**西 南 交 通 大 学**

**继续教育本科毕业设计（论文）**

**基于React框架的音乐播放小程序的设计与实现**

（以下各项居中列）

年 级:2020秋

学 号:20922963

姓 名:旷文杰

专 业:计算机与科学

指导老师:何太军

学习中心:深圳圆梦学习中心

西 南 交 通 大 学

网络教育学院

2023年 5月

**西南交通大学**

**继续教育本科毕业设计（论文）学术诚信声明**

本人郑重声明：所呈交的毕业设计（论文），是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者签名：旷文杰

日期： 2023 年 5 月 9 日

院 系 远程与继续教育学院 专 业 计算机与科学

年 级 2020秋 姓 名 旷文杰

题 目 基于React框架的音乐播放小程序的设计与实现

指导教师

评 语

指导教师 (签章)

评 阅 人

评 语

评 阅 人 (签章)

成 绩

答辩委员会主任 (签章)

年 月

**毕业设计（论文）任务书**

班 级 2020秋 学生姓名 旷文杰 学 号 20922963

定题日期： 2022 年 月 日 完成日期： 11 月 日

题 目 基于React框架的音乐播放小程序的设计与实现

1、本设计（论文）的目的、意义 本设计的主要有两个方面；第一个是对这两年所学的知识、技能做一个全面的考核；第二个是培养自我独立思考和解决问题的能力。

2、学生应完成的任务 1.毕业设计选题2、毕业设计的技术方案3、毕业设计的程序编码4、毕业设计的总结和改善

3、本设计（论文）与本专业的毕业要求达成度如何？（如在知识结构、能力结构、素质结构等方面有哪些有效的训练。）

在知识结构上，完成毕业设计运用软件工程所学进行需求分析与设计编码；能力结构上，学到如何运用前端、后端、数据库去构建一个微信小程序；素质结构方面，独立完成一项毕业设计提升自身思考和解决问题的能力。

4、本设计（论文）各部分内容及时间分配：（共 　 周）

第一部分 选题 ()

第二部分 技术选型 ()

第三部分 demo验证 ()

第四部分 确定方案开始程序编码 ()

第五部分 毕业设计过程稿编写 ()

评阅及答辩 ( 周)

备 注

指导教师： 年 月 日

审 批 人： 年 月 日

摘 要

本文介绍了基于使用React语法开发小程序的Taro多端开发框架在线音乐播放小程序的设计与实现过程，其中采用了Taro框架提供getBackgroundAudioManager音频API实现了音乐播放功能。首先，本文实现了首页推荐歌曲和专辑功能，为用户提供丰富的音乐选择。其次，在搜索页面，用户可以方便地搜索音乐、查看历史搜索记录以及收藏喜欢的音乐。音乐列表页面则将用户收藏的音乐进行渲染展示，方便用户查找和播放。此外，用户页面实现了登录、播放历史等功能，提高了用户体验。播放器页面实现了音乐播放控制、切换音乐、音乐列表、进度条控制以及收藏功能，满足了用户对音乐播放的基本需求.总之，本文设计并实现了一款基于React开发框架的在线音乐播放小程序，具有基础的用户体验和音乐播放的功能。

关键词： 音乐播放器；React语法；Taro框架

ABSTRACT

This article introduces the design and implementation process of an online music playback mini-program based on the Taro multi-end development framework using React syntax. The Taro framework provides the getBackgroundAudioManager audio API to implement music playback functionality. Firstly, this article implements the recommended songs and albums feature on the homepage, providing users with a rich selection of music. Secondly, on the search page, users can conveniently search for music, view historical search records, and favorite the music they like. The music list page then renders and displays the user's favorited music, making it easy for users to find and play. In addition, the user page implements login, playback history, and other functions, enhancing the user experience. The player page implements music playback control, switching music, music list, progress bar control, and favoriting features, meeting users' basic needs for music playback. In conclusion, this article designs and implements an online music playback mini-program based on the React development framework, providing a basic user experience and music playback functionality.

Key words: Music player; React syntax; Taro framework.

目录

[第1章 绪论 1](#_Toc136727878)

[1.1 选题背景 1](#_Toc136727879)

[1.2选题和主要研究方向 2](#_Toc136727880)

[第2章 可行性和需求分析 1](#_Toc136727881)

[2.1可行性分析 1](#_Toc136727882)

[2.2需求分析 1](#_Toc136727883)

[第3章 基础方案设计 1](#_Toc136727884)

[3.1开发技术 1](#_Toc136727885)

[3.2功能模块划分 2](#_Toc136727886)

[3.3性能指标 3](#_Toc136727887)

[3.4技术方案比较 4](#_Toc136727888)

[3.5项目开发周期与耗时 5](#_Toc136727889)

[第4章 详细设计过程与说明 1](#_Toc136727890)

[4.1用户模块 1](#_Toc136727891)

[4.1.1登录 1](#_Toc136727892)

[4.2首页模块 2](#_Toc136727893)

[4.3搜索模块 2](#_Toc136727894)

[4.3.1Search云函数 3](#_Toc136727895)

[4.4音乐列表模块 4](#_Toc136727896)

[4.5播放器模块 4](#_Toc136727897)

[4.6小程序背景音频 5](#_Toc136727898)

[4.7全局数据和状态 6](#_Toc136727899)

[4.8云数据库 7](#_Toc136727900)

[第5章 小程序测试 1](#_Toc136727901)

[5.1小程序功能测试 1](#_Toc136727902)

[第6章 总结与展望 1](#_Toc136727903)

[6.1总结 1](#_Toc136727904)

[6.2展望 1](#_Toc136727905)

[第7章 致谢 1](#_Toc136727906)

[第8章 参考文献 1](#_Toc136727907)

