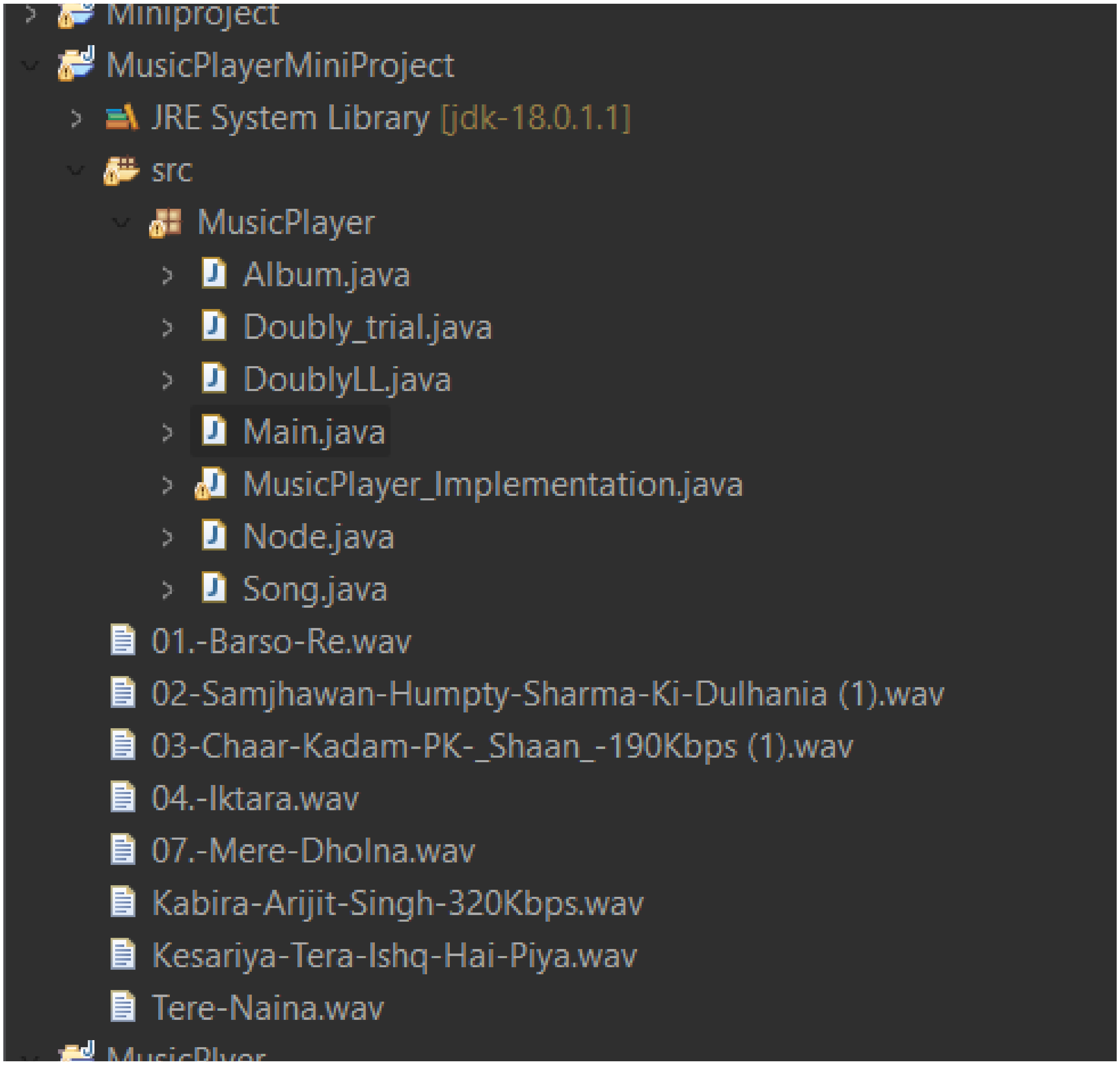
CODE

S1 : Create classes as Show in the picture below



```
> Miniproject
v MusicPlayerMiniProject
    > JRE System Library [jdk-18.0.1.1]
    v src
        v MusicPlayer
            > Album.java
            > Doubly_trial.java
            > DoublyLL.java
            > Main.java
            > MusicPlayer_Implementation.java
            > Node.java
            > Song.java
    01.-Barso-Re.wav
    02-Samjhawan-Humpty-Sharma-Ki-Dulhania (1).wav
    03-Chaar-Kadam-PK-_Shaan_-190Kbps (1).wav
    04.-Iktara.wav
    07.-Mere-Dholna.wav
    Kabira-Arijit-Singh-320Kbps.wav
    Kesariya-Tera-Ishq-Hai-Piya.wav
    Tere-Naina.wav
v MusicPlyer
```

