VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"Jnana Sangama", Belagavi, Karnataka, India.

A Mini Project Report On

Media Player Application

Submitted in partial fulfillment of the requirement for VI Semester

Mobile Application Development Laboratory[18CSMP68]

in
Computer Science and Engineering

Submitted By

Arpitha N G [1GA18CS031]
Amrutha K [1GA18CS194]

Under the guidance of

Dr. KAVITHA K S, Professor
Prof. SOWMYA M, Assistant Professor

Department of Computer Science and Engineering
(Accredited by NBA 2019-2022)
GLOBAL ACADEMY OF TECHNOLOGY
Rajarajeshwari Nagar, Bengaluru - 560 098
2020 – 2021

ABSTRACT

The project is an Android Studio-based media player application. In addition, the media player will add some features based on the user's usage scenario. The media Player, as usual, has a limited amount of tracks kept in it, and we may use it to play, pause, and advance an audio or song. To shift the audio forward or backward, use the seek bar's indicator. The project is comprised of a unique dashboard, and we can also refresh the media player; when we refresh, a notice stating "refreshing" appears. The song's name is likewise displayed in a similar manner. When we click on the "About" option on the dashboard, a pop-up window appears with information about the section.

IMPLEMENTATION

Program code:

- **activity_main.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Activity.MainActivity">
<RelativeLayout
   android:id="@+id/content_frame"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
<android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
    android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
    android:background="?attr/colorPrimary"
    android:theme="@style/ThemeOverlay.AppCompat.ActionBar" />
<LinearLayout
   android:id="@+id/ll_include_controls"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/toolbar">
```

```
<include layout="@layout/player_layout" />
</LinearLayout>
<LinearLayout
   android:id="@+id/ll_tab_layout"
    android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
    android:layout_below="@id/ll_include_controls"
    android:background="@color/colorPrimary">
<android.support.design.widget.TabLayout
    android:id="@+id/tabs"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:elevation="10dp"
    app:tabBackground="@color/colorPrimary"
    app:tabGravity="fill"
    app:tabIndicatorColor="@color/colorPrimary"
    app:tabMode="fixed"
    app:tabSelectedTextColor="@color/text_color"
    app:tabTextColor="@color/off_color" />
</LinearLayout>
<android.support.v4.view.ViewPager
    android:id="@+id/songs_viewpager"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/ll_tab_layout"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"/>
<android.support.design.widget.FloatingActionButton
    android:id="@+id/btn_refresh"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
```

```
    android:layout_alignParentBottom="true"
    android:layout_alignParentEnd="true"
    android:layout_marginBottom="15dp"
    android:layout_marginEnd="15dp"
    android:focusable="true"
    android:src="@drawable/refresh_icon"
    app:backgroundTint="@color/colorPrimary"
    app:layout_anchor="@id/songs_viewpager" />
</RelativeLayout>
<android.support.design.widget.NavigationView
    android:id="@+id/nav_view"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:layout_gravity="start"
    android:background="@color/drawer_color"
    android:fitsSystemWindows="true"
    app:headerLayout="@layout/drawer_header_layout"
    app:itemIconTint="@color/text_color"
    app:itemTextColor="@color/text_color"
    app:menu="@menu/navigation_menu" />
</android.support.v4.widget.DrawerLayout>
```

- **drawer_header_layout.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="200dp"
    android:layout_centerHorizontal="true"
    android:background="@drawable/header_background"
    android:fitsSystemWindows="true"
```

```
    android:gravity="center">
<ImageView
    android:layout_marginStart="10dp"
    android:id="@+id/img_icon"
    android:layout_width="120dp"
    android:layout_height="120dp"
    android:focusable="true"
    android:src="@drawable/headset_icon" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/img_icon"
    android:text="@string/title"
    android:textColor="@color/text_color"
    android:textSize="30sp"
    android:textStyle="bold" />
</RelativeLayout>
```