# 绪论

## 1.1 选题背景

随着互联网技术的迅速发展，人们对便利的在线服务需求不断增加。尤其在移动互联网时代，轻量级、易操作的应用成为人们满足日常需求的首选。自2017年1月9日小程序正式上线以来，它逐步成为移动应用的重要形式。到2022年，互联网小程序的日活跃用户已达7.8亿，其中微信小程序超过5亿。这一发展趋势预示着小程序在未来移动互联网市场中有巨大的潜力和空间。音乐作为日常生活中不可或缺的一部分，有着庞大的用户需求。因此，开发一款基于React的在线音乐播放小程序具有实际意义。

微信用户的使用已成为日常习惯，从而自然推动了微信小程序的发展；同时，微信小程序也成为新的流量入口和创业起点，具有以下优点：

无需下载安装：用户通过微信内部搜索或扫码即可打开小程序，降低了用户尝试新应用的门槛。

体积小：小程序体积较小，占用手机存储空间较少。

节省开发成本：使用Taro等跨平台开发框架，一次编写代码，适配多个平台。

更新快速：微信小程序的更新迭代速度较快，可快速修复问题、优化功能，并立即推送给用户。

用户粘性高：微信小程序依托于庞大的微信生态系统，用户粘性高，易于推广。

然而，微信小程序也存在以下缺点：

功能受限：与原生App相比，小程序在功能上存在一定限制。

性能劣势：由于小程序运行在微信内部，性能可能不及原生App。

依赖微信生态：微信小程序需要依赖微信平台，受微信政策和规定制约。

用户隐私和数据安全：用户在使用微信小程序时，可能会担心个人信息和数据安全问题。

总的来说，微信小程序在便捷性、体积、开发成本等方面具有优势，但在功能、性能、依赖性等方面存在劣势。开发者可以根据项目需求和目标用户群体，权衡利弊，选择适合的应用形式。

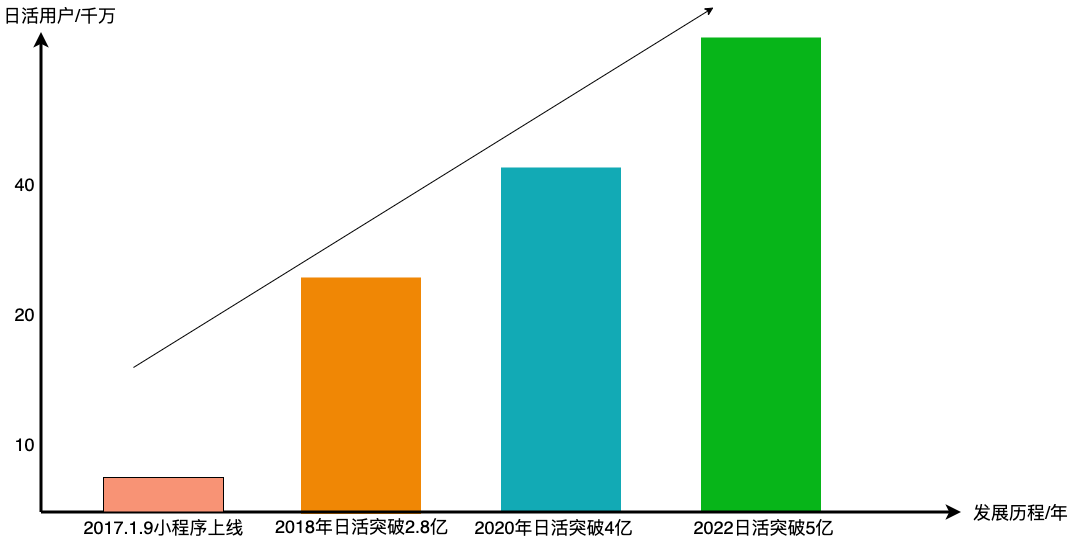


图 1‑1微信小程序日活用户增长趋势

## 1.2选题和主要研究方向

* + 1. 针对当前市场需求，本文选题为“基于React开发框架的在线音乐播放小程序的设计与实现”。本文将采用Taro多端开发框架和React语法，实现一个具有基本功能的在线音乐播放小程序。在功能设计方面，本文将重点研究以下几个方面：

首页推荐歌曲和专辑功能，为用户提供丰富的音乐选择。

搜索页面，实现音乐搜索、查看历史搜索记录以及收藏喜欢的音乐功能。

音乐列表页面，展示用户收藏的音乐，方便用户查找和播放。

用户页面，实现登录、播放历史等功能，提高用户体验。

播放器页面，实现音乐播放控制、切换音乐、音乐列表、进度条控制以及收藏功能。

通过实现以上功能，本设计将为用户提供一个具有良好体验和基本音乐播放功能的在线音乐播放小程序

# 可行性和需求分析

## 2.1可行性分析

在进行可行性和需求分析时，本设计需要从技术、市场、经济、法律等多个方面来评估项目的可行性。同时，需求分析要明确项目要实现的功能、性能和使用场景等。

技术可行性：基于React开发框架和Taro多端开发框架，本设计可以实现跨平台的音乐播放小程序。此外，利用Taro框架提供的getBackgroundAudioManager音频API，可以实现音乐播放功能，满足项目需求。

市场可行性：随着移动互联网的普及，音乐在线播放市场需求持续增长。根据提供的数据，截止到2022年，互联网小程序日活达7.8亿，微信小程序超过5亿日活用户。因此，在市场上有足够的需求和用户基础支持项目的实施。

