**MINI PROJECT**

**(2020-21)**

**Music Player App using Flutter and Dart Technology**



**PROJECT-REPORT**

**Institute of Engineering & Technology**

***Submitted by :-***

Manish Kumar (191599010)

Pawan Tyagi (191599012)

Pawnesh kumar pundhir (191599013)

***Supervised by :-***

Mr.Manoj Varshney

**Assistant Professor**

**Department of Computer Engineering & Applications**

**GLA University**

**Mathura- 281406,**

**INDIA**



**Department of computer Engineering and Applications**

**GLA University, Mathura**

**17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,**

**Mathura – 281406**

# DECLARATION

I hereby declare that the work which is being presented in the B.Tech. Project “**Music Player App using Flutter and Dart Technology**”, in partial fulfilment of the requirements for the award of the **Bachelor of Technology** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of **Mr. Manoj Varshney**.

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Sign\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Sign\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name : Manish Kumar Name : Pawan Tyagi

Uni. Roll No.: 191599010 Uni. Roll No.:191599012

Sign\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name : Pawnesh kumar pundhir

Uni. Roll No.: 191599013

Date:



**Department of computer Engineering and Applications**

**GLA University, Mathura**

**17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,**

**Mathura – 281406**

## CERTIFICATE

This is to be certify that project Report entitled “**Music Player App using Flutter and Dart Technology**” which is submitted by Manish Kumar, Pawan Tyagi and Pawnesh Kumar Pundhir in partial fulfillment of the requirement for the reward of degree of B.Tech in Stream of Computer Science & engineering of GLA University, is a record of the candidate’s own work carried out by then under my supervision. The matter embodied in this thesis is original and has not been submitted for the reward of any other degree.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Supervisor**

Mr. Manoj Varashney

Assistant Professor

Institute of Engineering & Technology

GLA University

# ACKNOWLEDGEMENT

I found this golden chance to acknowledge all those people who had blessed, encouraged and supported me technically and morally through all the phases of my project. I take this opportunity to express my profound sense of gratitude. I thank all mighty god for giving me this opportunity to express gratitude to all those who helped me in successful completion of this project.

I pay my immense gratitude to **Mr Anand singh Jalal , HOD Dept. of Computer Science** for providing help and giving me a chance for showing my skills through continued support and co-operation during the concerned project. I am deeply indebted to staff of Computer Dept., for their sincere co-operation and sparing time to answer questionnaires with their selfless efforts and co-operation because of which I am able to complete this project.

I am deeply indebted to my parents who have always been a perennial source of information, encouragement and inspiration for entire education required.

Thanks,

Name: Manish Kumar Name : Pawan Tyagi

University Roll No.: 191599010 University Roll No.: 191599012

Name : Pawnesh kumar pundhir

University Roll No.: 191599013

##### Abstract

In order to solve the problem of complex functions and large required memory of mobile phone music player on the current market, a new music player of simple, convenient, less required memory as well as user-friendly is developed. Based on the Android technology, using the Dart language and Flutter framework lead to design and coding of music player. The new design mainly realizes core functions including main play interface, playlists, menus, play settings, file browsing and song search.

This player has merits of high performance, simple operation, and run independently on the Android mobile devices. At the same time, the player can also browse and access files in mobile phones.

Table of Contents

[DECLARATION ………………………………………………………………………………………………………...2](#_Toc67071179)

[CERTIFICATE………………………………………………………………………………………………………..3](#_Toc67071180)

[ACKNOWLEDGEMENT …………………………………………………………………………………………..4](#_Toc67071181)

ABSTRACT…………………………….................................................................................................................5

1. INTRODUCTION ........................................................................................................................................1

1.1. INTRODUCTION ..................................................................................................................... 1

1.2. OBJECTIVE..................................................................................................................................2

1.3. TECHONOLOGIES USED ......................................................................................................2

1.3.1. Flutter ...........................................................................................………………………..2

1.3.2. Dart ....……........................................................................................................................3

1.4. WHAT IS A WEB APPLICATION (WEB APP) ..............................................................8

1.4.1. How a web application works ........................................................………………9 1.5. SCOPE………………………………………………………………………………………..11

2. SOFTWARE REQUIREMENT ANALYSIS .................................................................12

2.1. INTRODUCTION ................................................................................. ……..12

2.1.1. Purpose ........................................................................................................ 12

