

# MUSIC PLAYER APPLICATION - JukeBox

**A Mini-Project Report  
Under  
Project Workshop**

*Submitted by*

**Devansh Goel  
Kavya Jha**

*Under The Guidance Of*

**Prof Swarnlata B.**

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

**B. Tech  
IN  
Computer Science**

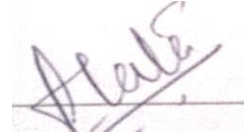
**at**

**SVKM'S MPSTME, NMIMS, MUMBAI  
April 2014**



## CERTIFICATE

This is to certify that the project entitled “Music Player Application - JukeBox” is the bonafide work carried out by Devansh Goel and Kavya Jha of B.Tech (Computer Engineering), MPSTME (NMIMS), Mumbai, during the VI semester of the academic year 2013-2014, in partial fulfillment of the requirements for the award of the Degree of Bachelors of Technology as per the norms prescribed by NMIMS. The mini-project work has been assessed and found to be satisfactory.



Swarnlata B.

Internal Mentor

---

Examiner 1

---

Examiner 2

---

Dean

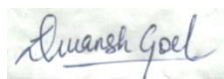
Dr. S. Y. Mhaiskar

## DECLARATION

We, Devansh Goel and Kavya Jha, Roll No., B030 and B040, respectively, of B.Tech (Computer Engineering), VI semester understand that plagiarism is defined as anyone or combination of the following:

1. Un-credited verbatim copying of individual sentences, paragraphs or illustration (such as graphs, diagrams, etc.) from any source, published or unpublished, including the internet.
2. Un-credited improper paraphrasing of pages paragraphs (changing a few words phrases, or rearranging the original sentence order)
3. Credited verbatim copying of a major portion of a paper (or thesis chapter) without clear delineation of who did wrote what. (Source: IEEE, The institute, Dec. 2004)
4. I have made sure that all the ideas, expressions, graphs, diagrams, etc., that are not a result of my work, are properly credited. Long phrases or sentences that had to be used verbatim from published literature have been clearly identified using quotation marks.
5. I affirm that no portion of my work can be considered as plagiarism and I take full responsibility if such a complaint occurs. I understand fully well that the guide of the seminar/ project report may not be in a position to check for the possibility of such incidences of plagiarism in this body of work.

Signature of the Student:



Name: Devansh Goel

Kavya Jha

Roll No. B030

B040

Place: Mumbai

Date: 23/04/2014

## **ACKNOWLEDGEMENTS**

For this mini project, the credit goes to Prof Swarnlata B., who guided and encouraged us to take this project up and helped us overcome the difficulties that we faced. We would like to thank our faculty mentor, Prof Prasashti Kanikar as well for her positive and encouraging feedback. We would also like to thank our parents for their continuous support.

## **Abstract**

In today's stressful world, entertainment has become a very integral part of any individual's life.

He seeks entertainment from various sources. Meanwhile, Smart phones and PDA's have become very dominant over the past few years. They have opened the doors to a wide range of commercial possibilities, especially, with the presence of cameras, high speed dual-core and quad-core processors, internet access, etc. Mobile devices are becoming more like PC's which also virtualizes almost everything around us. It was thought that to serve well to the individual's need for daily entertainment, a music player application should be developed.

The name of this Android Application is "JukeBox". It is a Music Player Application and the purpose of this music player is to let the users be able to listen to songs on the go. All the songs already present in the device's SD card are read and can be played. The user can have the songs segregated according to the song names, artists, albums, and the genres. The application gives the user the option of finding related YouTube videos of a particular song. The database that we would be working on is the Music Library Database, i.e., the SD card of the device on which the application is installed.

The application is developed in Java by using Eclipse IDE. The Android Software Development Kit (SDK) is used which includes a variety of custom tools that help us develop mobile applications on the Android platform. These include the Emulator and the Android Development Tools (ADT) plug-ins for Eclipse.