经济可行性：开发一个基于React的音乐播放小程序相较于开发原生App，成本较低。使用Taro跨平台开发框架可以节省开发成本和维护成本，降低投资风险。

法律可行性：在开发和运营过程中，本设计需要确保遵循相关的法律法规，如著作权法、数据保护法等。需要确保音乐资源的合法性，保护用户隐私和数据安全。

## 2.2需求分析

功能需求：

首页推荐歌曲和专辑功能

搜索页面的音乐搜索、历史搜索记录和收藏功能

音乐列表页面渲染用户收藏的音乐

用户页面实现登录、播放历史等功能

播放器页面实现音乐播放控制、切换音乐、音乐列表、进度条控制和收藏功能

性能需求：

音乐播放质量高，缓冲速度快

界面响应迅速，用户体验良好

良好的网络适应性，保证在不同网络环境下的稳定运行

使用场景需求：

用户在休闲、工作、学习等场景下，通过小程序在线收听音乐

用户通过搜索功能发现新的音乐和专辑，收藏喜欢的音乐

用户在音乐列表页面查找和播放自己收藏的音乐

用户查看播放历史，方便找到之前听过的音乐。

通过对项目的可行性和需求分析，本设计可以确定项目具有较好的实施前景。本设计旨在实现一个基于React开发框架的在线音乐播放小程序，具备基础的用户体验和音乐播放功能。在技术可行性、市场可行性、经济可行性和法律可行性方面，项目均具备相应的优势，为项目的成功实施奠定了基础。

接下来，本设计将对项目的功能和性能需求进行详细设计，以确保项目的顺利进行。在设计过程中，我们将密切关注用户体验，确保项目实现的功能能够满足用户的需求和期望。同时，本设计还将关注项目的网络适应性、安全性和稳定性等方面，以确保项目在不同网络环境下的稳定运行。

通过对项目的可行性和需求分析，本设计为项目的设计和实施奠定了基础。本设计期待在项目实施过程中，不断优化和完善功能，提高用户体验，最终为用户带来一个高质量的在线音乐播放小程序。

# 基础方案设计

## 3.1开发技术

在本设计的基础方案设计中，我们将使用React语法支持的Taro多端开发框架作为前端技术，以实现跨平台的音乐播放小程序。后端和数据库则采用微信云开发，以实现便捷的开发和部署过程。

前端技术：Taro多端开发框架 Taro是一款由京东凹凸实验室开发的多端统一开发框架，支持使用React、Vue等主流前端框架编写代码。通过Taro，我们可以将编写的代码编译成微信小程序、支付宝小程序、H5等多个平台的应用。在本设计中，我们将采用React作为前端框架，搭配Taro进行开发。这样，我们可以充分利用React的组件化开发思想和丰富的生态，提高开发效率和可维护性。

后端技术：微信云开发 微信云开发（CloudBase）是腾讯云推出的一种无服务器云应用开发服务，为开发者提供了完整的后端服务和基于云端的开发环境。通过微信云开发，我们可以快速地搭建一个具备数据库、云存储、云函数等能力的后端服务，无需关注服务器运维和底层实现。 在本设计中，我们将使用微信云开发作为后端技术，搭建音乐播放小程序的后端服务。这样，我们可以将更多精力投入到功能开发和优化上，提高项目的整体质量。

数据库：微信云数据库 微信云数据库是微信云开发的一部分，为开发者提供了一个稳定、安全、高效的云端数据库服务。在本设计中，我们将使用微信云数据库存储音乐信息、用户收藏、播放历史等数据。通过微信云数据库，我们可以轻松地实现数据的增删改查操作，满足项目的数据存储需求。

综上所述，在本设计的基础方案设计中，我们将采用React语法支持的Taro多端开发框架作为前端技术，微信云开发作为后端技术，以及微信云数据库作为数据存储 方案。通过这一设计，我们将实现一个具备良好用户体验和音乐播放功能的在线音乐播放小程序。数据流图是一种用于描述系统内部数据流动、处理和存储情况的图形。在本设计中，数据流图将帮助我们更好地理解小程序的数据流动和交互情况，接下来是实现小程序的数据流图:

图形用户界面

描述已自动生成

图3-1 小程序数据流图

用户：用户通过微信小程序界面与系统进行交互，包括查看歌曲、播放歌曲、收藏歌曲等操作。

Taro应用：基于React和Taro开发的前端应用，负责接收用户的操作请求，并将结果展示给用户。

微信云开发：接收来自Taro前端的请求，处理业务逻辑，与微信云数据库进行交互，并将结果返回给前端。

微信云数据库：存储所有的音乐信息、用户收藏、播放历史等数据，接收来自微信云开发后端的数据操作请求，并返回操作结果。

## 3.2功能模块划分

在下面的功能模块中，我们将运用一系列的技术进行实现：

首页推荐模块：我们将利用React和Taro的组件化动态的推荐歌曲和专辑组件。组件将从微信云开发后端获取数据，展示在用户的首页上。为了优化用户体验，我们会利用React的状态管理和Taro的页面生命周期函数，使得用户在刷新或重新进入首页时，能够看到最新的推荐内容。

搜索页面模块：此模块需要实现搜索功能、历史记录以及收藏功能。我们将使用微信云开发的全文搜索功能，实现用户输入关键词后，能够从数据库中检索到相关的音乐。历史记录将在本地存储用户的搜索历史，以方便用户再次搜索。收藏功能则会将用户喜欢的音乐信息发送到后端，并存储在微信云数据库中，以实现用户在不同设备上查看自己的收藏列表。

音乐列表模块：在这个模块中，我们将设计一个音乐列表组件，展示用户收藏的音乐。组件将从后端获取数据，并利用React的列表渲染功能，将数据以友好的方式展示给用户。同时，我们会在音乐列表项上添加播放按钮，让用户可以直接从收藏列表中播放音乐。

用户页面模块：用户页面将提供用户登录和查看播放历史的功能。我们将使用微信的登录接口，实现用户的一键登录。同时，我们会在后端记录用户的播放历史，并在用户页面上展示。

播放器模块：播放器是音乐播放小程序的核心部分，它需要实现音乐播放控制、切换音乐、查看音乐列表、控制进度条和收藏功能。我们将使用微信小程序提供的背景音频管理API，实现音乐的播放、暂停、切换等功能。为了提供更好的用户体验，我们将设计一个动态的进度条，反馈当前音乐的播放进度。另外，我们将在播放器页面添加收藏按钮，让用户在享受音乐的同时，能够轻松收藏喜欢的音乐。

这些功能模块的实现，将使我们的在线音乐播放小程序具备良好的用户体验和完整的音乐播放功能。在未来的开发过程中，我们也会根据用户的反馈和市场的需求，不断优化和扩展这些功能。树形结构图如下：



