# Media Player And Management

**Mini-Project Report of**

**Object Oriented Programming through Java**

***Submitted by***

**N060 Yash Dukhande**

***Under the Guidance Of***

**PROF. Sahil Mehta**

***In partial fulfillment for the award of the degree of***

**MBA Tech.**

**COMPUTER ENGINEERING**

**At**

**MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMENT & ENGINEERING**

NMIMS (Deemed –to-be University)

JVPD Scheme Bhaktivedanta Swami Marg, Ville

Parle (W), Mumbai-400 056.

**2023-2024**

## Table of Contents

## Problem Statement and Scope

### Problem Statement:

The project aims to develop a music streaming application similar to Spotify using Java programming language. The primary goal is to implement essential functionalities such as playing, pausing, stopping, and managing playlists. This project is intended to demonstrate proficiency in object-oriented programming principles and Java application development.

### Scope:

The scope of the project encompasses the following key aspects:

**Basic Functionality:**

Implementation of core functionalities including playing, pausing, and stopping songs.

Management of playlists, allowing users to add, remove, and reorder songs within playlists. **User Interface:**

Development of a user interface (either graphical or command-line) to interact with the application.

Interface should provide intuitive controls for playback and playlist management.

**Song Management:**

Ability to store song metadata such as title, artist, and duration.

Support for loading songs from external sources or adding them manually.

**Error Handling:**

Implementation of error handling mechanisms to manage exceptions during runtime.

Graceful handling of unexpected inputs or invalid operations.

**Scalability and Extensibility:**

Design of classes and components to be modular and easily extensible.

Consideration for potential future enhancements such as user authentication, advanced playback features, and integration with external APIs.

**Testing and Documentation:**

Conducting thorough testing to ensure correctness and reliability of the application.

Creation of comprehensive documentation including code comments and user guides.

The project will not cover advanced features such as user authentication, recommendation algorithms, or integration with external APIs like music databases. Focus will be primarily on implementing a functional prototype demonstrating core music playback and playlist management capabilities.

## Project Synopsis

### Objective:

The primary objective of the project is to develop a music player application in Java that provides basic functionalities similar to popular music streaming platforms like Spotify. The application allows users to play, pause, stop, and manage playlists. It serves as an educational exercise to demonstrate proficiency in object-oriented programming principles and Java application development.

### Key Features:

**Playback Controls:**

Play: Start playback of selected audio file.

Pause: Pause the currently playing audio.

Stop: Stop playback and reset to the beginning of the track.

Loop: Option to continuously loop the currently playing track.

Playlist Management:

Add: Add audio files to the playlist.

Remove: Remove selected audio files from the playlist.

Next/Previous: Navigate through the playlist and play the next or previous track.

**User Interface:**

Graphical user interface (GUI) using Swing framework for intuitive interaction.

Text field to display the path of the currently playing audio file.

Buttons for playback controls and playlist management.

**Scope:**

The scope of the project encompasses the development of a functional prototype of a music player application with the aforementioned features. The application provides a user-friendly interface for seamless playback and management of audio files. It does not include advanced functionalities such as user authentication, recommendation algorithms, or integration with external APIs.

**Deliverables:**

Java source code implementing the music player application.

Report documenting the project objectives, design, implementation details, and challenges encountered during development.

User guide providing instructions for running the application and utilizing its features.