## **Table of contents**

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
	Abstract	5
	List of Figures	7
	Abbreviations	8
1.	INTRODUCTION	
	1.1 Project Overview	9
	1.2 Hardware Specification	9
	1.3 Software Specification	9
2.	INSTALLATION & SETUP	
	2.1 Android SDK Setup	10
	2.2 Creating an Android Application	11
3.	ANALYSIS & DESIGN	
	3.1 Functional Requirements	15
	3.2 Non-Functional Requirements	15
	3.3 Design Constraints	15
4.	PROJECT IN DETAIL	
	4.1 General Requirements	16
	4.2 General Constraints	17
	4.3 Interface	17
5.	CONCLUSION & FUTURE SCOPE	22
	REFERENCES	23
	APPENDIX	24

## **List of Figures**

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
2.	INSTALLATION & SETUP	
	Fig 2.1 Android SDK Manager	10
	Fig 2.2 Android Virtual Device Manager	11
	Fig 2.3 Setup Page for Application Name	11
	Fig 2.4 Creating Logo	12
	Fig 2.5 Creating Activity	12
	Fig 2.6 Specifying Activity and Layout	13
	Fig 2.7 Contents in Package Explorer	13
	Fig 2.8 Emulator	14
4.	PROJECT IN DETAIL	
	Fig 4.1 JukeBox on the Apps Page	17
	Fig 4.2 JukeBox Home	18
	Fig 4.3 Albums List	18
	Fig 4.4 Songs List	19
	Fig 4.5 Genres List	19
	Fig 4.6 Artists List	20
	Fig 4.7 Player	20
	Fig 4.8 YouTube through Chrome	21
A	APPENDIX	
	Fig A3.1 Use-Case Diagram	26

## Abbreviations

Abbreviation	Description
MPA	Music Player for Android
Android	The application development platform
SDK	Software Development Kit
GUI	Graphical User Interface
Qt	A cross-platform application framework used to develop applications using GUI.
DFD	Data Flow Diagram
API	Application Programming Interface
QEMU	Quick Emulator
IDE	Integrated Development Environment
ADT	Android Development Tools



# **1. Introduction**

## **1.1 Project Overview**

The report under design is for a Music Player Application called JukeBox. It contains the overall as well as the specific descriptive details of the software under consideration. It includes the result of analysis done for the project. Various techniques were used to elicit the requirements and the needs were identified, analyzed and refined. The objective of this document, therefore, is to formally describe the system's high level requirements including functional requirements, non-functional requirements, business rules and constraints. The document is organized as follows:

Section 2 of the document includes description of the product, user characteristics, general constraints and assumptions for the Music Player. This model demonstrates the development's team understanding for the product and aims to maximize the team's ability to build a system that supports the business. . It elaborates on the functional and non-functional requirements of the system and focuses on the constraints on the software design.

Section 3 presents the detailed requirements which comprise the domain model. It focuses on the interface of the final application, a detailed structuring of the product.

## **1.2 Hardware Specifications**

In this system, all interaction with the hardware is done through the API's in Android.

## **1.3 Software Specifications**

This application is developed in the Java language using the Android SDK. The SDK includes a set of development tools including a debugger, software libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. The officially supported IDE is Eclipse using the ADT plugin.

## 2. Installation and setup

### 2.1 Android SDK Setup

1) Download the latest version of Android SDK ADT Bundle from the official Android website for your operating system. The ADT Bundle provides everything you need to start developing apps, including a version of the Eclipse IDE with built-in ADT (Android Developer Tools). Also it is required that you install Java JDK before installing Android SDK.

2) Unpack the ZIP file and save it to an appropriate location.