图 3‑2 小程序功能树形结构图

## 3.3性能指标

在编写性能指标时，我们需要关注几个主要方面：性能、稳定性、易用性和可扩展性。

1.性能指标： a. 页面加载速度：主要页面的加载速度应在1秒内完成。同时，对于图片和其他大型媒体文件的加载，我们需要采用懒加载或预加载等技术，以优化加载速度。 b. 音乐播放延迟：音乐播放的响应时间应小于500毫秒，但同时我们也应确保在网络条件较差的情况下，应用仍能正常工作，例如通过提前缓冲等方式。 c. 搜索响应速度：搜索结果应在1秒内显示，但对于复杂的搜索查询，我们应该提供 进度反馈，让用户知道搜索正在进行中。

2.稳定性指标： a. 系统稳定性：应用的崩溃率应低于0.1%，同时我们也应有相应的错误报告机制，以便及时发现和修复问题。 b. 音乐播放稳定性：音乐播放过程中的卡顿和中断率应低于0.5%，并且在网络不稳定的情况下，应用能够优雅地处理，例如暂停播放并等待网络恢复。

3.易用性指标： a. 用户界面：用户界面应简洁易懂，操作流程清晰，但同时也应具有一定的吸引力，以提高用户的使用愿意性。 b. 功能完整性：应用应覆盖主要的音乐播放和用户体验功能，但我们也应注意不要过度复杂化，以免对用户造成困扰。

4.可扩展性指标： a. 模块化设计：代码应采用模块化设计，但同时我们也应注重各模块之间的解耦，避免修改一个模块导致其他模块的问题。 b. 系统可扩展性：应用应具备良好的可扩展性，支持后续功能的开发和集成，同时我们也应考虑到性能的可扩展性，例如通过适当的架构设计，确保应用在用户量增加时，依然能保持良好的性能。

## 3.4技术方案比较

在开始开发项目时，我们主要考虑了两个技术方案：微信原生框架和使用React语法的Taro多端开发框架。以下是我们对两种方案的比较和分析：

1.微信原生框架： 微信原生框架为开发者提供了一套丰富的API和组件，可以满足大部分小程序的开发需求。然而，微信原生框架的缺点也十分明显：一是代码不能跨平台使用，如果要开发其他平台的应用，需要重新编写代码；二是微信原生框架的学习曲线较陡峭，需要开发者熟悉微信小程序的开发规范和API。

2.使用React语法的Taro多端开发框架： Taro是一套统一的、基于React语法的多端开发解决方案，支持微信/京东/百度/支付宝/字节跳动/QQ小程序、快应用、H5、React Native等多端开发。使用Taro开发，一套代码即可适配多个平台，极大地提高了开发效率。同时，Taro遵循React语法规范，对于熟悉React的开发者来说，学习成本相对较低。不过，Taro也有一些限制，例如不能使用一些特定平台的原生API和组件，可能需要额外的插件或者工具支持。

以下是使用微信原生框架和使用React语法的Taro多端开发框架开发小程序的一个数据对比例子：

代码行数：

使用微信原生框架开发：由于需要为每个页面编写独立的.wxml, .wxss, .js和.json文件，假设我们的音乐小程序有5个主要页面（首页、搜索、音乐列表、用户、播放器），并且每个页面的代码行数平均为200行，那么总共需要编写约4000行代码。

使用Taro多端开发框架开发：由于Taro支持React语法和组件化开发，可以大大减少重复代码。假设通过组件化和代码重用，每个页面的代码行数可以减少到150行，那么总共只需要编写约1500行代码。

开发时间：

使用微信原生框架开发：由于需要学习微信小程序的专有语法和API，开发者可能需要更多的时间来熟悉和掌握。假设我们的开发团队每天能完成200行代码的开发，那么完成4000行代码需要约20个工作日。

使用Taro多端开发框架开发：由于Taro遵循React语法，对于熟悉React的开发者来说，学习成本更低，开发速度更快。假设我们的开发团队每天能完成300行代码的开发，那么完成1500行代码只需要约5个工作日。

可维护性：

使用微信原生框架开发：由于每个页面的代码是独立的，对于复杂的小程序来说，维护和更新代码可能会变得困难。

使用Taro多端开发框架开发：由于Taro支持组件化开发，使得代码的结构更清晰，易于维护和更新。

总体来说，使用Taro多端开发框架开发小程序可以提高开发效率，降低学习成本，提高代码的可维护性。虽然Taro也有一些限制，但其优点明显优于微信原生框架：一是提高了开发效率，支持一套代码适配多个平台；二是降低了学习成本，开发者使用React的人数众多，解决问题或者修bug的参考也会更多。因此，我们认为Taro是更适合我们项目的技术方案。

## 3.5项目开发周期与耗时

1.技术选型（开始于2022年12月15日，耗时约3周）： 在这个阶段，我们对可用的技术进行了全面的调研和比较，包括学习和评估各种开发框架和技术的优缺点，最终确定了使用React语法支持的Taro多端开发框架和微信云开发作为项目的技术方案。

2.项目启动（开始于2023年3月10日，耗时约1周）： 从这个阶段开始，我们正式着手进行项目的开发工作。在此期间，我们完成了需求分析、设计方案、功能模块划分等工作，并制定了项目开发计划。

3.基础功能实现（截止到2023年5月9日，耗时约2个月）： 在这个阶段，我们已经完成了项目的所有基础功能，包括首页推荐歌曲和专辑、搜索页面、音乐列表页面、用户页面和播放器页面等。此外，我们也进行了一些初步的测试和优化工作，以确保这些功能的基本稳定性和可用性。

4.持续优化测试（进行中，预计耗时约1个月）： 在这个阶段，我们将对已实现的功能进行持续的优化和改进，以提高用户体验和性能。同时，我们还将进行更深入和全面的测试，包括功能测试、性能测试、兼容性测试等，以确保功能的稳定性和可靠性。我们也会根据测试结果进行必要的修复和优化工作。