2.1.2. Document Convention: ................................................................................ 12

2.1.3. Intended Audience: .......................................................................................12

2.2. PERSPECTIVE: .................................................................................... ……..13

2.3. PRODUCT FUNCTIONS: .................................................................... ……..13

2.4. OPERATING ENVIRONMENT: ............................................................ …...13

2.5. DESIGN CONSTRAINS: ..................................................................... ……...13

2.6. ASSUMPTIONS AND DEPENDENCIES: .......................................... ……..14

2.7. PERORMANCE REQUIREMENTS: ..............................................................14

2.7.1. Hardware Requirements: ........................................................................... ..14

2.7.2. Software Requirements: .......................................................................... …14

3. SOFTWARE DESIGN ....................................................................................... 15

3.1. CONTENT DIAGRAM .................................................................... ……….15

3.2 DATAFLOW DIAGRAM……………………………………………............16.

3.2.1. DFD Level - 0 ..................................................................................16

3.2.2. DFD Level - 1 ................................................................................. 17

3.3. DATA STORAGE .......................................................................... …….17 3.3.1. LOCAL STORAGE. .................................................................. …….17 3.3.2. CLOUD STORAGE ...................................................................... 18

4. User Interface………………………………………………………………………..19

CONCLUSION AND FUTURE WORK…………………………………………………………….24

APPENDIX…………………………………………………………………………………………………24

REFERENCES**…………………………………………………………………………………………………..24**

##### 

##### INTRODUCTION

##### 1.1. INTRODUCTION

Android is open source code mobile phone operating system that comes out by Google in November 2007. Its appearance has broken the traditional closed mobile phone operating system. Anyone can modify the mobile phone operating system as well as function according to personal preference, which is also the most attractive merit of Android.

Music player in this article is application software based on Google Android. Android’s application on mobile terminals also completely broke the traditional understanding of the mobile terminals. And appreciate music is one of the best ways to relieve pressure in stressful modern society life. Therefore, many kinds of mobile phone players are also developed.

However, a lot of players devote to fancy appearance and function, while caused resources wasting to the user's mobile phone, such as large required memory and CPU, which brings a lot of inconvenience as multiple programs running at the same time. For the most ordinary users, many functions are useless. The purpose of this article is to develop a player which can play the mainstream music file format. To browse and query the storage space as well as operation of adding, deleting, and playing can be realized. Meanwhile, this software can play, pause and select songs with latest Btn and next Btn according to users’ requirement as well as set up songs’ order and etc,. Music player based on Android application is popular in the market at the present.

The completing development of Android operating system gives developers a nice platform, which can learn the popular computer technology combining with learned knowledge, and master the latest knowledge, enrich oneself, and enjoy entertainment.

##### 1.2. OBJECTIVE

The main objective of this project is to create a simple and interactive application so that it is easier to navigate through the application. Our main focus would be on User experience rather than UI.

**1.3. TECHONOLOGIES USED**

**1.3.1. Flutter**

Flutter is an open-source mobile SDK developer can use to build native-looking Android and iOS applications from the same code base. Flutter has been around since 2015 when Google introduced it and remained in the beta stage before its official launch in December 2018. Since then, the buzz around Flutter has been growing stronger.

**Widgets**

The central idea behind Flutter is the use of widgets. It’s by combining different widgets that developers can build the entire UI. Each of these widgets defines a structural element (like a button or menu), a stylistic element (a font or color scheme), a layout aspect (like padding), and many others.

Note that Flutter doesn’t use OEM widgets, but providers developers with its own ready-made widgets that look native to Android or iOS apps (following Material Design or Cupertino). Naturally, developers can create their own widgets as well.

Flutter also provides developers with reactive-style views. To avoid performance issues deriving from using a compiled programming language to serve as the JavaScript bridge, Flutter uses Dart. It compiles Dart ahead of time (AOT) into the native code for multiple platforms.

That way, Flutter can easily communicate with the platform without needing a JavaScript bridge that involves a context switch between the JavaScript realm and the native realm. As you can imagine, compiling to native code also boosts the app startup time.

Today, Flutter is the only mobile SDK that offers reactive views without the need for a JavaScript bridge. That’s why so many mobile developers have been trying it out in their projects.