S3 : Copy the code for respective class

Album :

```
package MusicPlayer;
```

```
import java.util.*;
public class Album {                              //USER DEFINED CLASS ALBUM
        String albumName;
        String artistName;                            //DECLARING VARIABLES
        String creationDate;
        ArrayList<Song> songList;                         //CREATING ARRAY LIST

        Album(String albumName, String artistName, String creationDate){
//CONSTRUCTOR OF CLASS ALBUM
                this.albumName = albumName;
                this.artistName = artistName;
                this.creationDate = creationDate;
                songList = new ArrayList<>();
        }

        void insertSongInAlbum(Song song) {                          //CREATING A
METHOD TO INSERT SONG IN ALBUM
                int flag = 0;
                for(Song s : songList) {                              //ENHANCED FOR
LOOP
                        if(s.getTitle().equals(song.getTitle())) {
                                System.out.println("Song is already present in the Album :
)");            //CHECKING WHETHER SONG ALREADY EXISTS IN THE LIST
                                flag = 1;
                                break;
                        }
                }
                if(flag == 0) {
                        songList.add(song);                    //IF SONG IS NOT PRESENT
THEN ADDING THE SONG IN THE LIST
                }

        }

        void removeSongFromAlbum(Song song) {                    //METHOD TO
REMOVE SONG FROM THE ALBUM
                int flag = 0;
```

```
		for(Song s : songList) {				//ENHANCED FOR LOOP
			if(s.getTitle().equals(song.getTitle())) {
				songList.remove(song);
				System.out.println("Song deleted from the album ");
//DELETING THE SONG FROM THE ALBUM
				flag = 1;
				break;
			}
		}
		if(flag == 0) {
			System.out.println("Song not found :(");		//PRINTING
SONG NOT FOUND IF THE SONG IN NOT IN THE LIST
		}
	}

	@Override
	public String toString() {				//INBUILT METHOD IN
JAVA TO RETURN THE STRING
		return "Album [albumName=" + albumName + ", artistName=" +
artistName + ", creationDate=" + creationDate
				+ ", songList=" + songList + "]";
	}


}
```

Doubly_trial :

```
package MusicPlayer;

public class Doubly_trial {				//USER DEFINED CLASS
	public static void main(String[] args) {
		DoublyLL d = new DoublyLL();			//CREATING OBJECT
OF Doubly_trial
		Song s = new Song("Mood", "5:00", "Arijit");
		Song s1 = new Song("Fantasy", "4:00", "Mona");
		Song s2 = new Song("Chisel", "6:00", "Taylor");
		d.add(s);			//ADDING SNGS AND PRINTING IT
		d.add(s1);
		d.add(s2);
		d.print();

		System.out.println(d.getHead().songInfo);			//PRINTS
THE INFORMATION OF THE SONG
	}

}
```

DoublyLL :

```
package MusicPlayer;

public class DoublyLL {				//CREATING USER DEFINED CLASS
DOUBLY LL
	Node head;
	int count;				//DECLARING COUNT VARIABLE

	int size() {				//METHOD TO RETURN THE SIZE OF
PLAYLIST
		return count;
	}

	boolean isEmpty() {			//METHOD TO CHECK WHETHER
PLAYLIST IS EMPTY
		return size() == 0;
	}

	Node getHead() {			//METHOD TO RETURN HEAD
		return head;
	}

	void add(Song song) {			//METHOD TO ADD THE SONG
		if(isEmpty()) {
```

```
                    head = new Node(song);
                    head.next = null;
                    head.prev = null;
                    count++;
            }else {
                    Node temp = new Node(song);
                    head.prev = temp;
                    temp.next = head;
                    head = temp;
                    count++;
            }
    }

    void print() {                          //METHOD TO DISPLAY THE SONGS
        Node ptr1 = head;
        if(head == null) {
           System.out.println("List is empty");          //CHECKING FOR THE
CONDITION WHETHER THE PLAYLIST IS EMPTY OR NOT
           return;
        }
        while(ptr1 != null)                         //LOOP TILL HEAD IS NOT NULL
          {
           System.out.print("Song Name : "+ptr1.songInfo.title + ", ");
//DISPLAYING THE DETAILS OF THE SONG
           System.out.print("Duration : "+ptr1.songInfo.duration + ", ");
           System.out.println("Singer Name : "+ptr1.songInfo.singerName + " ");
           ptr1 = ptr1.next;
           }
        System.out.println();
        }
}
```