以上是对项目的开发周期和耗时的预估，实际进度可能会因各种因素（如技术问题、需求变更等）而有所调整。

# 详细设计过程与说明

## 4.1用户模块

### 4.1.1登录

我们在一个特定的React组件中实现了用户登录的功能。此组件包含以下几个主要函数：

getUserProfile函数：此函数通过调用Taro.getUserProfile方法获取用户的个人信息，并将这些信息储存在本地缓存中。如果成功获取到用户信息，系统会弹出一个提示框告知用户登录成功，并把用户信息保存到组件的状态中。若获取信息失败，系统会弹出一个提示框，要求用户重新进行授权。

resetUserId函数：该函数被用来重置用户id，它会将组件状态中的userId设为null。

outLogin函数：此函数负责处理用户退出登录的操作，它会清除本地缓存中的用户信息，并将组件状态中的userInfo和isLogin设为null和false。

setUserInfo函数：这个函数负责设置用户信息，它会将传入的user参数保存到组件的状态中。

setIsLogin函数：此函数用于设置用户的登录状态，将传入的isLogin参数保存到组件的状态中。

登录功能的实现流程如下：

用户点击登录按钮后，系统会调用getUserProfile函数。

getUserProfile函数通过调用Taro.getUserProfile方法获取用户信息。如果获取成功，系统会弹出提示框告知用户登录成功，并将用户信息保存到本地缓存和组件的状态中，同时将isLogin设置为true。

如果获取用户信息失败，系统会弹出提示框，要求用户重新授权。

当用户想要退出登录时，系统会调用outLogin函数，清除本地缓存中的用户信息，并将组件状态中的userInfo和isLogin设为null和false。

在需要重置用户id的场合，系统会调用resetUserId函数，将组件状态中的userId设为null。

下图是微信小程序登录的流程：

图片包含 图示

描述已自动生成

图 4‑1 小程序登录流程图

## 4.2首页模块

在React组件中，实现了一个名为getRecommendedData的函数，该函数负责从云数据库中获取推荐的音乐和专辑数据。

以下是函数中涉及的主要步骤：

初始化云数据库实例：首先，我们使用 Taro.cloud.database() 初始化一个云数据库实例 db。

获取数据集合：我们通过调用 db.collection("musiclist") 获取了包含歌曲数据的集合 recommendedSongsCollection 和专辑数据的集合 recommendedAlbumsCollection。在这个例子中，歌曲和专辑数据都存储在同一个集合 "musiclist" 中。

获取推荐歌曲和专辑数据：我们分别调用 recommendedSongsCollection.get() 和 recommendedAlbumsCollection.get() 从云数据库获取推荐歌曲和专辑数据。由于get() 方法返回一个 Promise，所以我们可以通过.then() 方法处理获取到的数据。在 then() 中，我们将获取到的数据（res.data）分别保存到state中的 recommendedSongs 和 recommendedAlbums 数组中。

实现首页推荐音乐的流程如下：

在组件的componentDidMount生命周期函数中，我们调用getRecommendedData函数。

在getRecommendedData函数中，我们通过Taro.cloud.database()方法初始化数据库实例db，并通过db.collection("musiclist")方法获取推荐歌曲和专辑的集合实例recommendedSongsCollection和recommendedAlbumsCollection。

我们调用recommendedSongsCollection.get()和recommendedAlbumsCollection.get()方法，从云数据库获取推荐歌曲和专辑的数据，并将这些数据保存到state中。

最后，在首页渲染过程中，我们遍历state中的recommendedSongs和recommendedAlbums数组，展示出推荐的音乐和专辑。

总的来说，以上步骤展示了如何在React组件中通过Taro框架和云数据库获取和展示推荐的音乐和专辑数据。下面是首页index实现推荐音乐的流程图:

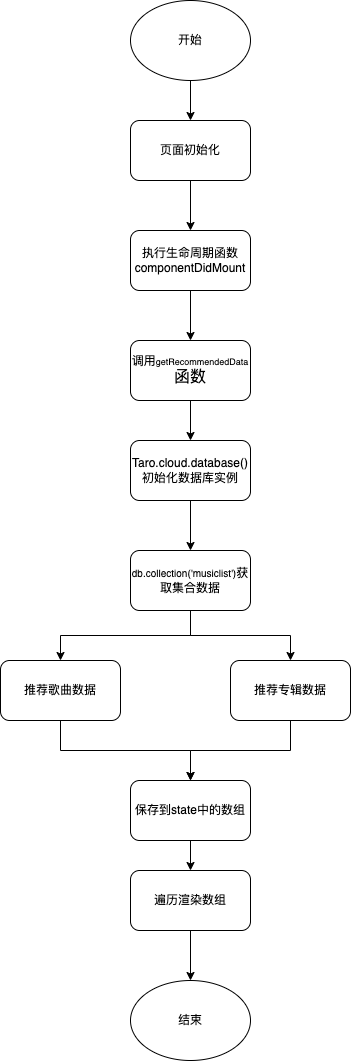


图 4‑2 首页推荐音乐的实现流程

## 4.3搜索模块

在React组件中，我们设计了多个函数来实现音乐搜索和历史记录的功能：

removeItemIfExists函数：此函数负责从列表中移除指定项，如果列表中没有这个项，则返回原始列表。

addItemToBeginning函数：此函数将指定项添加到列表的开头。

debounce函数：此函数对传入的函数进行防抖处理，防止频繁调用。

handleInput函数：此函数处理用户的输入，将用户输入的内容通过setState方法保存到组件的状态中，并利用debounce函数对callCloudFunction函数进行防抖处理。

addToSearchHistory函数：此函数负责将搜索记录添加到历史记录列表中，将最新的记录添加到列表的开头，并确保列表长度不超过5个。

playTrack函数：此函数通过Taro.getBackgroundAudioManager()方法获取背景音乐管理器实例backgroundAudioManager，并调用其相关方法实现音乐的播放、暂停以及错误处理等功能。

handleItemClick函数：此函数处理用户点击搜索结果的事件，调用playTrack函数播放音乐，并将搜索记录添加到历史记录列表中，通过setState方法更新组件的状态。