3) Then launch the Android SDK Manager

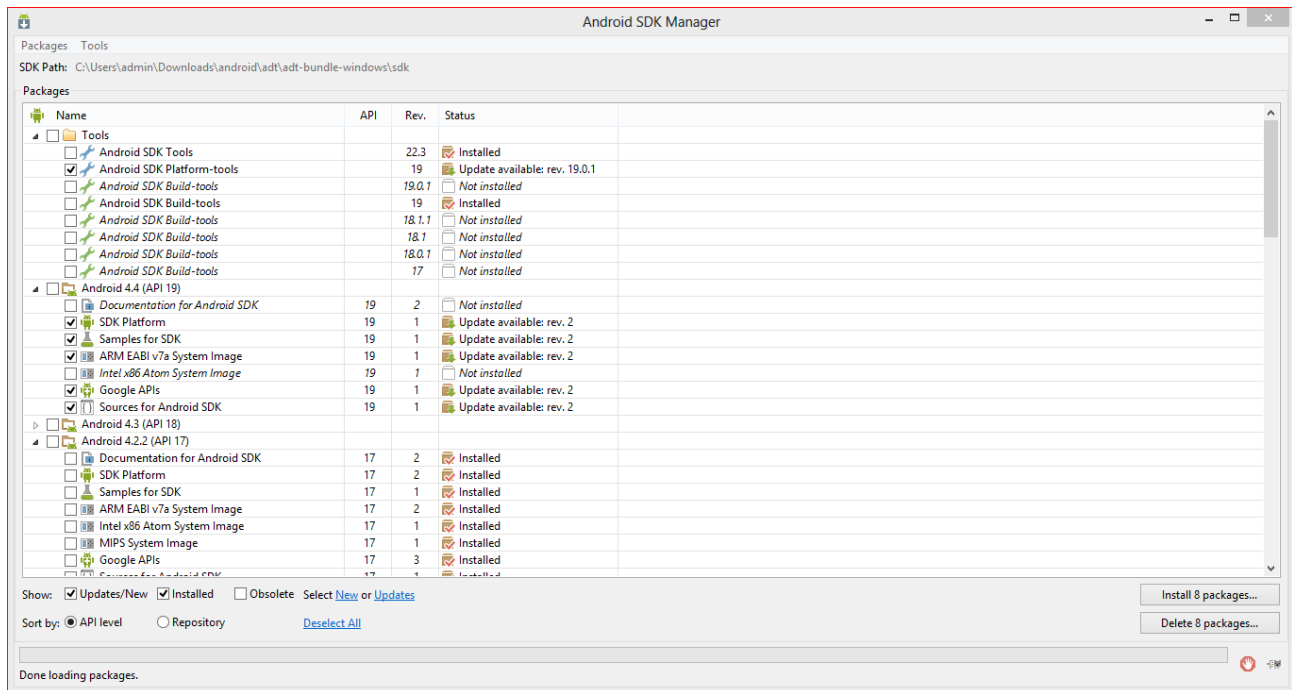


Fig2.1 Android SDK Manager

4) Once you have launched SDK manager, you should install other required packages.

5) You will have to create Android Virtual Device to test your Android applications. Open the AVD from the extracted Android SDK ADT Bundle. It will launch Android AVD Manager. Use New button to create a new Android Virtual Device with the suitable requirements. Then click on ok button to create AVD.

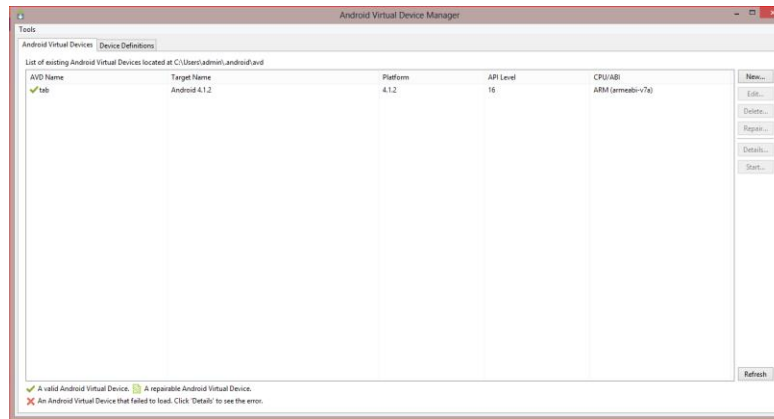


Fig2.2 Android Virtual Device Manager

## 2.2 Creating an Android Application

- 1) Open the Eclipse IDE from the extracted Android SDK ADT bundle. Then click on File -> New -> Project. Then select Android Application Project from the list. Then fill in the form details as required and then click on next screen.

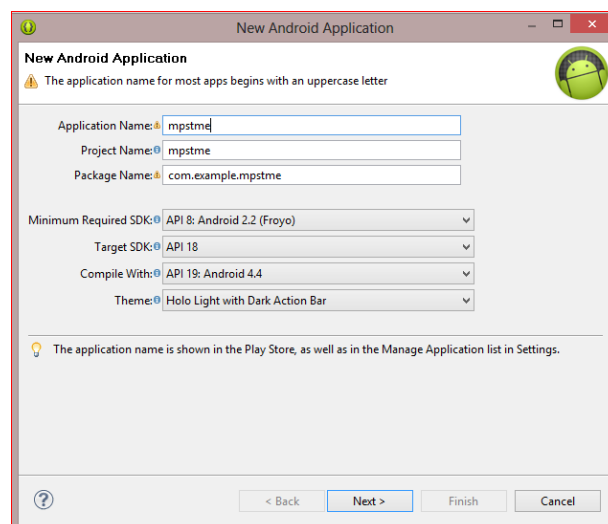


Fig2.3 Setup Page for Application name

- 2) For now we can keep the default settings. Then click on next.
- 3) This screen allows you to specify the Launcher Icon. After choosing the icon, click on next.