Main :

```
package MusicPlayer;
import java.io.File;                        //IMPORTING JAVA CLASSES
import java.io.IOException;
import java.util.Scanner;
import javax.sound.sampled.*;

public class Main {                                    //USER DEFINED CLASS MAIN
static Scanner scanner;
 public static void MainPlaymusic(String filePath) throws
UnsupportedAudioFileException, IOException, LineUnavailableException{

          scanner = new Scanner(System.in);

  File file = new File(filePath);                              //IN BUILT FILE CLASS
  AudioInputStream audioStream = AudioSystem.getAudioInputStream(file);
  Clip clip = AudioSystem.getClip();                           //IN BUILT CLIP CLASS
  clip.open(audioStream);

  String response = "";

  while(!response.equals("Q")) {
   System.out.println("P = play, S = Stop, R = Reset, Q = Quit");
//ACCEPTING RESPONSE FROM THE USER
   System.out.print("Enter your choice: ");

   response = scanner.next();
   response = response.toUpperCase();

   switch(response) {
    case ("P"): clip.start();                       //IF USER ENTERS P : PLAY THE
SONG
    break;
    case ("S"): clip.stop();                        //IF USER ENTERS S : STOP THE
SONG
    break;
    case ("R"): clip.setMicrosecondPosition(0);        //IF USER ENTERS R :
REPLAY THE SONG
    break;
    case ("Q"): clip.close();                        //IF USER ENTERS Q : QUIT THE
SONG
    break;
    default: System.out.println("Not a valid response");              //ELSE NOT A
VALID RESPONSE
   }

  }
 }
}
```