因此，音乐搜索和历史记录的实现流程如下：

用户在搜索框中输入内容，调用handleInput函数处理用户输入。

handleInput函数通过setState方法将用户输入的内容保存到组件的状态中，并使用debounce函数对callCloudFunction函数进行防抖处理。

callCloudFunction函数调用云函数获取音乐搜索结果，并将结果保存到组件的状态中。

用户点击搜索结果，此时会调用handleItemClick函数处理用户点击事件。

handleItemClick函数会调用playTrack函数播放音乐，并将搜索记录添加到历史记录列表中，然后通过setState方法更新组件的状态。

下面是搜索页面实现搜索的基本流程图:



图 4‑3 搜索实现流程

### 4.3.1Search云函数

微信云函数作为一种无服务器（Serverless）的云计算服务，使得开发者无需关心服务器的管理和维护，只需在微信开发者工具中编写和部署代码，专注于业务逻辑的实现。

在云函数中，首先通过cloud.init方法初始化云开发环境。随后，通过db.database方法获取数据库实例db，同时使用db.command方法获取command对象。

在main函数中，我们从event中获取搜索关键字key。然后，使用aggregate方法进行聚合查询，借助match方法筛选出包含关键字的音乐记录，通过db.RegExp方法进行正则匹配，并利用\_.or方法实现“或”查询。最后，通过end方法获取查询结果，并将结果返回。

## 4.4音乐列表模块

此模块主要通过React组件实现收藏列表的获取和渲染，涉及以下几个函数：

fetchFavoriteList函数：使用Taro.cloud.database()方法从云数据库中获取收藏列表，然后存储到组件状态中。

componentWillMount函数：在组件挂载前，调用fetchFavoriteList函数获取收藏列表。

componentDidShow函数：当页面显示或切入前台时，重新调用fetchFavoriteList函数更新收藏列表。

获取和渲染收藏列表的实现流程如下：

在componentWillMount函数中，调用fetchFavoriteList函数从云数据库获取收藏列表并保存至组件状态中。

当页面显示或切入前台时，componentDidShow函数会调用fetchFavoriteList函数重新获取并保存最新的收藏列表，保证其他页面的收藏操作能够及时反映在列表中。

在render函数中，使用this.state.favoriteList.map方法遍历收藏列表并渲染每一项。

在组件卸载前，componentWillUnmount函数可以进行必要的清理操作，如取消订阅或清除定时器等。

下面是实现音乐列表页面的流程图:

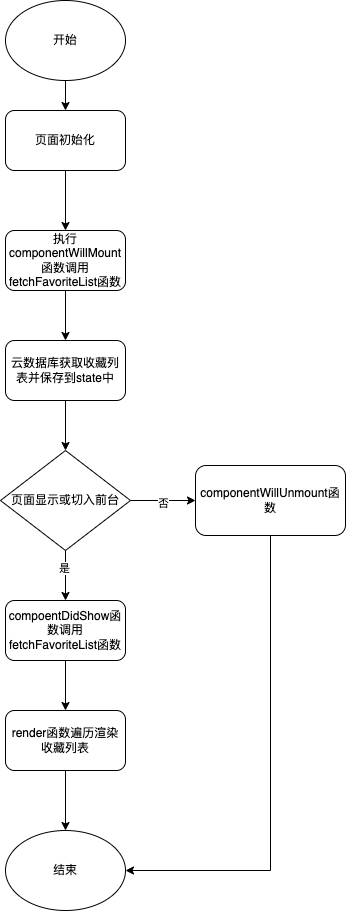


图 4‑4 音乐列表实现流程

## 4.5播放器模块

在React组件中，播放器模块实现了播放、暂停、切换歌曲和显示播放进度等功能，包含以下主要函数：

togglePlay：控制音乐播放和暂停，更新播放状态isPlaying和isUserPaused，然后根据状态决定音乐是播放还是暂停。

stateClick：切换播放状态isPlaying并输出当前状态至控制台。

handleTimeUpdate：监听音频播放进度、结束状态和总时长，实时更新播放时间、总时长等信息，音频结束后自动播放下一首。

setCustomPlaylist：设置自定义播放列表，接受一个数组参数作为当前播放列表。

playNext和playPrevious：播放下一首或上一首歌曲，计算播放索引，获取并播放新的歌曲信息。

formatTime和formatSeconds：转换秒数与"mm:ss"格式时间字符串。

handleProgress和handleProgressChanging：处理进度条改变，获取音频总时长，计算并更新当前播放时间和进度条值，音频跳转到指定播放时间。

播放器页面的功能模块实现流程如下：

初始化和更新收藏状态：在组件挂载componentDidMount和更新componentDidUpdate时，更新收藏状态。

渲染播放器页面：创建播放器容器，渲染歌曲封面、信息、播放列表按钮、播放列表、收藏按钮、进度条和播放控制按钮。

播放控制：点击播放按钮切换播放状态togglePlay，点击上/下一曲按钮播放相应歌曲（playPrevious/playNext）。

收藏歌曲：点击收藏按钮切换收藏状态。

进度条控制：创建进度条展示播放进度，监听进度条变化onChange，用户调整进度条时onChanging更新播放进度。

播放列表：点击播放列表按钮切换播放列表显示状态，遍历dataList数组创建播放列表项，点击播放列表项播放对应歌曲。



图 4‑5 播放器实现流程

## 4.6小程序背景音频

以下是通过调用Taro.getBackgroundAudioManager()来实现背景音频的步骤：

初始化云开发环境：在app.js中，利用Taro.cloud.init()进行环境初始化。

获取背景音频管理器：在componentDidMount生命周期函数内，利用Taro.getBackgroundAudioManager()获取背景音频管理器实例（backgroundAudioManager）。