- **fragment_tab.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingLeft="10dp"
    android:paddingRight="10dp">
<ListView
    android:id="@+id/list_playlist"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```
    android:divider="@color/gray_color"
    android:dividerHeight="1dp"
    android:longClickable="true" />
</LinearLayout>
```

- **player_layout.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="100dp"
    android:background="@color/colorPrimary">
<LinearLayout
    android:id="@+id/ll_player_controls"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp"
    android:gravity="center_horizontal"
    android:orientation="horizontal">
<ImageButton
    android:id="@+id/img_btn_replay"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="10dp"
    android:backgroundTint="@color/colorPrimary"
    android:src="@drawable/undo_icon" />
<ImageButton
    android:id="@+id/img_btn_previous"
    android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_marginStart="30dp"
        android:backgroundTint="@color/colorPrimary"
        android:src="@drawable/previous_icon" />
    <ImageButton
        android:id="@+id/img_btn_play"
        android:layout_width="wrap_content"
       android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:backgroundTint="@color/colorPrimary"
        android:src="@drawable/play_icon" />
    <ImageButton
        android:id="@+id/img_btn_next"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:backgroundTint="@color/colorPrimary"
        android:src="@drawable/next_icon" />
    <ImageButton
        android:id="@+id/img_btn_setting"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="30dp"
        android:backgroundTint="@color/colorPrimary"
        android:src="@drawable/ic_settings" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/ll_player_controls"
```

```
    android:layout_marginTop="10dp"
    android:gravity="center_horizontal"
    android:orientation="horizontal">
  <TextView
    android:id="@+id/tv_current_time"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/start_time"
    android:textColor="@color/text_color"
    android:textSize="15sp"
    android:textStyle="bold" />
  <SeekBar
    android:id="@+id/seekbar_controller"
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:backgroundTint="@color/text_color"
    android:fitsSystemWindows="true"
    android:progressTint="@color/text_color"
    android:thumbTint="@color/text_color" />
  <TextView
    android:id="@+id/tv_total_time"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/final_time"
    android:textColor="@color/text_color"
    android:textSize="15sp"
    android:textStyle="bold" />
  </LinearLayout>
  </RelativeLayout>
```

- **playlist_items.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="@color/text_color"
   android:padding="10dp">
<ImageView
   android:id="@+id/iv_music_list"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:padding="5dp"
   android:src="@drawable/ic_music_player" />
<TextView
   android:id="@+id/tv_music_name"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_marginStart="10dp"
   android:layout_toEndOf="@id/iv_music_list"
   android:singleLine="true"
   android:text="Title"
   android:textColor="@color/color_black"
   android:textSize="16sp"
   android:textStyle="bold" />
<TextView
   android:id="@+id/tv_music_subtitle"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_below="@id/tv_music_name"
```

```
    android:layout_marginStart="10dp"
    android:layout_toEndOf="@id/iv_music_list"
    android:singleLine="true"
    android:text="SubTitle"
    android:textColor="@color/gray_color"
    android:textSize="12sp"
    android:textStyle="normal" />
</RelativeLayout>
```

- **colors.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#45b5ec</color>
    <color name="colorPrimaryDark">#1c8fc7</color>
    <color name="colorAccent">#000000</color>
    <color name="text_color">#ffffff</color>
    <color name="color_black">#000000</color>
    <color name="drawer_color">#335763</color>
    <color name="light_color">#fafafa</color>
    <color name="off_color">#dfdada</color>
    <color name="gray_color">#9c9b9b</color>
</resources>
```