MusicPlayer_Implementation :

```
package MusicPlayer;
import java.util.*;                                   //IMPORTING INBUILT JAVA
CLASSES
import java.io.File;
import java.io.IOException;
import java.util.Scanner;
import javax.sound.sampled.*;

public class MusicPlayer_Implementation {                          //CREATING USER
DEFINED MUSIC PLAYER IMPLEMENTATION
        static Scanner obj;
        static Main m = new Main();                                  //INSTANCE OF
MAIN CLASS
        static ArrayList<Album> albumList = new ArrayList<>();
        static ArrayList<Song> likedSongList = new ArrayList<>();
//ARRAY LIST OF LIKED SONG
        static Album a1 = new Album("Arijit's hit", "Arijit Singh", "9 Dec 2022");
//CREATING ALBUM1 OF ARJIT SINGH
        static Album a2 = new Album("Sham with Shreya", "Shreya Ghoshal", "6
Dec 2022");           //CREATING ALBUM2 OF SHERYA GHOSHAL
        public static void main(String[] args) throws
UnsupportedAudioFileException, IOException, LineUnavailableException {
                char ans;
                obj = new Scanner(System.in);
                a1.insertSongInAlbum(new Song("Kesariya-Tera-Ishq-Hai-Piya.wav",
"4:23", "Arijit Singh"));       //INSERTING SONGS IN ALBUM1
```

```
		a1.insertSongInAlbum(new Song("Kabira-Arijit-Singh-320Kbps.wav",
"4:35", "Arijit Singh"));
		a1.insertSongInAlbum(new
Song("02-Samjhawan-Humpty-Sharma-Ki-Dulhania (1).wav", "4:45", "Arijit
Singh"));
		a1.insertSongInAlbum(new Song("04.-Iktara.wav", "3:45", "Arijit
Singh"));

		a2.insertSongInAlbum(new Song("07.-Mere-Dolna.wav", "4:45",
"Shreya Ghoshal"));			//INSERTING SONGS IN ALBUM2
		a2.insertSongInAlbum(new Song("01.-Barso-Re.wav", "4:35",
"Shreya Ghoshal"));
		a2.insertSongInAlbum(new
Song("03-Chaar-Kadam-PK-_Shaan_-190Kbps (1).wav", "4:23", "Shreya
Ghoshal"));
		a2.insertSongInAlbum(new Song("Tere-Naina.wav", "3:45", "Shreya
Ghoshal"));



		albumList.add(a1);
		albumList.add(a2);

		DoublyLL playlist = new DoublyLL();			//OBJECT OF
CLASS DOUBLY LL
		playlist.add(new Song("07.-Mere-Dolna.wav", "4:45", "Shreya
Ghoshal"));			//ADDING SONGS TO THE PLAYLIST
		playlist.add(new
Song("02-Samjhawan-Humpty-Sharma-Ki-Dulhania(1).wav", "4:45", "Arijit
Singh"));
		playlist.add(new Song("Tere-Naina.wav", "3:45", "Shreya Ghoshal"));
		playlist.add(new Song("04.-Iktara.wav", "3:45", "Arijit Singh"));
		playlist.add(new Song("Kesariya-Tera-Ishq-Hai-Piya.wav", "4:23",
"Arijit Singh"));
		playlist.add(new Song("01.-Barso-Re.wav", "4:35", "Shreya
Ghoshal"));
		playlist.add(new Song("Kabira-Arijit-Singh-320Kbps.wav", "4:35",
"Arijit Singh"));
```

```
		playlist.add(new Song("03-Chaar-Kadam-PK-_Shaan_-190Kbps
(1).wav", "4:23", "Shreya Ghoshal"));

		System.out.println("--------------WELCOME TO NASA MUSIC
PLAYER-----------");
		Node ptr = playlist.getHead();
		do {								//MENU
DRIVEN PROGRAM
			System.out.println("Menu : \n1. Play song. \n2. Change song.
\n3. Replay song. \n4. List of all the songs in the playlist. \n5. Display all the
Albums available. \n6. Display songs from the selected album. \n7. Play songs
from selected album. \n8. Liked songs playlist.");
			System.out.println("\nEnter your choice : ");
			int choice = obj.nextInt();
			switch(choice) {
			case 1 :
				System.out.println("******************************PLAY
SONGS******************************");
				play(ptr, playlist);			//CALL TO PLAY
METHOD
				break;

			case 2 :
				System.out.println("******************************CHANGE
SONGS******************************");
				ptr = changeSong(ptr, playlist);		//CALL TO
CHANGE SONG METHOD
				break;

			case 3 :
				System.out.println("******************************REPLAY
SONGS******************************");
				ptr = replay(ptr, playlist);			//CALL TO REPLAY
SONG METHOD
				break;

			case 4 :
				System.out.println("******************************DISPLAY
SONG LIST******************************");
				Node ptr1 = playlist.head;
				displayAllSongs(ptr1, playlist);		//CALL TO
DISLAY ALL SONGS METHOD
				break;

			case 5 :
				System.out.println("******************************DISPLAY
ALL ALBUMS******************************");
				displayAllAlbums();				//CALL TO
DISPLAY ALL ALBUMS METHOD
				break;

			case 6 :
				System.out.println("******************************SONGS
FROM ALBUM******************************");
				songsFromAlbum();					//CALL TO
DISPLAY SONGS FROM ALBUM METHOD
				break;

			case 7 :
				System.out.println("******************************SONGS
FROM FAVOURITE ALBUM******************************");
				System.out.println("List of albums : ");
				int no3 = 1;
				for(Album a : albumList) {
//PRINTING SONGS FROM THE FAVOURITE ALBUM
					System.out.print(no3+". "+a.albumName+", ");
					System.out.print(a.artistName+", ");
					System.out.println(a.creationDate);
					no3++;
				}
				System.out.println("Enter your favourite album number: ");
				int ans1 = obj.nextInt();
				songsFromFavouriteAlbum(ans1);
				break;
```

```
                    case 8 :
                            System.out.println("******************************LIKED
SONGS******************************");
                            displayLikedSongs();
//DISPLAYING LIKED SONGS LIST
                            break;
                    }
                    System.out.println("Do you want to continue with player? (Press
'y' - yes && 'n' - no)");
                    ans = obj.next().charAt(0);
            }while(ans == 'y' || ans == 'Y');
        }

        static void play(Node ptr, DoublyLL playlist) throws
UnsupportedAudioFileException, IOException, LineUnavailableException {
                if(playlist.isEmpty()) {                                //CHECKING
WHETHER PLAYLIST IS EMPTY OR NOT
                        System.out.println("Playlist is empty. Add some songs :)");
                }else {
                        System.out.println("Song : "+playlist.head.songInfo.toString());
                        m.MainPlaymusic(playlist.head.songInfo.title);
                        System.out.println("Liked this song ? (Press 'L' - yes)");
                        char like = obj.next().charAt(0);
                        int flag = 0;
                        if(like == 'l' || like == 'L') {
                                for(Song s : likedSongList) {
