MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION, MUMBAI

A PROJECT REPORT ON

"The Music Player "

Submitted By

Name Of Student	Enrollment No
Purushottam Tapase	**2115460158**
Mayur Garkhede	**2115460159**
Purushottam Choudhari	**2115460160**
Prathmesh Solanke	**2115460166**

Under the Guidance of

Prof. D. L. Deshmukh

Dept. of Computer Science & Engineering
In partial fulfillment for the award of

Diploma in Computer Science & Engineering

Department Of Computer Science & Engineering
Masyodari Shikshan Sanstha's

COLLEGE OF ENGINEERING AND TECHNOLOGY[POLYTECHNIC WING]
Jalna

CERTIFICATE

This is to certify that Purushottam Tapase, Mayur Garkhede, Purushottam Choudhari, Prathmesh Solanke of Fifth Semester of **Diploma in Computer Science & Engineering** of Institute, **Matsyodari Shikshan Sanstha's Collage of Engineering & Technology** has completed the Micro Project satisfactorily in subject "Client Side Scripting(JavaScript Language)" **(22519)**for the academic year **2023- 2024** as prescribed in the curriculum.

Date: ……………………… Place: Jalna.

Prof. D.L.Deshmukh	**Prof. A. S. Kamble**
Guide	**Head of department**
Department of Computer Science & Engineering	**computer science & Engineering**

Dr. S. K. Biradar
Principal
MSS's College of Engineering and Technology, Jalna

ACKNOWLEDGEMENT

We would like to express our guider prof. A. S. Kamble Sir. Head of department Computer Science & Engineering department, MSS’s Polytechnic, Jalna who as guide us of The Music Player project.

Finally, we would like to thank our project managers and mentors who provided guidance and support throughout the project, ensuring that it was delivered on time and to the highest standards.

We sincerely thank Dr. S. K. Biradar sir, principal, MSS’s polytechnic, jalna for their continous encouragement and active interst in my progress that they gave throughout the work.

Name Of Student	**Enrollment No**
Purushottam Tapase	**2115460158**
Mayur Garkhede	**2115460159**
Purushottam Choudhari	**2115460160**
Prathmesh Solanke	**2115460166**

Approval Sheet

This Project report entitled **“The Music Player”** submitted by Purushottam Tapase, Mayur Garkhede, Purushottam Choudhari, Prathmesh Solanke, is approved for the Diploma in Computer Science and Engineering of Maharashtra State Board of Technical Education, Mumbai.

Examiner

____

____

Guide

____

____

Date: ________________

Place: ________________

DECLARATION

We hereby declare that we have form completed & return that dissertation entitled **“The Music Player”** it has not been submitted for the basis of the award of any diploma or other similar title of this any other diploma examination body/university.

Place: MSS’s CET, Jalna

Date:___________

Abstract

The "Music Player App using Java" is an innovative multimedia application that offers music enthusiasts a convenient and user-friendly platform for managing and enjoying their music collections. Developed using the Java programming language and leveraging the JavaZoom JLayer library for audio playback, this project aims to simplify the process of selecting, playing, and organizing audio files.

The app's primary objectives include providing a visually appealing and intuitive user interface that facilitates the selection of audio files and controlling playback. With features for play, pause, and stop, users can easily navigate and manage their music experience. Basic playlist functionality allows for the organization of favorite tracks and seamless transition between songs.

In summary, the "Music Player App using Java" project presents a foundational music player application with a strong focus on user experience and simplicity. It provides a solid starting point for future improvements and features, making it a valuable tool for music enthusiasts seeking to streamline their music listening and management activities.

1.Brief Introduction

Java is a class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let application developers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle) and released in 1995 as a core component of Sun Microsystems' Java platform. The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun had relicensed most of its Java technologies under the GNU General Public License. Oracle offers its own HotSpot Java Virtual

Machine, however the official reference implementation is the OpenJDK JVM which is free open source software and used by most developers and is the default JVM for almost all Linux distributions.

Music Player lets you manage all your music files quickly and easily . This audio player supports almost all types of mp3 , midi ,wav , flac raw aac files and other audio formats . Easily browse and play music songs by genres , albums , artists , songs and folder. Media is integral parts of our lives. We are creating media player using java to handle all the music requirements of user.

2. System Architecture

The system architecture of the "Music Player App using Java" project consists of the following key components:

User Interface

The user interface is developed using Java Swing, a powerful and versatile GUI toolkit for Java applications. It provides a visually appealing and interactive platform for user interaction. The interface includes elements such as file selection dialogs, playback controls, and playlist management.