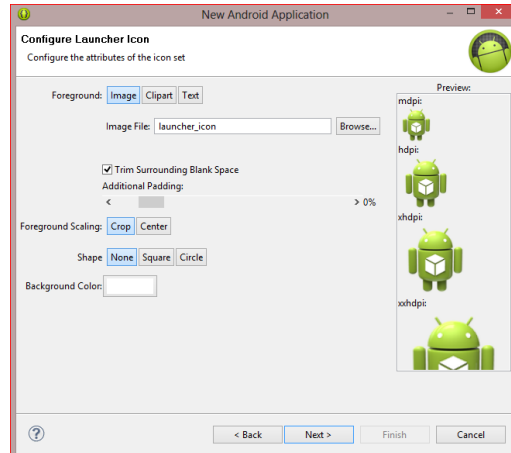


Fig 2.4 Creating Logo

- 3) This screen allows you to specify the type of activity. After selecting the type of activity click on next.

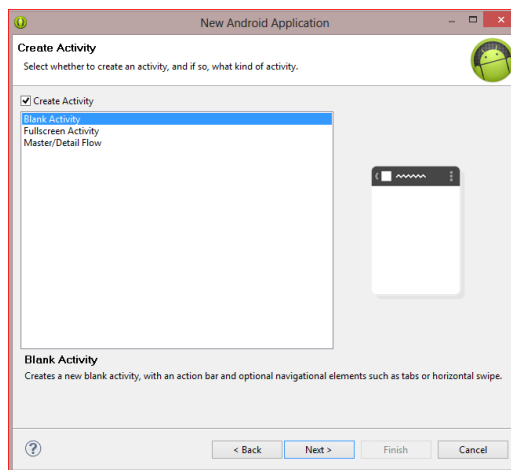


Fig 2.5 Creating Activity

- 4) This screen allows you to specify the name of Activity and Layout. After specifying the names click on finish.

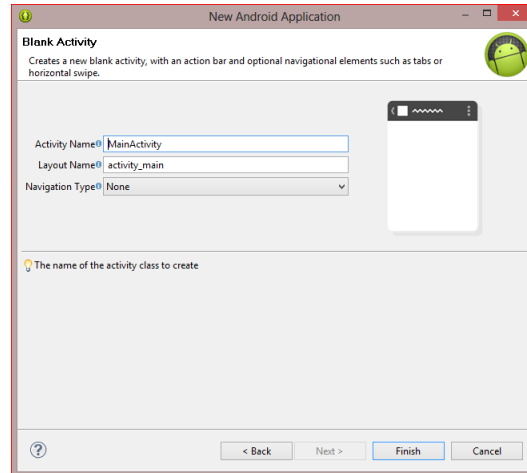


Fig 2.6 Specify Activity and Layout

- 5) After this you can see the contents of the project in the package explorer.

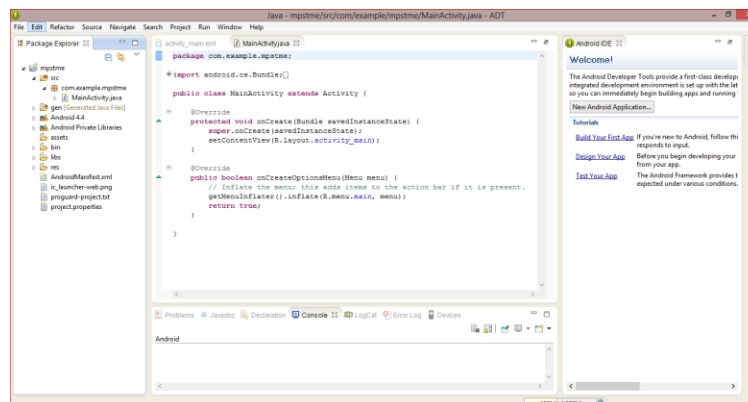


Fig 2.7 Contents of the project in the Package Explorer

- 7) Since we have already created the android virtual device, we can run the application by right clicking the project and selecting run as -> Android Application. The emulator may take a few minutes to start.