//ENHANCED FOR LOOP
                                        if(s.title.equals(playlist.head.songInfo.getTitle())) {
                                                flag = 1;
                                                break;
                                        }
                                }
                                if(flag == 0) {
                                        likedSongList.add(playlist.head.songInfo);
                                }else {
                                        System.out.println("Song already exists in the liked
song playlist :)");
                                }
                        }
                }
        }

        static Node replay(Node ptr, DoublyLL playlist) throws
UnsupportedAudioFileException, IOException, LineUnavailableException {
//CREATING REPLAY METHOD
                if(ptr.prev != null) {
                        System.out.println("Song : "+ptr.songInfo.toString());
                        m.MainPlaymusic(ptr.songInfo.title);
                        System.out.println("Liked this song ? (Press 'L' - yes)");
                        char like = obj.next().charAt(0);
                        int flag = 0;
                        if(like == 'l' || like == 'L') {
                                for(Song s : likedSongList) {
//ENHANCED FOR LOOP
                                        if(s.equals(playlist.head.songInfo)) {
                                                flag = 1;
                                                break;
                                        }
                                }
                                if(flag == 0) {                         //CHECKING WHETHER
SONG IS ALREADY LIKED
                                        likedSongList.add(ptr.songInfo);
                                }else {
                                        System.out.println("Song already exists in the liked
song playlist :)");
                                }

                        }
                }else {
                        if(ptr.next != null) {
                                System.out.println("Song : "+ptr.songInfo.toString());
                                m.MainPlaymusic(ptr.songInfo.title);
                                System.out.println("Liked this song ? (Press 'L' - yes)");
                                char like = obj.next().charAt(0);
                                int flag = 0;
                                if(like == 'l' || like == 'L') {
                                        for(Song s : likedSongList) {
//ENHANCED FOR LOOP
                                                if(s.equals(playlist.head.songInfo)) {
                                                        flag = 1;
                                                        break;
                                                }
                                        }
                                        if(flag == 0) {
                                                likedSongList.add(ptr.songInfo);
                                        }else {
                                                System.out.println("Song already exists in the
liked song playlist :)");
                                        }
                        }
                                else {
                                        System.out.println("Previous song does not exits.");
                                }

                }
        }
                return ptr;
}

        static Node changeSong(Node ptr, DoublyLL playlist) throws
UnsupportedAudioFileException, IOException, LineUnavailableException {
//CREATING CHANGE SONG METHOD
                System.out.println("You want previous or next song (Prev - '1' &&
Next - '2') : ");
                int wish = obj.nextInt();
                if(wish == 1) {
                        if(ptr.prev == null) {
                                System.out.println("Prev song does not exists :(");
                        }else{
                                System.out.println("Song : "+ptr.prev.songInfo.toString());
                                m.MainPlaymusic(ptr.prev.songInfo.title);
                                System.out.println("Liked this song ? (Press 'L' - yes)");
                                char like = obj.next().charAt(0);
                                int flag = 0;
```

```
                    if(like == 'l' || like == 'L') {
                        for(Song s : likedSongList) {           //USING
ENHANCED FOR LOOP
                            if(s.title.equals(ptr.prev.songInfo.getTitle())) {
                                flag = 1;
                                break;
                            }
                        }
                        if(flag == 0) {
                            likedSongList.add(ptr.prev.songInfo);
                        }else {
                            System.out.println("Song already exists in the
liked song playlist :)");
                        }

                    }
                    ptr = ptr.prev;
                }
            }else {
                if(ptr.next == null) {
                    System.out.println("Playlist has finished :(");
                }else{
                    System.out.println("Song : "+ptr.next.songInfo.toString());
                    m.MainPlaymusic(ptr.next.songInfo.title);
                    System.out.println("Liked this song ? (Press 'L' - yes)");
                    char like = obj.next().charAt(0);
                    int flag = 0;
                    if(like == 'l' || like == 'L') {
                        for(Song s : likedSongList) {
                            if(s.title.equals(ptr.next.songInfo.getTitle())) {
                                flag = 1;
                                break;
                            }
                        }
                        if(flag == 0) {
                            likedSongList.add(ptr.next.songInfo);
                        }else {
```

```
                                        System.out.println("Song already exists in the
liked song playlist :)");
                                    }
                              }
                              ptr = ptr.next;
                        }