Here are some more benefits Flutter brings to mobile software development.

Extra advantage: Dart programming language

One of the most interesting features of Flutter is the language it uses: Dart. Like other systems that use reactive views, Flutter refreshes the view tree for every new frame. To accomplish that, it creates many objects that may live for no more than one frame. Dart uses generational garbage collection that has proven to be very efficient for this type of systems.

Moreover, Dart has a “tree shaking” compiler that only includes the code you need in your app. Even if you need just a widget or two, you can use its large library of widgets freely.

Finally, Dart comes with a repository of software packages for extending the capabilities of apps. For example, it offers a few packages that help to access Firebase so that developers can build serverless apps. Another package allows accessing a Redux data store or makes it easier to access platform services and hardware like the camera.

**Benefits of Flutter:**

**It saves you time and money**

Flutter is a cross-platform development tool. That means software developers can use the same code base for building an iOS and Android app. Cross-platform development is the best method for saving time and resources throughout the development process.

**Excellent performance**

Flutter offers outstanding performance for two reasons. First, is uses Dart, which compiles into native code. Second, Flutter has its own widgets, so there’s no need to access OEM ones. As a result, there’s less communication between the app and the platform. These two features of Flutter ensure fast app startup times and fewer performance issues in general.

**Quick development thanks to hot reload**

Flutter is gaining a lot of traction among mobile developers because of hot reload. Hot reload allows to instantly view the changes applied to the code on emulators, simulators, and hardware. The changed code is reloaded in less than a second. All the while, the app is running and developers don’t need to waste time on restarting it.

That makes building UIs, adding new features, and fixing bugs easier. If an app encounters an error, it’s usually possible to fix it and then continue using the app as if it never happened. Even if you’re forced to do a full app reload, you can be sure that it’s completed in no time, accelerating the development process.

**Compatibility**

Another advantage of Flutter is the fact that it comes with its own widgets that result in fewer compatibility issues. Developers will see fewer problems on different OS versions and can spend less time on testing the app on older OS versions. Also, you can be confident that your app will work on future OS versions

**1.3.2. Dart**

Dart is an open-source, general-purpose, object-oriented programming language with C-style syntax developed by Google in 2011. The purpose of Dart programming is to create a frontend user interfaces for the web and mobile apps. It is under active development, compiled to native machine code for building mobile apps, inspired by other programming languages such as Java, JavaScript, C#, and is Strongly Typed. Since Dart is a compiled language so you cannot execute your code directly; instead, the compiler parses it and transfer it into machine code.

It supports most of the common concepts of programming languages like classes, interfaces, functions, unlike other programming languages. Dart language does not support arrays directly. It supports collection, which is used to replicate the data structure such as arrays, generics, and optional typing.

In Flutter all apps are composed with Dart. So, Dart is a OOP that is created and kept up by Google. It is generally utilized within Google, it has been demonstrated to have the ability to create enormous web apps, for example, AdWords. Dart was initially created as a substitution and successor of Java Script. Subsequently, it actualizes the majority of the significant attributes of Java Script's next standard (for example, watchwords "async" and "wait". Notwithstanding, so as to draw in designers that are curious about Java Content, Dart has a Java such as linguistic structure. Likened to different frameworks that use responsive perspectives, Flutter app invigorates the see tree on each new casing. This conduct prompts a downside that numerous items, which may alive for just a single casing, will be made. Dart, as an advanced programming code, is upgraded to deal with this situation in memory standard with the assistance of "Generational Waste Accumulation". Dart is a programming coding that we will utilize to build up our app in Flutter. Study it is not hard on the off chance that we have involvement with java content. We will rapidly fetch it. Dart is the customer enhanced language for quick applications on any stage do by Google.

Superior and efficiency in Flutter are accomplished by utilizing a few procedures:

1) Not at all like numerous other prominent versatile stages, Flutter makes not utilize java content at all. Dart is the coding. It assembles to parallel code, and that is the reason keeps running with the local execution of target java, quick, C.

2) Flutter does not utilize local user interface parts. That can sound clumsy at first. In any case, since segments are executed in Flutter itself, there is not correspondence see between the layer and the code. Because of this, diversions hit the best performance for their illustrations out of the cell phones, So catch, content, media components, foundation are altogether drawn by Flutter is designs motor. As a part, it ought to be referenced that the heap of the Flutter "Hi World" app is very little: android = 4Mb and iOS = 2.5Mb.