8) Emulator starts. We can now see the app that we have created.

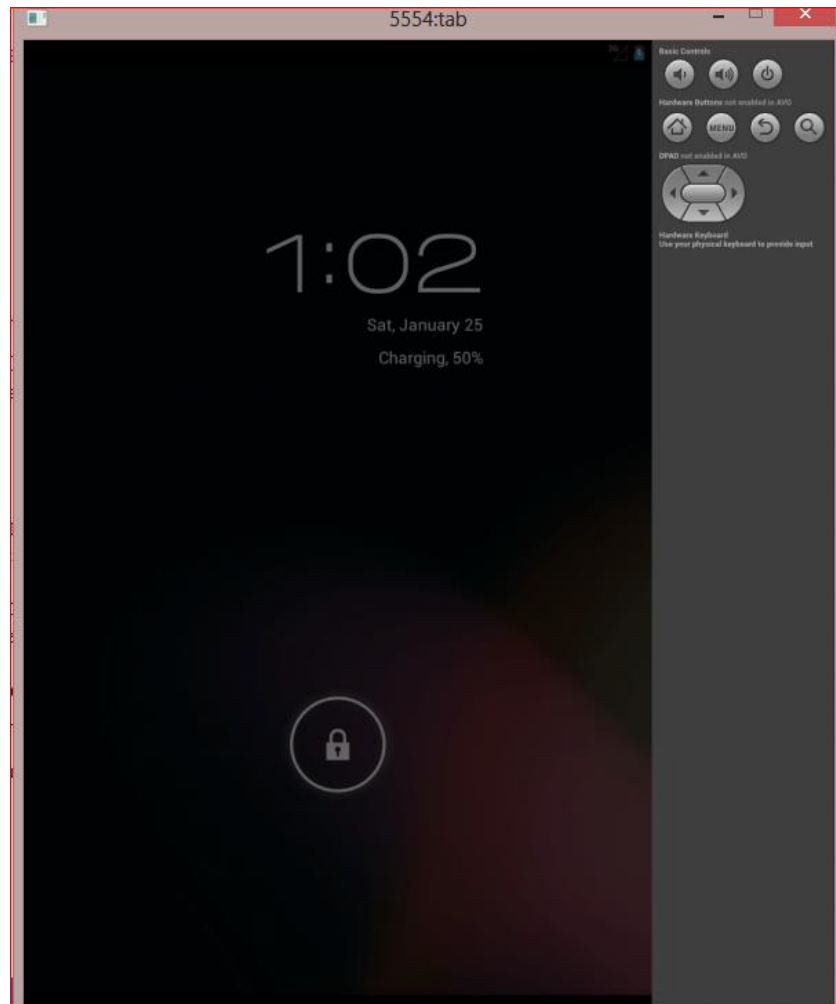


Fig 2.8 Emulator

### **3. Analysis and Design**

#### **3.1 Functional Requirements**

The basic idea is to develop a primary music player with an extremely user-friendly interface, which would serve to the basic needs of a music player user, with an additional feature of YouTube links.

The application accesses YouTube through Chrome browser. Hence, it becomes an absolute necessity for the device to have Chrome installed to be able to use this feature of JukeBox.

The various functions that the application performs include:

1. Select songs to play
2. Rename songs
3. Segregate songs based on various parameters
4. Play, Pause, Fast Forward a song
5. Scroll the song to a point using the scroll
6. Provide with YouTube Links

#### **3.2 Non-Functional Requirements**

1. Performance
2. Reliability
3. Availability
4. Security
5. Portability

#### **3.3 Design Constraints**

1. The system should be a mobile application.
2. The software development should be done on Android platform.
3. The music player should use the Phonon as the back-end.
4. Qt using JavaScript should be used for the GUI.

## **4 Project in Detail**

The software product to be produced is a Music Player for Android, called JukeBox.

The software will provide the user with the convenience of having an application on his/her device which would make it easy for the user to listen to songs on the go.

The user can select songs based on the song name, the artist of that song, the song album name, or the genre to which the song belongs.

The software to be produced has a smooth Interface and thus, makes the whole experience very convenient for the user.