Audio Playback

Audio playback is implemented using the JavaZoom JLayer library, a widely recognized and efficient solution for decoding and playing audio files in various formats. JLayer supports formats like MP3 and provides smooth and high-quality playback. This component is responsible for decoding audio data and rendering it to the user.

Data Storage

Data storage and management are facilitated through file handling. The application utilizes file I/O operations to handle audio files. Users can select audio files from their local storage, and the app manages these files for playback and organization. While this version of the app doesn't employ a traditional database for audio file storage, it effectively manages files and their metadata.

User Controls

The user controls component consists of interactive elements, such as buttons for play, pause, and stop functionality. These controls enable users to manage the playback of audio files seamlessly. Play, pause, and stop buttons provide essential functionality for controlling the music playback experience, enhancing user satisfaction.

The well-defined system architecture ensures that the application operates cohesively, offering an intuitive and efficient music playback platform. The synergy between these components results in a user-friendly and functional music player application.

This architecture allows for the easy integration of additional features and enhancements, making it adaptable to future development and user requirements.

3.Implementation Details

The implementation of the "Music Player App using Java" project involves several key aspects, including the setup of the Java application, integration of the JavaZoom JLayer library for audio playback, and the handling of user actions:

Java Application Setup

The project is developed in Java, a versatile and widely used programming language. The application is structured according to Java best practices and object-oriented principles. It follows a modular and organized approach to ensure maintainability and extensibility. Java provides a robust foundation for building a platform-independent music player.

Integration of JavaZoom JLayer Library

The core of audio playback functionality is achieved through the integration of the JavaZoom JLayer library. This library is recognized for its efficient decoding of audio files in various formats, making it an excellent choice for audio playback in the project.

JLayer ensures high-quality audio rendering and supports popular formats such as MP3.

User Interaction and Controls

The project's codebase is responsible for handling user actions and interactions. It implements event-driven programming to respond to user input. Key user actions include play, pause, stop, and file selection. The code allows users to select audio files from their local storage, play them, pause the playback, and stop the audio. It also provides error handling to address issues related to file loading and playback.

The codebase is well-structured, modular, and follows coding best practices, which contributes to the project's maintainability and ease of future development.

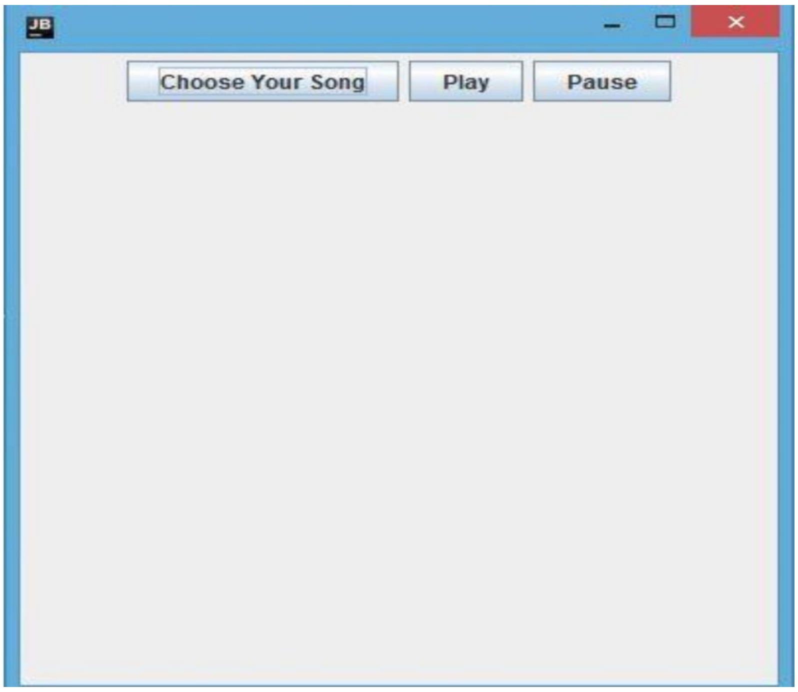
By successfully implementing these details, the "Music Player App using Java" project offers a robust and functional music player application that meets its core objectives. The integration of the JavaZoom JLayer library enhances the audio playback experience, while the codebase ensures a smooth and responsive user interface. The implementation is designed to be adaptable and extensible, allowing for potential future enhancements and features.