3) Flutter utilizes a definitive methodology, motivated by the respond web system, to manufacture its user interface dependent on gadgets (named "parts" in the realm of the site). To fetch increasingly out of gadgets, they are delivered just when fundamental, for the most part when their case has been modificated (simply such as the practical DOM accomplishes for us). Notwithstanding the majority of the above mentioned, the system has coordinated harm reload, so run of the mill for the web, yet at the same time missing on local stages. This enables the Flutter system to naturally revamp the gadget tree, enabling to rapidly see the impacts of our modulations.

**1.4. WHAT IS AN ANDROID APPLICATION**

Android apps can be written using Kotlin, Java, C++, and dart languages. The Android SDK tools compile your code along with any data and resource files into an APK, an Android package, which is an archive file with an .apk suffix. One APK file contains all the contents of an Android app and is the file that Android-powered devices use to install the app.

Each Android app lives in its own security sandbox, protected by the following Android security features:

* The Android operating system is a multi-user Linux system in which each app is a different user.
* By default, the system assigns each app a unique Linux user ID (the ID is used only by the system and is unknown to the app). The system sets permissions for all the files in an app so that only the user ID assigned to that app can access them.
* Each process has its own virtual machine (VM), so an app's code runs in isolation from other apps.
* By default, every app runs in its own Linux process. The Android system starts the process when any of the app's components need to be executed, and then shuts down the process when it's no longer needed or when the system must recover memory for other apps.

**1.4.1. How an android application works**

**Life Cycle Methods and Callbacks**

Android initiates the program within an activity with a call to *onCreate()* callback method. There is a sequence of callback methods that starts up an activity and then tear down in different methods shown in the above Activity life cycle diagram:

**1. onCreate()**: In this state, the activity is created.

**2. onStart():**This callback method is called when the activity becomes visible to the user.

**3. onResume()**: The activity is in the foreground and the user can interact with it.

**4. onPause()**: Activity is partially obscured by another activity. Another activity that’s in the foreground is semi-transparent.

**5. onStop()**: The activity is completely hidden and not visible to the user.

6. onRestart(): From the Stopped state, the activity either comes back to interact with the user or the activity is finished running and goes away. If the activity comes back, the system invokes onRestart()

7. onDestroy(): Activity is destroyed and removed from the memory.

So these are the various methods of the Activity Life Cycle. Now let’s see the situations where the life cycle methods and states will occur.

* **When you open the app it will go through below states:**

onCreate() –> onStart() –>  onResume()

* **When you press the back button and exit the app**

onPaused() — > onStop() –> onDestory()

* **When you press the home button**

onPaused() –> onStop()

* **After pressing the home button, again when you open the app from a recent task list**

onRestart() –> onStart() –> onResume()

* **After dismissing the dialog or back button from the dialog**

onResume()

* **If a phone is ringing and user is using the app**

onPause() –> onResume()

* **After the call ends**

onResume()

* **When your phone screen is off**

onPaused() –> onStop()

* **When your phone screen is turned back on**

onRestart() –> onStart() –> onResume()



**1.5. SCOPE**

This project can be further improved by

1. Implementing Machine Learning Algorithms to recommend songs as per the taste of user.
2. Implementing editing a song in real time like trimming.
3. Automating playlist making of user’s recent songs played.

**2. SOFTWARE REQUIREMENT ANALYSIS**

**2.1. INTRODUCTION**

The aim of this part is to gather and analyze and give an in-depth insight of the complete Music Player App Project by defining the problem statement in detail. Nevertheless, it also concentrates on the capabilities required by stakeholders and their needs while defining high-level product features. The detailed requirements of the Music Player App Project are provided in this document.

**2.1.1. Purpose**

The purpose of the document is to collect and analyze all assorted ideas that have come up to define the system, its requirements with respect to consumers. Also, we shall predict and sort out how we hope this product will be used in order to gain a better understanding of the project, outline concepts that may be developed later, and document ideas that are being considered, but may be discarded as the product develops.