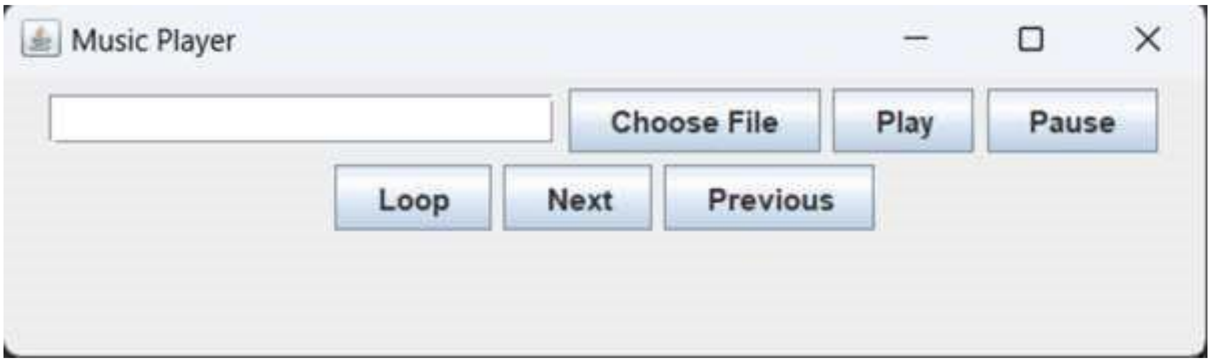
**Technologies Used:**

Java programming language for application development.

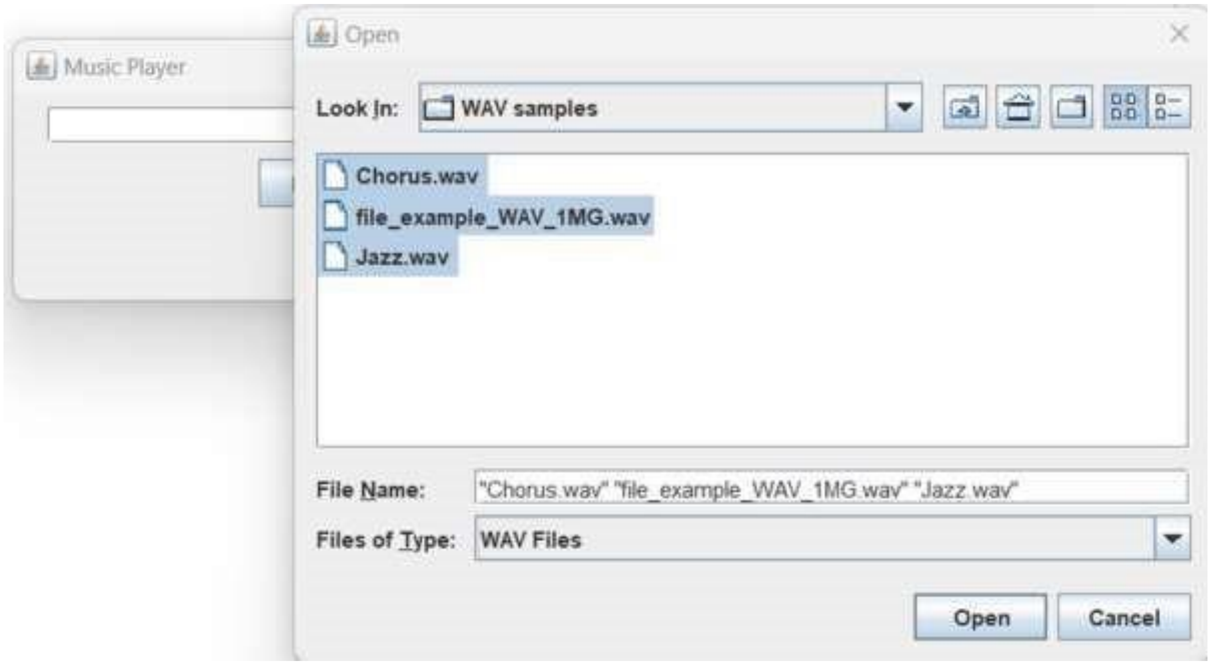
Swing framework for GUI development.

Java Sound API for audio playback functionality.