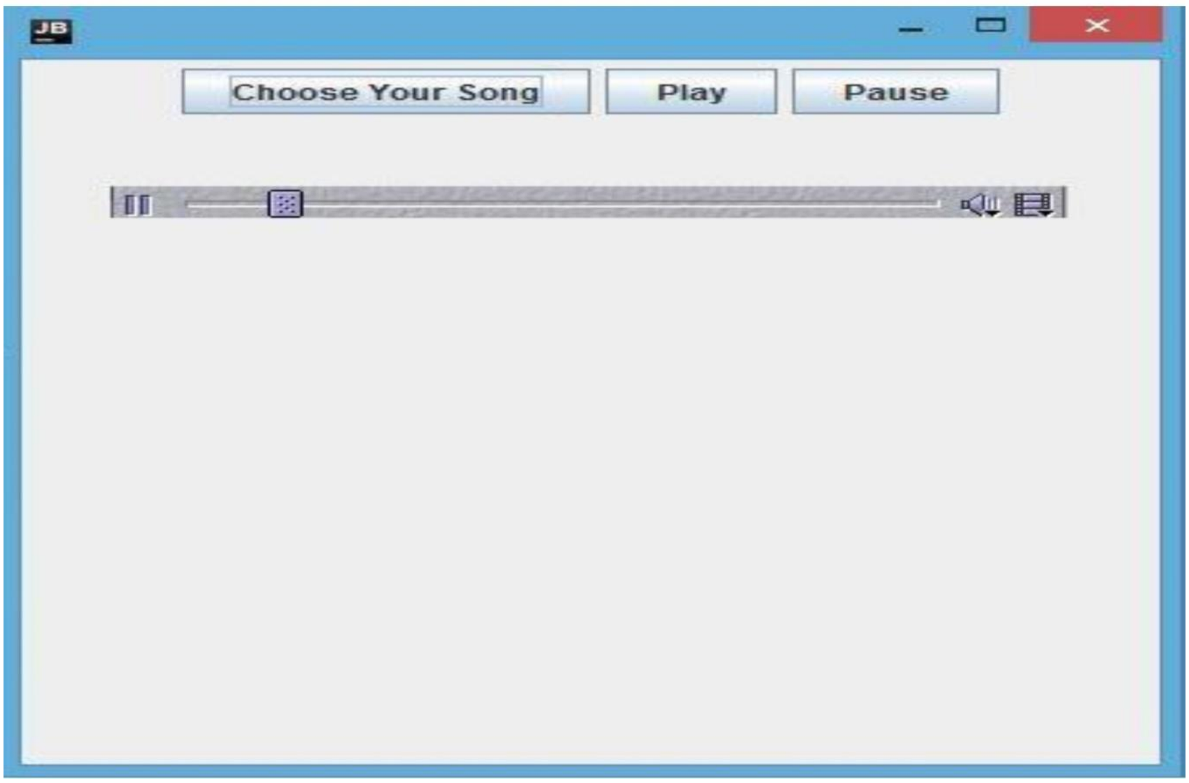
4.Source Code

```
package jlay;
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;

import javazoom.jl.decoder.JavaLayerException;
import javazoom.jl.player.Player;

public class MusicPlayer {
    Player player;
    BufferedInputStream bis;
    FileInputStream fis;
    File file;

    public MusicPlayer() {
        JFrame main = new JFrame();
        main.setVisible(true);
        main.setSize(400, 440);  // Use setSize() to set the window size
        main.setLayout(new FlowLayout());
        main.getContentPane().setBackground(Color.MAGENTA);     // Use
getContentPane() to set the background color
```

```
JButton choose = new JButton("Choose Your Song");
choose.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent actionEvent) {
        openDialog();
    }
});
main.add(choose);

JButton play = new JButton("Play");
play.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent actionEvent) {
        try {
            player.play();
        } catch (JavaLayerException e) {
            e.printStackTrace();
        }
    }
});
main.add(play);

JButton pause = new JButton("Pause");
pause.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent actionEvent) {
        if (player != null) {
            player.close();
        }
    }
    });
    main.add(pause);

    main.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
}
void openDialog() {
    JFileChooser fc = new JFileChooser();
    int result = fc.showOpenDialog(null);
    if (result == JFileChooser.APPROVE_OPTION) {
        try {
            file = new File(fc.getSelectedFile().getAbsolutePath());
            fis = new FileInputStream(file);
            bis = new BufferedInputStream(fis);
            try {
                player = new Player(bis);
            } catch (JavaLayerException e) {
                e.printStackTrace();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        MusicPlayer player = new MusicPlayer();
    });
} }
```

5.User Interface





Thank You