The application gives the user the option of finding related YouTube videos of a particular song.

The database that we would be working on is the Music Library Database.

The software will provide the user with the convenience of having an application on his/her device which would make it easy for the user to listen to songs on the go.

The user can select songs based on the song name, the artist of that song, the song album name, or the genre to which the song belongs.

The software gives the users an option to find the YouTube links for the song he/she is listening to. The software to be produced has a smooth Interface and thus, makes the whole experience very convenient for the user.

### **4.1 General Requirements**

The general requirements of this system are:

- A clear idea of what the user wants from the system
- The functions that the user would want the software to have
- The interface specifications



## 4.2 General Constraints

The constraints on this system are:

- The software has to be delivered within 8 weeks.
- The software has to be compatible with Android.
- The software should consume the least amount of memory possible.
- User-friendly interface should be the main concern.
- Bugs should be minimized.
- Changing functions should be as fast as possible.
- The device needs to have Chrome browser installed to be able to access YouTube.

## 4.3 Interface



Fig 4.1 JukeBox on the Apps Page

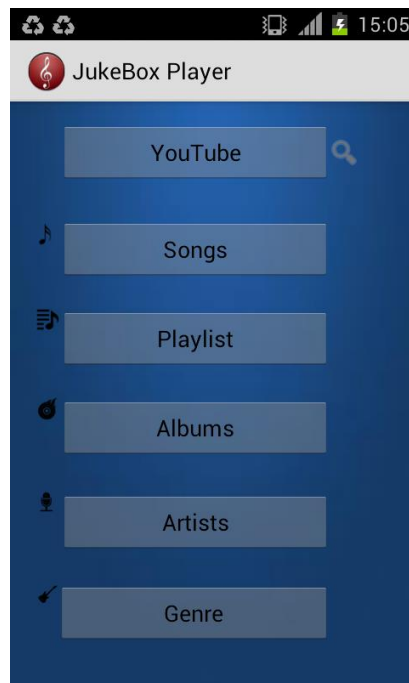


Fig 4.2 JukeBox Home

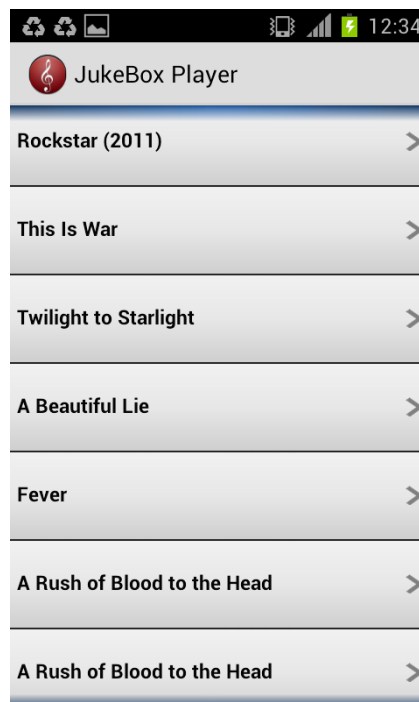


Fig 4.3 Albums List



Fig 4.4 Songs List

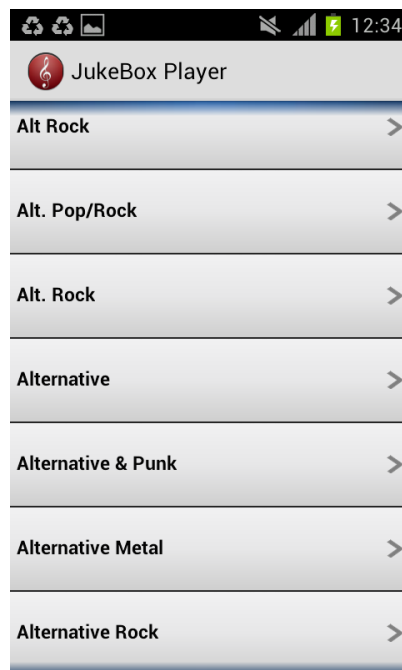


Fig 4.5 Genre List



Fig 4.6 Artist List



Fig 4.7 Player

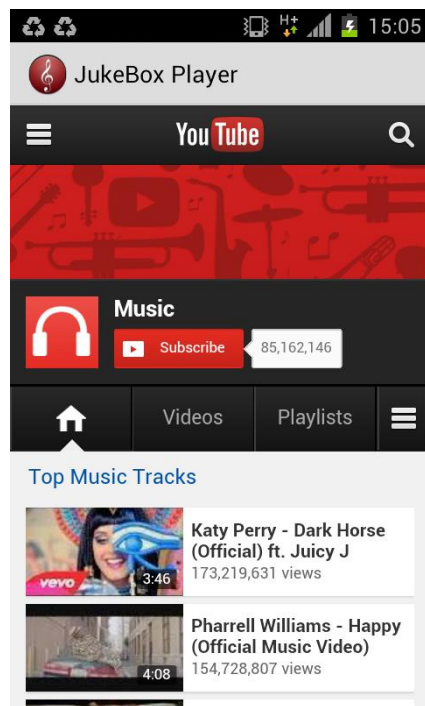


Fig 4.8 YouTube through Chrome

### **3 Conclusion and Future Scope**

The product is being developed keeping in mind the future prospects of this software. As of now, the application serves as a basic music player.

Right now due to indexing, creating a customized playlist is not possible. But the software would be developed keeping in mind that in the future, it could be possible to create multiple customized playlists.

Also, it could be connected to the web server for the option of being an online music player with a more dynamic library. Furthermore, the product would be incorporated with the dynamic retrieval of the lyrics of the song currently playing and would also give an option to perform on a karaoke track.

This music player could also be incorporated with a voice recorder.

Hence, we have been successful in designing and implementing an Android Application titled JukeBox, in our Project Workshop for Semester 6, Year 3.

## References

- [1] `Qt Documentation' <http://qt-project.org/doc/>
- [2] `Phonon Library' <http://phonon.kde.org/>
- [3] `Android Developers' <http://developer.android.com/develop/index.html>
- [4] `How to make an Android app' <https://developer.android.com/training/basics/firstapp/index.html?hl=it>