                  }

                  return ptr;
            }

            static void displayAllSongs(Node ptr, DoublyLL playlist) {
//METHOD TO DISPLAY ALL SONGS IN THE LIST
                  if(playlist.isEmpty()) {                    //CHECKING WHETHER
PLAYLIST IS EMPTY
                        System.out.println("Playlist is Empty");
                  }else {
                        while(ptr != null) {
                              System.out.println(ptr.songInfo.toString());
                              ptr = ptr.next;
                        }
                  }
            }

            static void displayAllAlbums() {               //METHOD TO DISPLAY ALL
THE ALBUMS
                  int no = 1;
                  for(Album a : albumList) {                //USING ENHANCED FOR
LOOP
                        System.out.print(no+". "+a.albumName+", ");
                        System.out.print(a.artistName+", ");
                        System.out.println(a.creationDate);
                        no++;
                  }
            }

            static void songsFromAlbum() {                      //METHOD TO
DISPLAY SONGS FROM THE SELECTED ALBUM
```

```
		System.out.println("List of albums : ");
		int no1 = 1;
		for(Album a : albumList) {
			System.out.print(no1+". "+a.albumName+", ");
			System.out.print(a.artistName+", ");
			System.out.println(a.creationDate);
			no1++;
		}
		System.out.println("Enter the album's name whose songs you want to
explore : ");		//ACCEPTING ALBUM'S NAME FROM THE USER
		String searchAlbumName = obj.nextLine();
		searchAlbumName = obj.nextLine();
		int flag = 0;
		Album collect = null;
		for(Album a : albumList) {			//USING ENHANCED FOR
LOOP
			if(a.albumName.equals(searchAlbumName)) {
				flag = 1;
				collect = a;
				break;
			}
		}
		if(flag == 1) {
			System.out.println("Songs from album "+searchAlbumName);
			for(Song s : collect.songList) {
				System.out.println(s.toString());
			}
		}else {
			System.out.println("Album not found :(");
		}
	}

	static void displayLikedSongs() throws UnsupportedAudioFileException,
IOException, LineUnavailableException {
		int no2 = 1;
		for(Song s : likedSongList) {			//USING ENHANCED
FOR LOOP
			System.out.print(no2+". "+s.title+", ");
```

```
                        System.out.print(s.duration+", ");
                        System.out.println(s.singerName);
                        no2++;
                }
                DoublyLL playlist3 = new DoublyLL();                    //OBJECT OF
DOUBLY LL CLASS
                for(Song s : likedSongList) {
                        playlist3.add(s);
                }
                Node p3 = playlist3.getHead();
                char wish2;
                do {
                        System.out.println("Menu : \n1. Play song. \n2. Replay song.
\n3. Change song.");
                        int ans2 = obj.nextInt();
                        switch(ans2) {
                        case 1:
                                play(p3, playlist3);                    //CALL TO PLAY METHOD
                                break;

                        case 2:
                                p3 = replay(p3, playlist3);             //CALL TO REPLAY
METHOD
                                break;

                        case 3:
                                p3 = changeSong(p3, playlist3);         //CALL TO
CHANGE THE METHOD
                        }
                        System.out.println("Do you want to continue with the album ?
(Press 'y' - yes)");
                        wish2 = obj.next().charAt(0);
                }while(wish2 == 'y'||wish2 == 'Y');
        }

        static void songsFromFavouriteAlbum(int ans1) throws
UnsupportedAudioFileException, IOException, LineUnavailableException {
//CREATING METHOD TO DISPLAY SONGS FROM FAVOURITE ALBUM
```

```
		switch(ans1) {
		case 1:
			DoublyLL playlist1 = new DoublyLL();		//OBJECT OF
DOUBLY LL CLASS
			for(Song s : a1.songList) {
				playlist1.add(s);
			}
			Node p1 = playlist1.getHead();
			for(Song s : a1.songList) {			//USING ENHANCED
FOR LOOP
				System.out.println(s.toString());
			}
			char wish;
			do {
				System.out.println("Menu : \n1. Play song. \n2. Replay
song. \n3. Change song.");
				int ans2 = obj.nextInt();
				switch(ans2) {
				case 1:
					play(p1, playlist1);			//CALL TO PLAY
METHOD
					break;

				case 2:
					p1 = replay(p1, playlist1);		//CALL TO
REPLAY METHOD
					break;

				case 3:
					p1 = changeSong(p1, playlist1);		//CALL TO
CHANGE SONG
				}
				System.out.println("Do you want to continue with the
album ? (Press 'y' - yes)");
				wish = obj.next().charAt(0);
			}while(wish == 'y'||wish == 'Y');
			break;
```

```
            case 2:
                    DoublyLL playlist2 = new DoublyLL();                //OBJECT OF
DOUBLY LL
                    for(Song s : a2.songList) {                         //USING
ENHANCED FOR LOOP
                            playlist2.add(s);
                    }
                    Node p2 = playlist2.getHead();
                    for(Song s : a2.songList) {                         //USING
ENHANCED FOR LOOP
                            System.out.println(s.toString());
                    }
                    char wish1;
                    do {
                            System.out.println("Menu : \n1. Play song. \n2. Replay
song. \n3. Change song.");
                            int ans2 = obj.nextInt();
                            switch(ans2) {
                            case 1:
                                    play(p2, playlist2);                        //CALL TO PLAY
METHOD
                                    break;

                            case 2:
                                    p2 = replay(p2, playlist2);                 //CALL TO
REPLAY METHOD
                                    break;

                            case 3:
                                    p2 = changeSong(p2, playlist2);             //CALL TO
CHANGE SONG METHOD
                            }
                            System.out.println("Do you want to continue with the
album ? (Press 'y' - yes)");
                            wish1 = obj.next().charAt(0);
                    }while(wish1 == 'y'||wish1 == 'Y');
            }
    }
```

```
}
```

