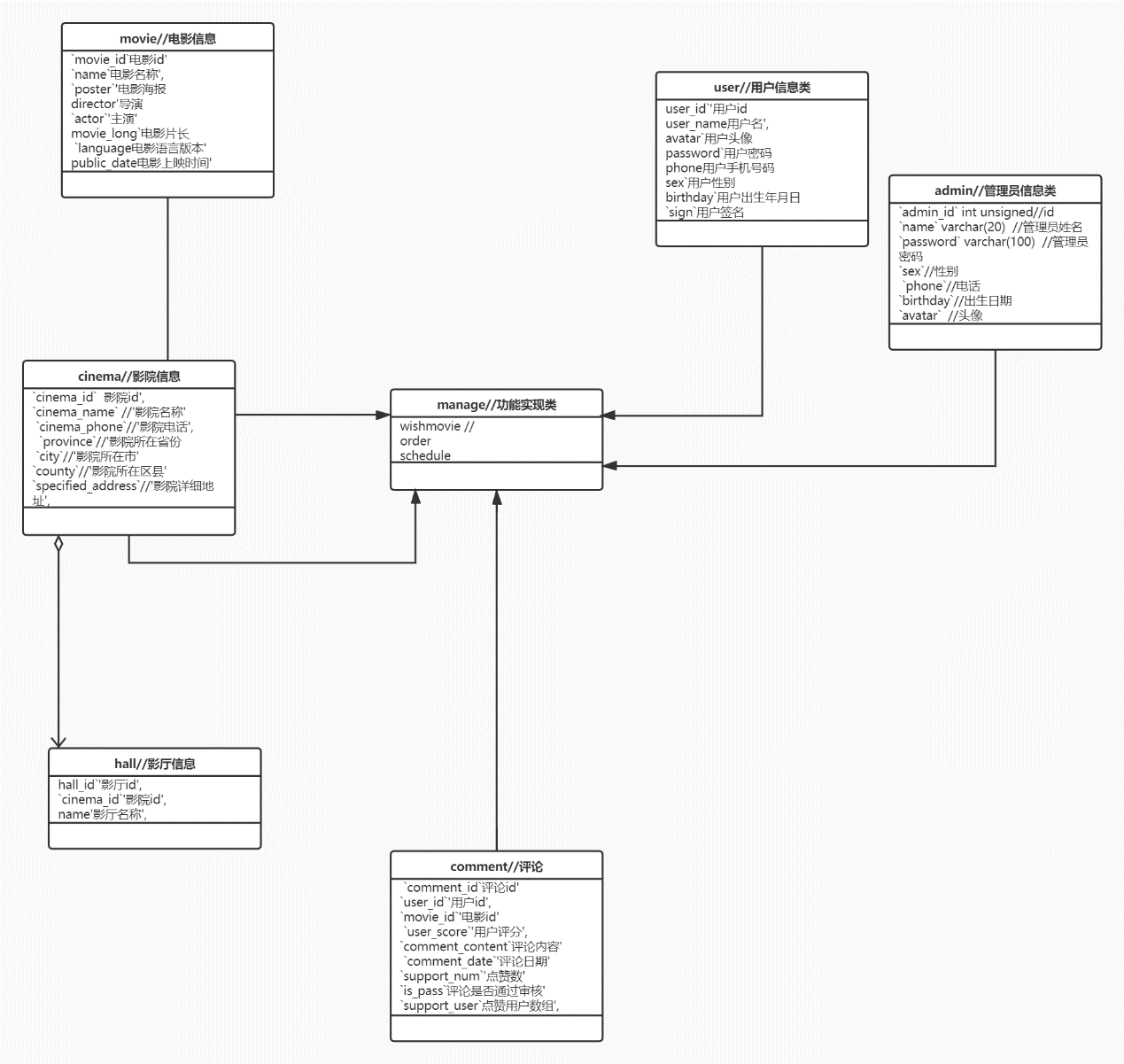
## 5.7流程逻辑





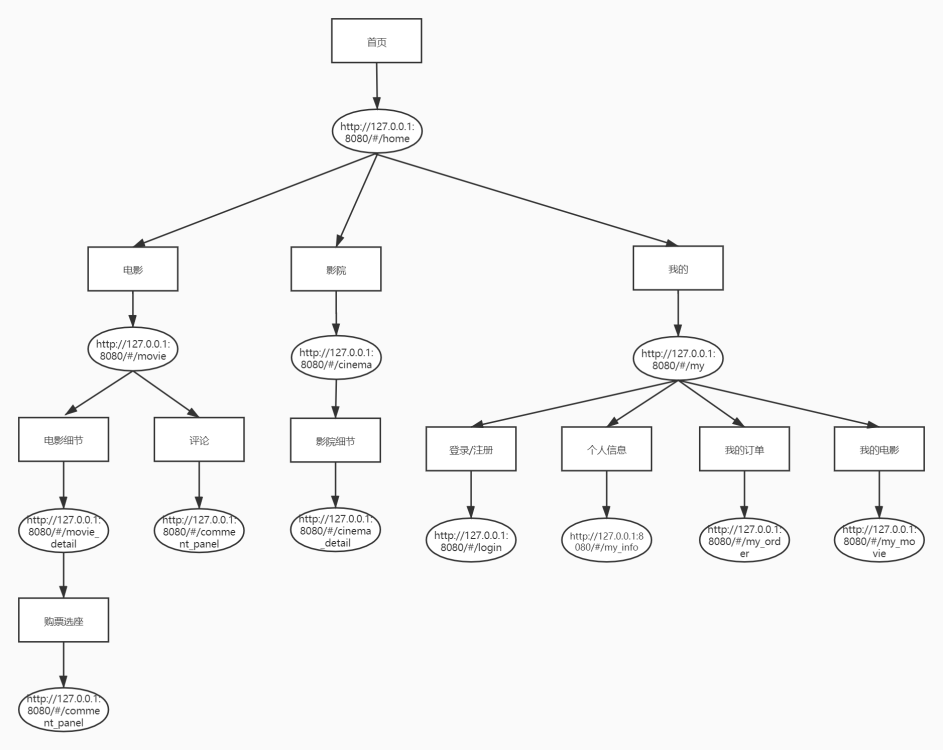






# 6.接口：

接口图：



# 7.注释设计

源文件外部对各部件组成功能进行说明。

函数开始前应进行注释，列出：函数的目的/功能 等

# 8.限制条件

后端只在windows系统下进行开发和测试，在其他系统可能出现无法运行或出现异常的情况。

前端可以在移动端访问，也可以在windows端访问。

# 9.测试计划

技术要求：在windows系统下，安装多个浏览器，如chrome 火狐 ie edge 360 qq浏览器等市面常用浏览器，分别使用它们打开测试项目，观察是否能正常运行。

输入数据：进行搜索，注册，选座，看评论，看电影信息和海报界面看能否正常显示。

预期结果：预期在这些浏览器下都能正常完成操作。

# 10.尚未解决的问题

# 11.注解

参考资料

[1] 《Vue.js从入门到项目实战》 胡同江编著，清华大学出版社

[2] 《Vue.js前端开发技术》， 王凤丽, 豆连军编著 人民邮电出版社

[3][美]Shari Lawrence Pfleeger [加]Joanne M.Atlee 著，《软件工程(第4版)》，人民邮电出版社

[4]刘竹林 编著，《软件工程案例开发与实践》，清华大学出版社

# 12.总结

基于vue.js的在线电影售票系统是一种精简，快捷，方便的实用性平台。该系统不仅可用于影院以及电影的在线管理，也能够记录用户信息和评论、初步实现社区化信息交流。网站式的售票平台使得用户和电影资源的管理更加合理化，信息化。买票的同时还具有电影搜索、电影评价、选座、等功能，可以实现较为灵活的网站内容切换和文本信息管理。

基于vue.js的在线电影售票系统是一个较为整合的系统，它涉及到管理功能和售卖功能等。在这次设计中，小组完成了用户注册/登录，相关数据处理，实时排片等基本功能。通过这次设计，小组对html、JavaScipt和Vue等技术应用方面有了更深入的认识和理解，取得了更大的进步。