- [5] `Clickable Links' <http://stackoverflow.com/questions/9204303/android-is-it-possible-to-add-a-clickable-link-into-a-string-resource>

# Appendix

## A1. Functional Requirements

### **Functional Requirement 1: Playing a song**

Introduction: From a list of songs, the user selects the song to be played.

Input: The user clicks on the song to play it.

Processing: The system takes the 'click' as a trigger and selects that particular song.

Output: The chosen song is played.

### **Functional Requirement 2: Segregating songs based on various parameters**

Introduction: The songs are segregated based on the alphabetical order of their names, alphabetical order of the artists' names, the alphabetical order of the albums' names and the genres to which they belong.

Input: Addition of a new song is an input to this function.

Processing: The system takes the addition of a new audio file as the trigger and then segregates it accordingly.

Output: The playlists are well-ordered.

### **Functional Requirement 3: Pause, Play, Rewind and Fast Forward a song**

Introduction: The buttons to pause a playing song, play the paused song, to rewind a song by 2X value and to fast forward a song by a 2X value is provided to the user.

Input: The user clicks on the Pause, Play, Rewind or Fast Forward button.

Processing: The system takes the 'click' as a trigger and processes the activity requested.

Output: The song is paused, played, rewound, fast forwarded, according to the request sent by the user.

### **Functional Requirement 4: Scroll the song to a point using the seek bar**

Introduction: The option of a moving scroll is provided, which allows the user to scroll the song to a particular point.

Input: The user clicks on the scroll and drags it.

Processing: The system takes the 'drag' as a trigger and puts the time counter on that point of the song.

Output: The song is played from that chosen time value.

### **Functional Requirement 5: Provide with YouTube Links**

Introduction: The YouTube links related to the currently playing song are retrieved.

Input: The user clicks on the 'YouTube Links' button to retrieve the links.

Processing: The system takes the 'click' as a trigger and redirects to YouTube.

Output: The application is redirected to YouTube via Chrome.



## **A2. Non-Functional Requirements**

### **Non-Functional Requirement 1: Performance**

Response Time: The response time to play a song should be less than 1 sec.

Capacity: The capacity of songs that can be added to the music library should not be restricted to any number.

License: The system will be licensed under GPLv3. music library should not be restricted to any number.

User Interface The interface should respond within 2 seconds.

### **Non-Functional Requirement 2: Reliability**

The system should maintain the database of the records well, without losing even a single song from the memory.