In short, the purpose of this report document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements. It defines how our client, team and audience see the product and its functionality. Nonetheless, it helps any designer and developer to assist in software delivery lifecycle (SDLC) processes.

**2.1.2. Document Convention:**

In this text, it will use font small 2 and overstriking for primary title, font small 3 for secondary title and font 4 for the content. And it will use the italic when mentions the name of the application Music Player.

**2.1.3. Intended Audience:**

This SRS about Music Player App is for developers, mentors, users and testers. The article mainly introduces the overall description, external interface requirements system features and other non-functional requirements. I suppose mentor to read the whole article carefully and user pay attention to overall description especially. Users and testers read the system features carefully.

**2.2. PERSPECTIVE**

An android application preferably using the Flutter and Dart Language.

**2.3. PRODUCT FUNCTIONS:**

The android application has a simple interface with the songs available on a separate screen which can be played simply by clicking on them. It has two screens one for the songs and another for the main player.

In this application, User can simply interact with the application and play local songs available by default when user grants permission to the application.

**2.4. OPERATING ENVIRONMENT:**

This Application is based on the android platform. It is compatible and can run on any smartphone with android api level above 16.

**2.5. DESIGN CONSTRAINS:**

This user must accept the storage permission in order to load any songs inside the application. We must consider about the arrangement and beautification of the interface; Prioritization of processing operations and it depends upon the difficulty of coding and testing. After, the user grants permission of storage to the application then the application fetches all the available local songs in the storage.

**2.6. ASSUMPTIONS AND DEPENDENCIES:**

The people who manage the application should know about the knowledge of the Flutter SDK and Dart language as well as the architecture with lifecycle of android application.

List- Flutter SDK and Dart language, flutter android query, Material design.

**2.7. PERFORMANCE REQUIREMENTS**

**2.7.1. Hardware Requirements:**

Table 1.1 Hardware specifications

|  |  |
| --- | --- |
| **HARDWARE** | **MINIMUM SYSTEM REQUIREMENTS** |
| Processor | SD 610 or above |
| Memory | 50Mb |
| RAM | 4GB |

**2.7.2. Software Requirements:**

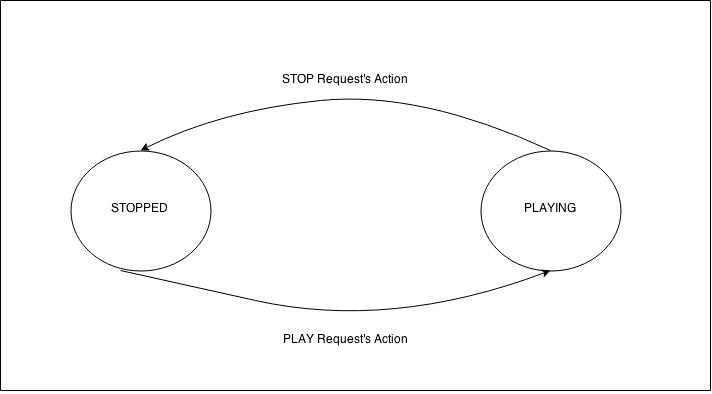
Table 1.2 Software specifications

|  |  |
| --- | --- |
| **SOFTWARE** | **MINIMUM SYSTEM REQUIREMENTS** |
| Client Operating System | Android 5 or above |
| Programming Languages | Dart |
| Dependencies | Flutter SDK, Flutter audio query |
| Text Editor | Visual Studio Code |

**3.SOFTWARE DESIGN**

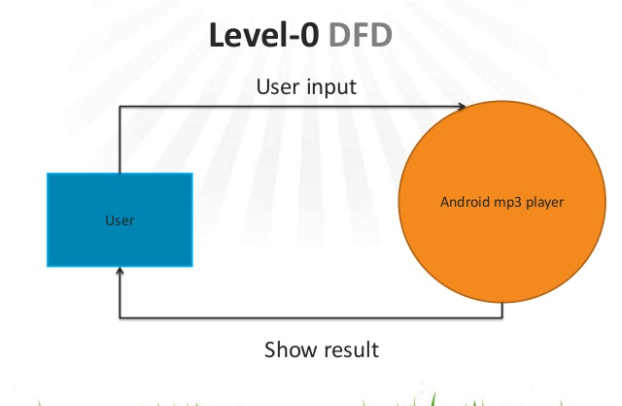
Function Oriented Design for procedural approach and different diagram to show the designing of the application.