Node :

```
package MusicPlayer;

public class Node {                         //USER DEFINED CLASS NODE
        Song songInfo;
        Node next;                          //DECLARING VARIABLES
        Node prev;

        Node(Song songInfo){                        //METHOD TO INITIALIZE SONG
INFO
                this.songInfo = songInfo;
        }
}
```

Song :

```
package MusicPlayer;

public class Song {                                 //USER DEFINED CLASS SONG
        String title;
        String duration;                                //DECLARING VARIABLES IN
CLASS SONG
        String singerName;

        Song(String title, String duration, String singerName){
//CONSTRUCTOR OF THE CLASS SONG
                this.title = title;
                this.duration = duration;
                this.singerName = singerName;
        }
        String getTitle() {                               //METHOD TO RETURN TITLE
OF THE SONG
                return title;
        }

        String getDuration() {                            //METHOD TO RETURN THE
DURATION OF THE SONG
                return duration;
        }

        String getSingerName() {                          //METHOD TO RETURN
THE SINGER NAME
                return singerName;
        }

        @Override
        public String toString() {                        //IN BUILT JAVA METHOD TO
RETURN STRING
                return "Song [title=" + title + ", duration=" + duration + ",
singerName=" + singerName + "]";
        }


}
```

INSERTING THE SONGS

STEP 1] DOWNLOAD THE SONGS IN .WAV format

STEP 2] COPY ALL THE SONGS AND PASTE IT IN THE PROJECT CREATED

STEP 3] RUN THE PROGRAM!!