### **Non-Functional Requirement 3: Availability**

The system should be available at all times.

### **Non-Functional Requirement 4: Security**

Login: The system will require root permissions for starting the port and configuring it to play music over it. The permissions required for this app pop up during Installation.

### **Non-Functional Requirement 5: Portability**

The system should be portable from one Android device to the other, while portability with respect to the Operating System is not allowed.

### A3. Use-Case Model

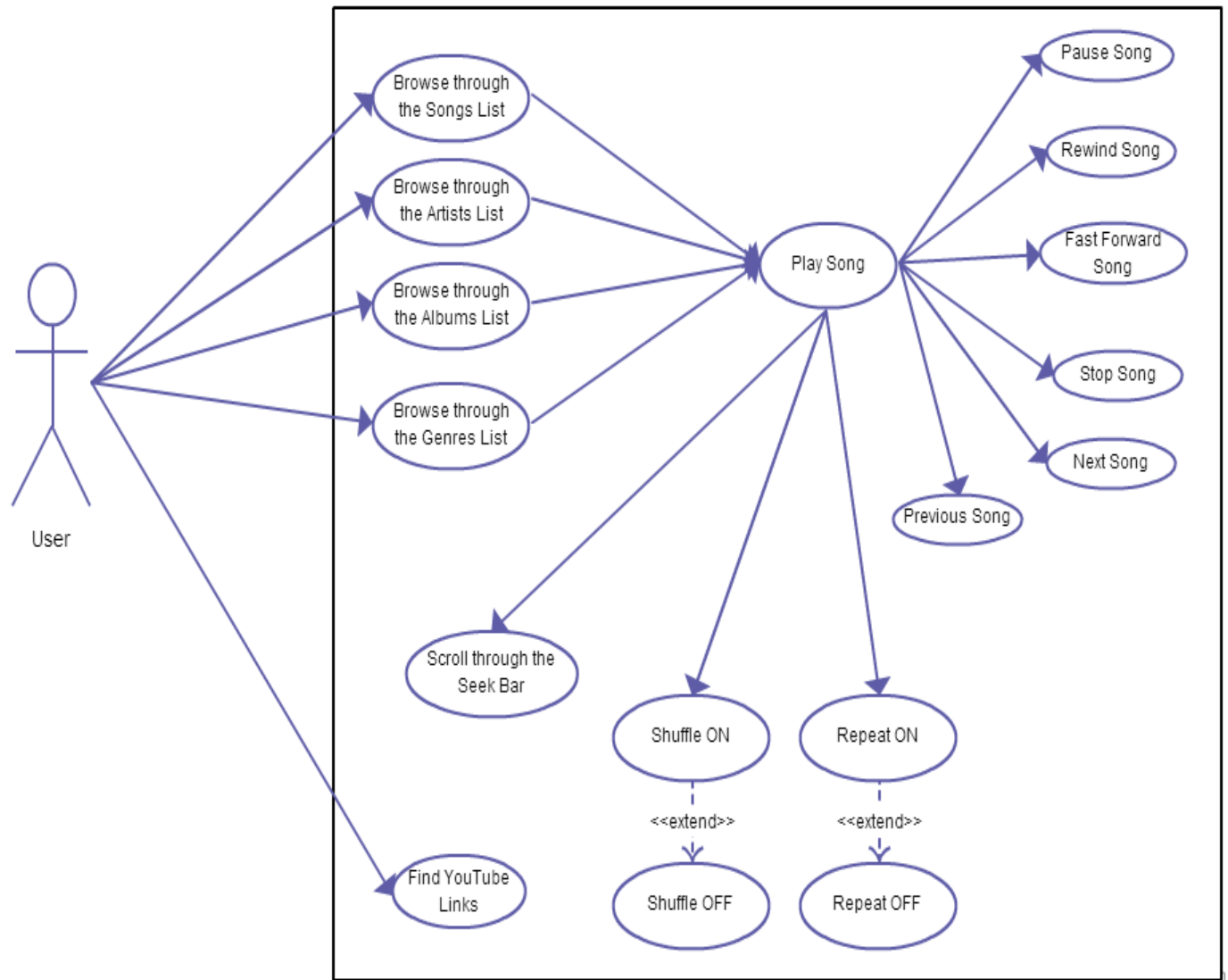


Fig A3.1 Use-Case Diagram