**3.1. CONTENT DIAGRAM**

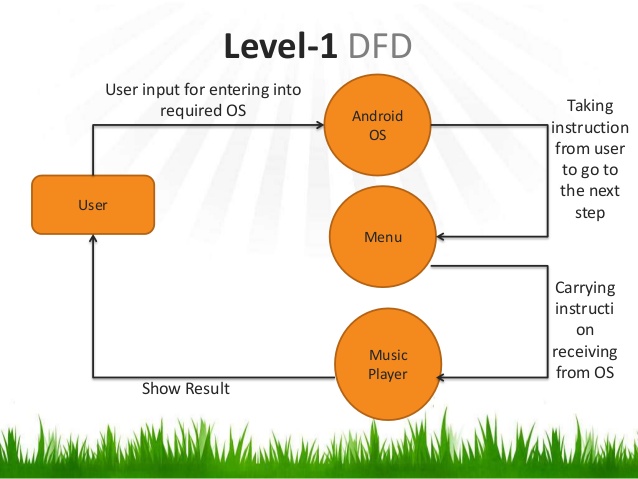
****

**3.2 DATAFLOW DIAGRAM**

**3.2.1. DFD Level - 0**

****

**3.2.2. DFD Level - 1**

****

**3.3. DATA STORAGE**

When players run normally, because of the switch among interface, in order to avoid the data lost, we need to store some data for local or cloud storage. As a kind of mobile phone operating system, Android provides the following ways for data storage: Preference (configuration), File (documents), SQLite data and network. Application component between each other is independent In Android, and data cannot be shared. In order to realize data sharing, Android provides Content Provider components to realize the sharing of data among applications.\

Now we have used local storage.In the future, we will be using cloud storage.

**3.3.1. LOCAL STORAGE**

**Using Shared Preferences**

Shared Preferences is the way to go if you’re saving primitive data as key-value pairs. It requires a key, which is a String, and the corresponding value for the said key. The value can be any of the following: a boolean, float, int, long, or another string.

Your Android device stores each app’s Shared Preferences inside of an XML file in a private directory. Apps can also have more than one Shared Preferences file, and they’re ideally used to store app preferences.

**Using internal storage**

There are plenty of times where you may need to persist data, but you find Shared Preferences too limiting. For example, you may need to persist objects or images in Java. You might also need to persist your data logically with the file system hierarchy. This is where internal storage comes in. It is specifically for when you need to store data on the file system, but you don’t want other apps or users to have access.

This data storage is so private, in fact, that it’s deleted from the device as soon as you uninstall your app.

Using internal storage is similar to saving with any other file system. You can get references to File objects, and you can store data of virtually any type using a FileOutputStream. What sets it apart is the fact that its contents are only accessible by your app.

**External Storage**

If your app tries to access a file that it did not create, you will have to permit it to do so every single time. Data you store outside of select folders will also disappear if you delete your app.

Using External Storage is identical to using Internal Storage and other filesystems. The difference here is that external storage devices can be removable drives, and all apps can read the contents. Since external storage can be removed and/or shared (mounted) with a computer or in one of several other states, before writing to external storage, your app should check for media availability.

**SQLite database**

Finally, Android provides support for apps to use SQLite databases for data storage. The databases you create remain specific to your app and can only be accessed inside your app. Of course, you should have at least some knowledge of SQL before you attempt to store data with an SQLite database.

**3.3.2. CLOUD STORAGE**

Cloud Music Player is a revolutionary music player that puts you in control of your music, no matter where it's stored. Use it as an offline music player or link your Dropbox, OneDrive and Google Drive to create a giant cloud jukebox for all your music. Stream from your cloud accounts and cache tracks for offline playback. Create playlists, add tracks, artists and albums to favorites and navigate through your music with our advanced search.