## Results



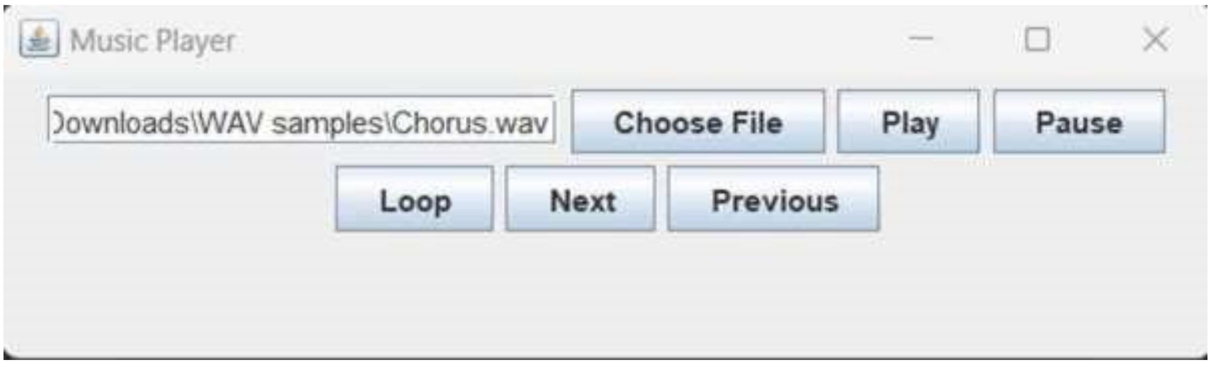
## Adding files into the playlist



File Name: "Chorus.wav" "file_example_WAV_1MG.wav" "Jazz.wav"