监听背景音频事件：利用backgroundAudioManager.onPlay()和backgroundAudioManager.onPause()函数，分别监听背景音频播放和暂停事件。

提取云数据库中的音乐列表：在onLaunch生命周期函数中，通过Taro.cloud.callFunction()调用云函数login并利用Taro.cloud.database()获取云数据库中的音乐列表。之后，将音乐列表和当前播放索引保存到组件的状态中（this.setState({ dataList: res.data, currentPlayingIndex: 0 })）。

初始化播放器：从音乐列表中获取第一首歌曲的信息，并将其设为当前播放歌曲（item）。

背景音频事件监听：在onLaunch生命周期函数中，利用backgroundAudioManager监听背景音频的各种事件，例如onPlay（开始播放）、onPause（暂停播放）、onError（播放错误）、onWaiting（音频加载中）以及onCanplay（音频可以播放）。

暂停播放直至音频加载完成：在onCanplay事件中，使用backgroundAudioManager.pause()暂停音频播放，直至音频完全加载。

## 4.7全局数据和状态

以下是创建一个全局数据的React Context对象GlobalDataContext和一个包含状态和事件处理函数的App组件的流程：

创建全局数据 Context 对象：在 GlobalDataContext.js 文件中，我们使用 React.createContext() 创建一个全局数据的 React Context 对象 GlobalDataContext 并导出。

初始化全局状态：在App组件的state中，我们初始化全局状态，如isPlaying、dataList、playList、isPlayListVisible等。

实现事件处理函数：在App组件中，我们实现各种事件处理函数，如handleTimeUpdate、handlePlay、handlePause、handleProgress、handleProgressChanging等。

获取所有事件处理函数：在getAllFunctions函数中，我们返回一个包含所有事件处理函数的对象。

使用全局数据 Context 包裹应用：在App组件的render函数中，我们使用GlobalDataContext.Provider来包裹应用，并通过其value属性传递全局状态和事件处理函数。

以下是一个简化实现全局数据的app.js代码片段截图:

文本

描述已自动生成

图 4‑6 app.js代码截图

## 4.8云数据库

微信云数据库是微信小程序中的重要组成部分，它为开发者提供了数据库服务，使得开发者可以在小程序中存储和管理数据。微信云数据库提供了对 JSON 数据类型的原生支持，以及对数据的实时同步。这意味着，无论何时数据发生改变，微信云数据库都能够保证数据的实时更新。

在微信云数据库中，数据以 JSON 文档的形式存储在集合（collection）中。每一个 JSON 文档都是一个对象，对象中的字段可以是各种数据类型，包括数字、字符串、数组、对象、日期等。这样的数据结构提供了极大的灵活性，可以满足各种复杂的数据需求。

在云数据库的音乐数据表 "musiclist" 中，每条数据记录代表一首音乐，这些记录包含了音乐的基本信息，例如音乐的 ID、标题、歌手、专辑、封面图片 URL、以及音乐文件 URL。另外，我们在 "favorite" 表中存储了每个用户对某首音乐的收藏信息，这包括了用户的标识符和音乐的 ID。在实际应用中，我们根据这些数据表内容进行增删改查等操作，这使得在线音乐播放小程序的各种功能得以实现。

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 描述 |
| \_id | string | 数据记录ID |
| id | number | 音乐Id |
| title | string | 音乐标题 |
| singer | string | 演唱者名称 |
| epname | string | 专辑名称 |
| coverImgUrl | string | 封面图片URl |
| src | string | 音乐文件URL |
| createTime | string | 创建时间 |
| updateTime | string | 更新时间 |

表4-1 数据库表musiclist

# 小程序测试

## 5.1小程序功能测试

测试报告

项目名称：音乐小程序 测试类型：功能测试 测试周期：2023-04-10 至 2023-05-1

测试目的： 对音乐小程序的各项功能进行全面测试，以确保程序的稳定性和性能。

一、测试概述： 本次测试包括了音乐小程序的主要功能，如：

用户登录

获取推荐歌曲和专辑数据

音乐播放与暂停功能

切换歌曲功能

歌曲收藏功能

二、测试环境：

设备：iPhone XR

系统：iOS 15.5

测试工具：微信开发者工具

三、测试结果概述： 通过本次功能测试，音乐小程序在各个功能方面均表现良好，满足预期功能要求。以下是各个功能的详细测试结果。

用户登录： 测试用例：使用模拟用户 ID 登录 测试结果：成功登录，无异常

获取推荐歌曲和专辑数据： 测试用例：从服务器获取推荐歌曲和专辑数据 测试结果：成功获取数据，数据格式正确，无异常

音乐播放与暂停功能： 测试用例：点击播放按钮播放歌曲 测试结果：歌曲正常播放，点击暂停按钮后歌曲暂停，无异常

切换歌曲功能： 测试用例：在播放列表中选择另一首歌曲播放 测试结果：成功切换歌曲，无异常

添加歌曲到我的歌单功能： 测试用例：将推荐歌曲收藏 测试结果：歌曲成功收藏到音乐列表中，无异常

四、测试问题与解决： 在测试过程中，遇到以下问题及解决方案：

问题：获取推荐数据时，部分数据返回为空 解决：检查后发现是服务器端的问题，修复服务器端问题后，问题得到解决。

问题：设备上歌曲加载缓慢 解决：调整歌曲加载策略，优化性能后问题得到解决。

五、总结： 经过本次功能测试，音乐小程序在各个功能方面均表现良好，满足预期功能要求。在测试过程中遇到的问题已经得到解决，确保了应用的稳定性和性能。

电子设备的屏幕截图

描述已自动生成

图 5-1小程序多账号调试

# 总结与展望

## 6.1总结

在本项目中，我们成功地实现了一款功能丰富且易于使用的微信小程序音乐播放器。在开发过程中，我们通过自学不断积累了许多关于小程序开发的相关知识，从而提高了我们在这一领域的技能水平。