Cloud-based or online storage solutions offer virtual data storage and convenient access to your materials from anywhere, not just a local computer or external hard disk. Ex. Heroku

**4.User Interface**

****

Fig. 1Homepage

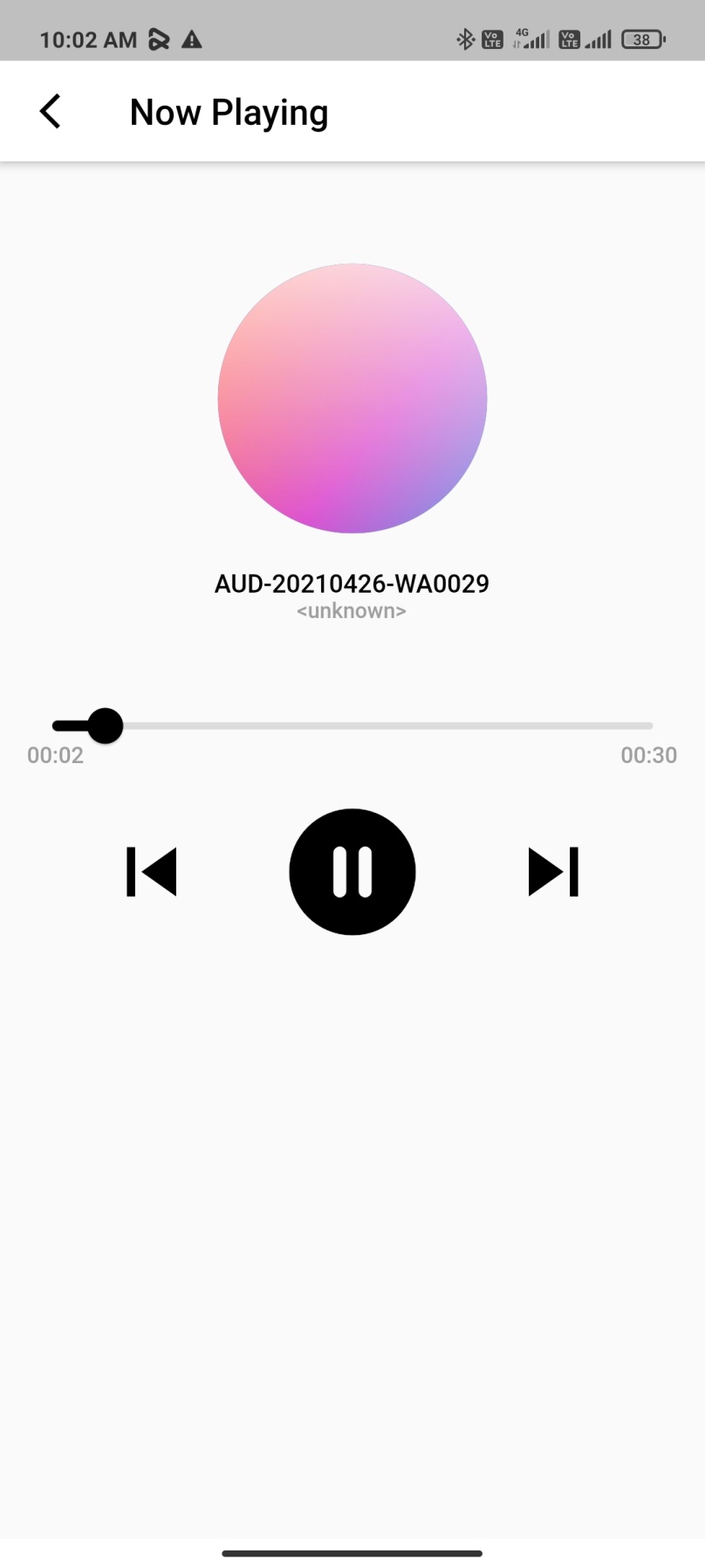
****

Fig 2 Screen while Playing Song

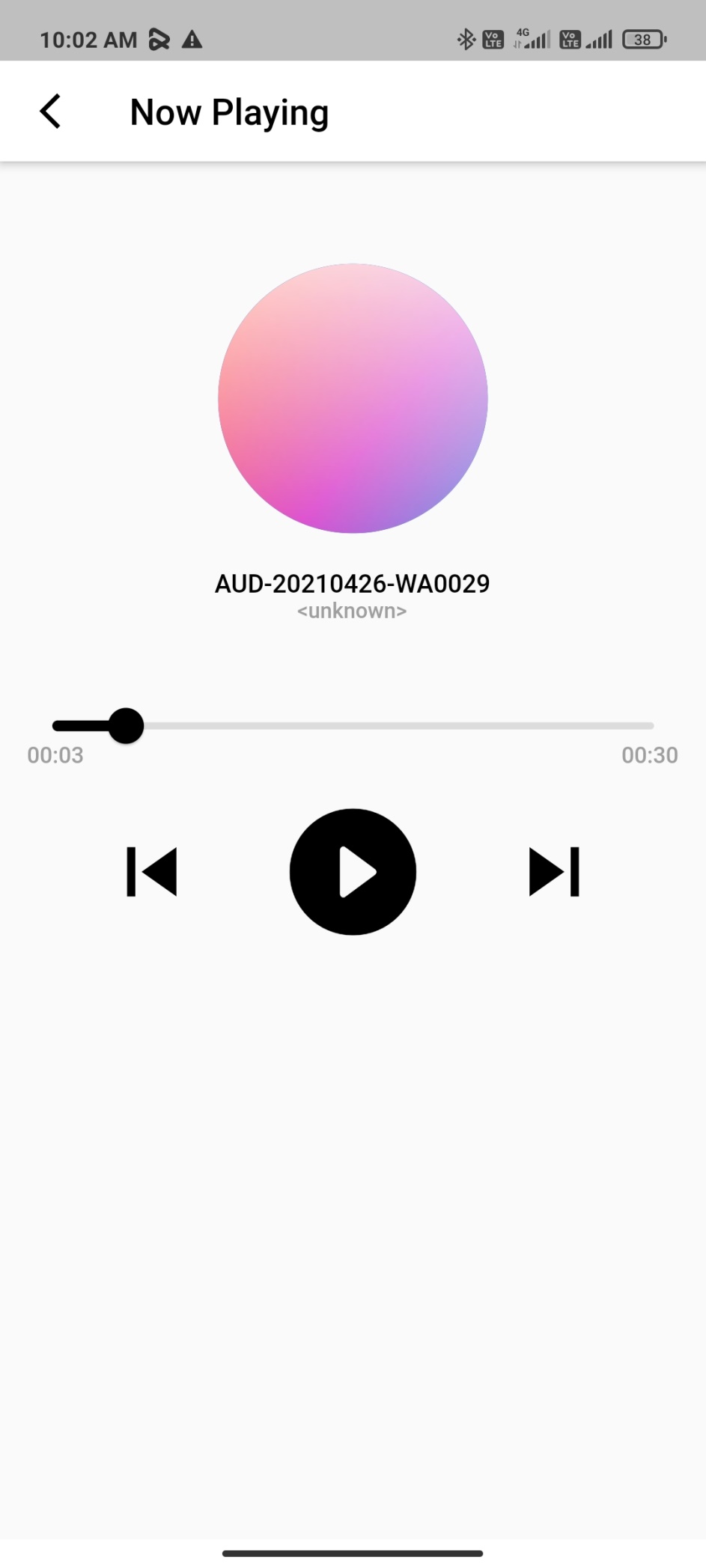
****

Fig 3. Stopped Song

**CONCLUSION AND FUTURE WORK**

**CONCLUSION**

Through the development of music player on Android platform, we get a clear understanding of overall process of the system. The core part of the music player is mainly composed of main interface, playlists, menus, file browsing and song search. Grasping the development of the six parts, the music player has had the preliminary scale. Based on the function of the six categories, add some other small features. Music player system realized the basic function of player: play, pause, and stop, up/down a, volume adjustment, play mode, song search, file browser, playlists query, and other functions. This development implicated the popular mobile terminal development technology. This is the combination management of Java language in the open-source mobile platform based on Linux system+ + SQLite database support+ Share Preference configuration file. The system realized the music player programming. This design of music player based on Android system requires elaborate design of the music player framework, by adopting Visual code studio + flutter language as technical support of this system, with the Android plug-in tools, and combination of Android SDK2.1 version lead to the comprehensive and smoothly design and development of the mobile terminal.

**FUTURE WORK**

This project can be further improved by

1. Implementing Machine Learning Algorithms to recommend songs as per the taste of user.
2. Implementing editing a song in real time like trimming.
3. Automating playlist making of user’s recent songs played.

We’ll work on another feature in our app like up and down slider to change songs

**APPENDIX**

* 1. https://github.com/manishkumar1/ipawantyagi