- **string.xml**

```
<resources>
    <string name="app_name">Nav Music</string>
    <string name="title">Nav Music</string>
    <string name="theme_color">Theme Color</string>
    <string name="sleep_timer">Sleep Timer</string>
```

```
    <string name="rate_app">Rate This App</string>
    <string name="about">About</string>
    <string name="search">Search</string>
    <string name="favorites">Favorites</string>
    <string name="about_text">Nav Music is an application to play mp3 music on your android device. With simple interface make you enjoy the music easily. Custom your color as you feel.</string>
    <string name="ok">OK</string>
    <string name="start_time">0.00</string>
    <string name="final_time">0.00</string>
    <string name="search_hint">Song Name</string>
    <string name="no">No</string>
    <string name="yes">Yes</string>
    <string name="delete">Delete</string>
    <string name="delete_text"> Are you sure you want to delete</string>
    <string name="play_next">Play Next</string>
</resources>
```

- **styles.xml**

```
<resources>
    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
     </style>
     <style name="ActionBar"
         parent="@style/Widget.AppCompat.Light.ActionBar">
     <item name="android:background">@color/colorPrimary</item>
```

```
    <item name="background">@color/colorAccent</item>
    </style>
</resources>
```

- **AndroidMainfest.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<manifest          xmlns:android=http://schemas.android.com/apk/res/android
package="com.example.soc_macmini_15.musicplayer">
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission    android:name="android.permission.READ_EXTERNAL_STORAGE"
/>
<application
   android:allowBackup="true"
   android:icon="@drawable/ic_music"
   android:label="@string/app_name"
   android:roundIcon="@mipmap/ic_launcher_round"
   android:supportsRtl="true"
   android:theme="@style/AppTheme">
<activity
   android:name=".Activity.MainActivity"
   android:launchMode="singleTop"
   android:screenOrientation="portrait">
   <intent-filter>
   <action android:name="android.intent.action.MAIN" />
   <category android:name="android.intent.category.LAUNCHER" />
   </intent-filter>
<meta-data
   android:name="android.app.searchable"
   android:resource="@xml/searchable" />
</activity>
```

```
</application>
</manifest>
```

- **MainActivity.java**

```
package com.example.soc_macmini_15.musicplayer.Activity;
import android.Manifest;
import android.app.SearchManager;
import android.content.Context;
import android.content.DialogInterface;
import android.content.pm.PackageManager;
import android.media.MediaPlayer;
import android.os.Handler;
import android.support.annotation.NonNull;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.NavigationView;
import android.support.design.widget.Snackbar;
import android.support.design.widget.TabLayout;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v4.view.ViewPager;
import android.support.v4.widget.DrawerLayout;
import android.support.v7.app.ActionBar;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Gravity;
import android.view.Menu;
import android.view.MenuItem;
import android.support.v7.widget.Toolbar;
import android.view.View;
```

```
import android.widget.ImageButton;
import android.widget.SearchView;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;
import com.example.soc_macmini_15.musicplayer.Adapter.ViewPagerAdapter;
import com.example.soc_macmini_15.musicplayer.DB.FavoritesOperations;
import com.example.soc_macmini_15.musicplayer.Fragments.AllSongFragment;
import
com.example.soc_macmini_15.musicplayer.Fragments.CurrentSongFragment;
import com.example.soc_macmini_15.musicplayer.Fragments.FavSongFragment;
import com.example.soc_macmini_15.musicplayer.Model.SongsList;
import com.example.soc_macmini_15.musicplayer.R;
import java.util.ArrayList;
public class MainActivity extends AppCompatActivity implements
View.OnClickListener, AllSongFragment.createDataParse,
FavSongFragment.createDataParsed, CurrentSongFragment.createDataParsed {
    private Menu menu;
    private ImageButton imgBtnPlayPause, imgbtnReplay, imgBtnPrev,
imgBtnNext, imgBtnSetting;
     private TabLayout tabLayout;
     private ViewPager viewPager;
     private SeekBar seekbarController;
     private DrawerLayout mDrawerLayout;
    private TextView tvCurrentTime, tvTotalTime;
    private ArrayList<SongsList> songList;
     private int currentPosition;
     private String searchText = "";
     private SongsList currSong;
    private boolean checkFlag = false, repeatFlag = false, playContinueFlag
```