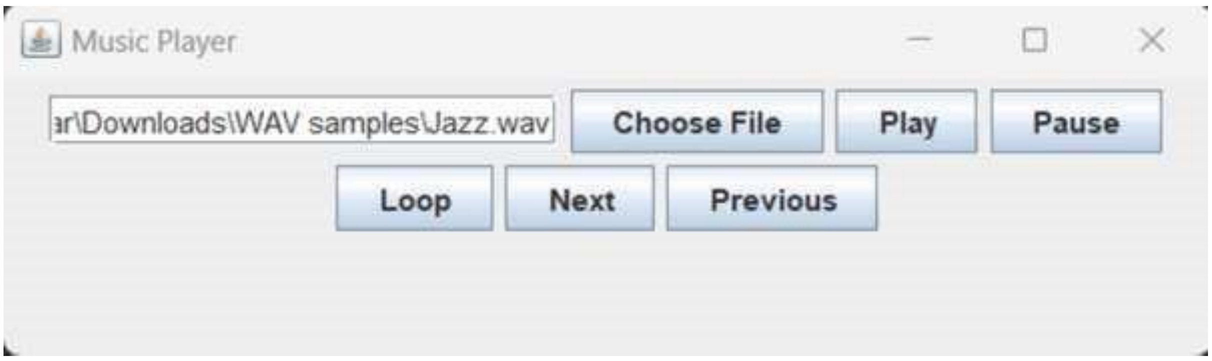
Files of Type: WAV Files

## Currently Audio file is running

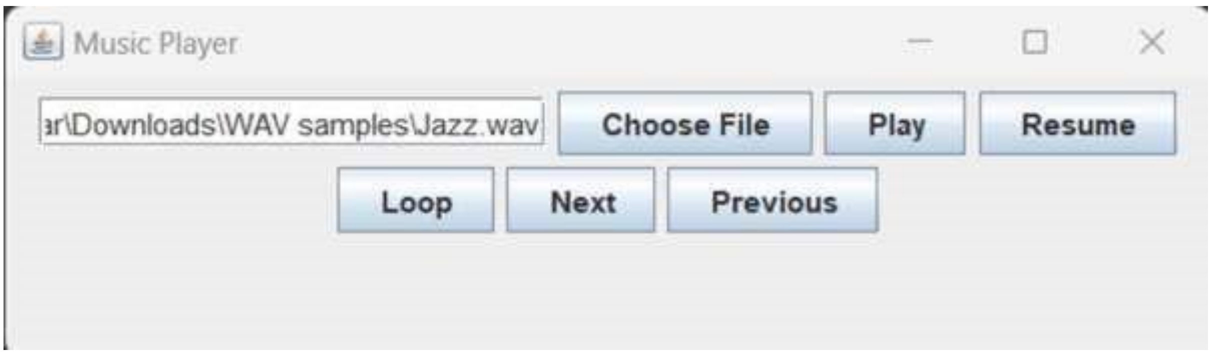


)ownloads\WAV samples\Chorus.wav

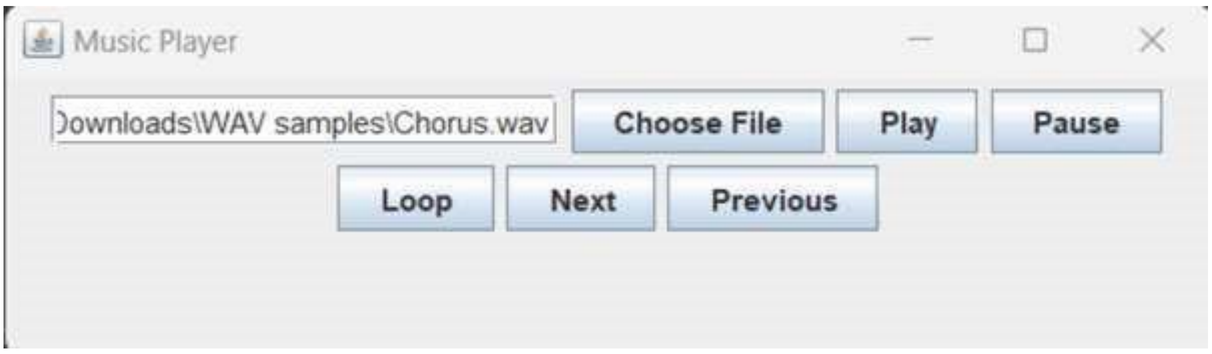
**On clicking next button the next audio file in the playlist is loaded**



**On clicking pause button the audio is stopped and the label of the button changes to Resume**



**Similar behavior of 'previous' button as of 'next' button, in this case instead of next song being played, the previous song is played**



## Conclusion

The development of the Java Music Player Application has been a valuable learning experience in the field of software development and object-oriented programming. Through this project, I have gained practical insights into various aspects of Java programming, GUI development, and audio processing.

**Key Achievements:**
Successfully implemented core functionalities including playback controls, playlist management, and user interface interactions.
Demonstrated proficiency in object-oriented programming principles such as encapsulation, inheritance, and polymorphism.
Utilized Java Swing framework to create an intuitive and user-friendly graphical interface for the music player application.
Leveraged Java Sound API for audio playback functionality, enabling seamless playback of audio files.

**Lessons Learned:**
Enhanced my understanding of event-driven programming and event handling mechanisms in Java Swing.
Explored techniques for handling audio files and managing playback using the Java Sound API.
Developed skills in debugging, troubleshooting, and addressing challenges encountered during software development.

**Future Directions:**
While the current version of the Java Music Player Application meets the basic requirements outlined in the project scope, there are opportunities for further enhancements and refinements. Potential areas for future development include:

**Advanced Features**: Integration of advanced features such as user authentication, personalized playlists, and recommendation algorithms to enhance the user experience.

**User Interface Improvements:** Refinement of the graphical user interface to improve aesthetics, usability, and accessibility for a wider range of users.

**Performance Optimization:** Optimization of the application's performance to ensure smooth playback of audio files and efficient management of large playlists.

**Compatibility and Portability:** Testing and optimization for compatibility across different operating systems and platforms to maximize reach and accessibility.

## References

https://www.geeksforgeeks.org/java-io-input-output-in-java-with-examples/

https://docs.oracle.com/en/java/javase/11/docs/api/java.desktop/javax/sound/sampled/packagesummary.html#:~:text=Package%20javax..playback%20of%20sampled%20aud io%20data.

https://www.geeksforgeeks.org/java-swing-jfilechooser/
https://www.w3schools.com/java/java_arraylist.asp#:~:text=The%20ArrayList%20class%20is%20a,to%20create%20a%20new%20one).