我们采用了基于React语法的Taro多端开发框架，这使得我们能够在多个平台上灵活地进行开发。此外，我们选择了微信云开发作为后台服务器，因为它提供了无缝衔接的云服务功能，不仅无需部署环境，而且能够大大降低后期维护成本。

在界面设计方面，我们参考了Apple Music的极简风格，并力求实现简洁优雅的用户体验。未来，我们将继续探索不同的设计风格，以满足更多用户的喜好。

## 6.2展望

音乐播放器小程序仍有改进的空间：

数据库查询方面：目前缺乏高效的索引支持。为了优化查询性能，我们需要为频繁查询的操作建立完整的索引，提高查询速度。

前端逻辑优化：目前各页面逻辑过于独立，我们需要编写通用的逻辑JS文件，以提高代码复用率和降低维护成本。

云服务选择：鉴于微信云开发服务已开始收费，我们计划在后期使用独立的云服务搭建后台，以降低成本并提高系统的可扩展性。

功能扩展：为了丰富播放器的功能，我们计划在未来加入歌词显示、动态封面以及高斯模糊等特性，增强用户的视听体验。

此外，我们还将关注新兴技术的发展趋势，以便在项目中应用最新的技术和方法。通过持续改进和优化，我们相信这款音乐播放器小程序将更加完善，为用户带来更好的使用体验。总之，我们将不断努力，以期为用户打造一款更符合他们需求的音乐播放器小程序。

# 致谢

经过两年半的努力学习和积累，我终于迎来了人生中的一个新起点。毕业在即，我将扬帆起航，勇敢地迈向更广阔的舞台，创造属于自己的辉煌未来。

在接受毕业设计的挑战过程中，我深刻体会到了成长的艰辛与快乐。在这两年半的学习生活里，我不仅掌握了扎实的专业知识，而且在实际操作中锻炼了自己的动手能力。毕业设计使我有机会将所学知识付诸实践，更好地为将来的工作和生活做准备。在进行毕业设计时，我从确定题目、查找资料到实际操作，全程都倾注了极大的心血。在图书馆查阅了大量关于软件工程、前端基础和微信小程序的参考资料后，我还观看了许多相关视频，锻炼了自己的思维分析、总结归纳和发现问题的能力。这些经历让我学会了独立解决问题的能力，奠定了扎实的职业基础，使我在未来能更好地适应工作环境和职场挑战。

在这里，我要衷心感谢关爱我、教育我和支持我的父母，是他们的关怀和鼓励让我有了成长的力量。同样，我要感谢何太军老师的悉心教导，使我在毕业设计的过程中不断进步。在这段时间里，我不仅锻炼了独立工作的能力，增强了自信心，还加深了与老师和同学们的深厚友谊。在创造过程中，我深刻感受到了面对挑战与困难时的坚持与勇敢，以及成功时的喜悦与满足。这些经历让我在未来步入社会和职场时充满信心，勇敢地去迎接挑战和机遇。

最后，我要向所有帮助过我的老师和同学们表示衷心的感谢和祝福。愿我们在未来的岁月里，不断追求卓越，共同创造更加美好的人生。

# 参考文献

[1] 微信官方文档. 小程序. 获取自<https://developers.weixin.qq.com/miniprogram/dev/>

[2] 刘刚. 微信小程序开发图解案例教程. 第2版. 人民邮电出版社. 2019, pp.119-125（在引用书籍时，"pp." 是个常用的缩写，表示 "pages"页数）

[3] 李骏, 边思. 微信小程序: 开发入门及案例详解. 第1版. 机械工业出版社. 2017

[4] 周洁. 微信小程序开发零基础入门. 第1版. 清华大学出版社. 2019, pp.251-260

[5] Jon Duckett. HTML&CSS设计与构建网站. 刘涛, 陈学敏译. 第1版. 清华大学出版社, pp.131-152

[6] 邓春晖, 秦映波. Web前端开发简明教程（html+css+javascript+jQuery）. 第1版. 人民邮电出版社. 2017

[7] Facebook Inc. React - A JavaScript library for building user interfaces. 获取自 [**https://reactjs.org/**](https://reactjs.org/)

[8] NervJS. Taro - 多端统一开发框架. 获取自 [**https://taro.jd.com/**](https://taro.jd.com/). 2021

[9] Robin Wieruch. The Road to React: Your journey to master plain yet pragmatic React.js. 4th Edition. RWieruch. 2019, pp.25-35

[10] Elijah Manor, Chakra UI. Taro UI - A Taro-based UI framework. 获取自 [**https://taro-ui.aotu.io/**](https://taro-ui.aotu.io/).

[11] 微信开放社区. 云开发. 获取自 [**https://developers.weixin.qq.com/miniprogram/dev/wxcloud/basis/getting-started.html**](https://developers.weixin.qq.com/miniprogram/dev/wxcloud/basis/getting-started.html).

[12] 微信开放社区. 云数据库. 获取自 [**https://developers.weixin.qq.com/miniprogram/dev/wxcloud/guide/database.html**](https://developers.weixin.qq.com/miniprogram/dev/wxcloud/guide/database.html).

[13] Alex Banks, Eve Porcello. Learning React: Modern Patterns for Developing React Apps. 2nd Edition. O'Reilly Media. 2020, pp.50-60

[14] Eric Elliott. Composing Software: An Exploration of Functional Programming and Object Composition in JavaScript. Leanpub. 2019, pp.70-85

[15] 张轩, 李振强. React全栈：Redux+React Router+Webpack实战. 第1版. 人民邮电出版社. 2017, pp.100-110

[16] Flavio Copes. The React Handbook. 获取自 [**https://www.freecodecamp.org/news/the-react-handbook-b71c27b0a795/**](https://www.freecodecamp.org/news/the-react-handbook-b71c27b0a795/).

[17] Bret Cameron. Fullstack React: The Complete Guide to ReactJS and Friends. 1st Edition. [Fullstack.io](http://Fullstack.io). 2017,