```
= false, favFlag = true, playlistFlag = false;
      private final int MY_PERMISSION_REQUEST = 100;
      private int allSongLength;
      MediaPlayer mediaPlayer;
      Handler handler;
      Runnable runnable;
@Override
      protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      init();
      grantedPermission();
   }
   /* Initialising the views*/
   private void init() {
      imgBtnPrev = findViewById(R.id.img_btn_previous);
      imgBtnNext = findViewById(R.id.img_btn_next);
      imgbtnReplay = findViewById(R.id.img_btn_replay);
      imgBtnSetting = findViewById(R.id.img_btn_setting);
      tvCurrentTime = findViewById(R.id.tv_current_time);
      tvTotalTime = findViewById(R.id.tv_total_time);
      FloatingActionButton refreshSongs = findViewById(R.id.btn_refresh);
      seekbarController = findViewById(R.id.seekbar_controller);
      viewPager = findViewById(R.id.songs_viewpager);
      NavigationView navigationView = findViewById(R.id.nav_view);
      mDrawerLayout = findViewById(R.id.drawer_layout);
      imgBtnPlayPause = findViewById(R.id.img_btn_play);
      Toolbar toolbar = findViewById(R.id.toolbar);
      handler = new Handler();
      mediaPlayer = new MediaPlayer();
```

```
      toolbar.setTitleTextColor(getResources().getColor(R.color.text_color));
      setSupportActionBar(toolbar);
      ActionBar actionBar = getSupportActionBar();
      assert actionBar != null;
      actionBar.setDisplayHomeAsUpEnabled(true);
      actionBar.setHomeAsUpIndicator(R.drawable.menu_icon);
      imgBtnNext.setOnClickListener(this);
      imgBtnPrev.setOnClickListener(this);
      imgbtnReplay.setOnClickListener(this);
      refreshSongs.setOnClickListener(this);
      imgBtnPlayPause.setOnClickListener(this);
      imgBtnSetting.setOnClickListener(this);
      navigationView.setNavigationItemSelectedListener(new
      NavigationView.OnNavigationItemSelectedListener() {
@Override
      public boolean onNavigationItemSelected(@NonNull MenuItem item) {
      item.setChecked(true);
      mDrawerLayout.closeDrawers();
      switch (item.getItemId()) {
           case R.id.nav_about:
           about();
           break;
                      }
      return true;
                }
           });
       }
   /* Function to ask user to grant the permission.*/
   private void grantedPermission() {
   if(ContextCompat.checkSelfPermission(MainActivity.this,
```

```
   Manifest.permission.READ_EXTERNAL_STORAGE)!=
   PackageManager.PERMISSION_GRANTED) {
      ActivityCompat.requestPermissions(MainActivity.this,new
   String[]{Manifest.permission.READ_EXTERNAL_STORAGE},MY_PERMISSION_REQUEST;
   if(ActivityCompat.shouldShowRequestPermissionRationale(MainActivity.this,
      Manifest.permission.READ_EXTERNAL_STORAGE)) {
      ActivityCompat.requestPermissions(MainActivity.this,new
   String[]{Manifest.permission.READ_EXTERNAL_STORAGE},MY_PERMISSION_REQUEST;
   } else{
   if (ContextCompat.checkSelfPermission(MainActivity.this,
   Manifest.permission.READ_EXTERNAL_STORAGE)!=PackageManager.PERMISSION_GRAT
   ED){
      Snackbar snackbar = Snackbar.make(mDrawerLayout, "Provide the Storage
      Permission", Snackbar.LENGTH_LONG);
      snackbar.show();
       }
    }
   } else{
         setPagerLayout();
        }
      }
/*Checking if the permission is granted or not, @param requestCode ,@param
permissions, @param grantResults*/
@Override
   public void onRequestPermissionsResult(int requestCode, @NonNull String[]
   permissions, @NonNull int[] grantResults) {
   switch (requestCode) {
      case MY_PERMISSION_REQUEST:
      if(grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
```

