**武汉大学计算机学院**

**本科生课程设计报告**

**音乐播放器总体设计与实现**

专 业 名 称 ：计算机科学与技术

课 程 名 称 ：软件构造

指 导 教 师 一：何政 职称：教授

学 生 学 号 ：2023

学 生 姓 名 ：

二○二五 年 五 月

**郑 重 声 明**

本人呈交的设计报告，是在指导老师的指导下，独立进行实验工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本设计报告不包含他人享有著作权的内容。对本设计报告做出贡献的其他个人和集体，均已在文中以明确的方式标明。本设计报告的知识产权归属于培养单位。

本人签名： 日期：

目录

[音乐播放器总体设计与实现 1](#_Toc26691)

[1系统介绍 4](#_Toc8186)

[2 设计思路 5](#_Toc23494)

[3 关键技术 6](#_Toc3583)

1系统介绍

本项目是一款基于C# WinForms框架和SunnyUI控件库开发的本地音乐播放器软件，旨在为用户提供简洁、实用且可扩展的音乐播放体验。系统支持音乐播放控制、歌词同步显示、多维度搜索、用户账户管理和曲库管理等核心功能，同时具备良好的界面交互体验。项目主页面如图1所示。



图 1 项目主页面

系统采用分层架构设计，将业务逻辑、数据访问和用户界面分离，确保代码的可维护性和可扩展性。通过使用WinForms的现代化UI库SunnyUI，实现了美观且响应迅速的界面效果。效果如图2所示。



图 2音乐播放主界面

用户可以通过系统导入本地音频文件，自动构建曲库，并以标签或歌单的形式进行分类与管理；支持关键词搜索和标签筛选便于用户找到自己喜爱的歌曲；在播放层面，支持顺序播放、随机播放、单曲循环等多种模式（如图3所示），同时支持倍速播放和进度跳转，使用户在不同场景下均可灵活操作；歌词功能方面，系统能够自动解析并展示歌词文件，配合音乐播放进度实时滚动歌词内容，提供沉浸式的听歌体验。

系统还具备完整的用户系统功能，用户可注册账户并登录（如图3所示）使用收藏、历史记录、个性化歌单等服务；同时提供管理员权限，支持音乐资源的批量导入与管理。如图5所示.



图 3 用户登录页面 图 4 播放顺序选择

系统界面统一使用 SunnyUI 控件库构建，风格现代简洁，整体界面美观、响应迅速，适配多种用户使用习惯。通过清晰的模块划分与面向对象的编程设计，整个系统具备良好的可维护性和扩展性，能够为未来功能升级与优化打下坚实基础。



图 5 管理员导入页面

2 设计思路

本系统采用分层设计与模块化架构相结合的思路，目的是在提升系统性能与用户体验的同时，确保代码结构清晰、职责明确、便于后期维护和功能拓展。系统整体被划分为数据层、服务逻辑层、界面表示层三大层次，每一层均以单一职责为原则进行功能划分。

在数据层方面，系统采用本地数据库Mysql建立和管理音乐、用户、收藏、歌单、歌词、标签等数据表，利用实体类与数据库表一一对应的方式封装数据访问逻辑。

在业务逻辑层中，设置多个服务类（如播放服务、用户服务、搜索服务等），负责处理与数据交互、状态维护和业务决策逻辑，最大程度减轻界面层负担，也方便团队成员进行开发设计。

界面表示层通过 SunnyUI 框架构建，保证页面美观简洁又不失娱乐性。并结合事件机制实现用户操作与后端逻辑的高效响应，极大降低延迟，提升整体交互流畅度。

系统还设计了用户权限管理与模块隔离机制。普通用户可使用播放、搜索、收藏等基础功能，而管理员拥有额外的曲库导入与编辑权限，保证系统操作的安全性与合理性。同时用户密码采取加密处理，以密文存于数据表中，极大保证用户隐私。在类设计上，充分使用面向对象的继承与封装机制，实现用户类、歌单类、播放列表类等核心结构的复用与扩展。同时，为提高系统响应速度与使用体验，部分功能采用异步处理（如导入歌曲、搜索任务等），避免界面卡顿。整套设计确保系统既能满足当前需求，又具备良好的扩展能力。

3 关键技术

本系统在开发过程中综合运用了多种关键技术与编程思想，以确保软件在功能性、稳定性和用户体验等方面的高质量表现。主要技术如下：

**面向对象编程**：采用 C# 语言的面向对象特性，将系统功能划分为多个类与模块，诸如 Song、User、Playlist、Lyrics 等实体类，使数据结构清晰，职责明确，便于维护与扩展。

**WinForms 窗体应用开发**：系统整体基于 WinForms 框架构建，实现窗口级图形用户界面交互，通过控件事件驱动方式处理用户输入，支持多窗体、多控件协作。

**数据库管理与 MySQL 连接**：系统采用 MySQL 作为后端数据库，存储用户信息、歌曲元数据、歌词、标签、歌单等内容，配合 ADO.NET 及 LINQ 实现数据库的增删查改操作。

**LINQ 查询技术**：在对本地数据集合（如搜索、筛选、标签分类）进行处理时，使用 LINQ 表达式简化查询逻辑，提高代码可读性与效率。

**正则表达式**：用于解析歌词（.lrc）文件、提取时间戳与歌词行，提升文本处理的灵活性，亦用于验证用户名、密码等格式的合法性。

**文件操作技术**：系统支持本地音频文件与歌词文件的导入、导出及路径解析，使用 System.IO 实现对文件的读取、写入与复制功能，并封装为通用工具类供调用。

**异常处理机制**：关键模块均实现完善的异常捕获与日志记录机制，防止界面崩溃，提升系统稳定性。

**异步编程模型（Async/Await）**：使用异步方法优化音乐导入、搜索、播放加载等耗时操作，提升应用响应速度，避免界面卡顿，保障用户操作流畅性。

**多媒体音频处理**：借助 NAudio 和其他音频库，实现对 MP3、WAV 等格式文件的播放控制，包括放、暂停、跳转、音量调整、播放模式切换等操作。