```
      if(ContextCompat.checkSelfPermission(MainActivity.this,
Manifest.permission.READ_EXTERNAL_STORAGE)==PackageManager.PERMISSION_GRANTE)
{Toast.makeText(this, "Permission Granted!", Toast.LENGTH_SHORT).show();
   setPagerLayout();}
      else{Snackbar  snackbar  =  Snackbar.make(mDrawerLayout,  "Provide  the
Storage Permission", Snackbar.LENGTH_LONG);
            snackbar.show();
            finish();
                    }
                 }
           }
       }
/*Setting up the tab layout with the viewpager in it.*/
private void setPagerLayout(){
ViewPagerAdapter adapter = new ViewPagerAdapter(getSupportFragmentManager(),
getContentResolver());
viewPager.setAdapter(adapter);
viewPager.addOnPageChangeListener(new ViewPager.OnPageChangeListener(){
@Override
   public  void  onPageScrolled(int  position,  float  positionOffset,  int
positionOffsetPixels) {}
@Override
    public void onPageSelected(int position) {}
@Override
    public void onPageScrollStateChanged(int state) {}
});
tabLayout = findViewById(R.id.tabs);
tabLayout.setTabGravity(TabLayout.GRAVITY_FILL);
tabLayout.setupWithViewPager(viewPager);
tabLayout.addOnTabSelectedListener(new TabLayout.OnTabSelectedListener(){
```

```
@Override
   public void onTabSelected(TabLayout.Tab tab) {
                  viewPager.setCurrentItem(tab.getPosition());}
@Override
    public void onTabUnselected(TabLayout.Tab tab) {}
@Override
    public void onTabReselected(TabLayout.Tab tab) {}
    });
}
/* Function to show the dialog for about us.*/
private void about() {
   AlertDialog.Builder builder = new AlertDialog.Builder(this);
   builder.setTitle(getString(R.string.about))
   .setMessage(getString(R.string.about_text))
   .setPositiveButton(R.string.ok, new DialogInterface.OnClickListener() {
@Override
   public void onClick(DialogInterface dialog, int which) {}
});
AlertDialog alertDialog = builder.create();
alertDialog.show();
}
@Override
       public boolean onCreateOptionsMenu(Menu menu) {
          this.menu = menu;
          getMenuInflater().inflate(R.menu.action_bar_menu, menu);
          SearchManager          manager          =          (SearchManager)
   getSystemService(Context.SEARCH_SERVICE);
          SearchView           searchView           =           (SearchView)
   menu.findItem(R.id.menu_search).getActionView();
searchView.setSearchableInfo(manager.getSearchableInfo(getComponentName());
```

```
searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
@Override
   public boolean onQueryTextSubmit(String query) {
   return false;
}
@Override
    public boolean onQueryTextChange(String newText) {
    searchText = newText;
    queryText();
    setPagerLayout();
    return true;
       }
   });
return super.onCreateOptionsMenu(menu);
}
@Override
    public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
    case android.R.id.home : mDrawerLayout.openDrawer(Gravity.START);
      return true;
      case R.id.menu_search:
       Toast.makeText(this, "Search", Toast.LENGTH_SHORT).show();
       return true;
      case R.id.menu_favorites:
       if(checkFlag)
       if(mediaPlayer!=null){
       if(favFlag){
       Toast.makeText(this, "Added to Favorites", Toast.LENGTH_SHORT).show();
                item.setIcon(R.drawable.ic_favorite_filled);
        SongsList     favList                        =                        new
```

```
      SongsList(songList.get(currentPosition).getTitle(),songList.get(current
      Position).getSubTitle(), songList.get(currentPosition).getPath());
        FavoritesOperations favoritesOperations = new
      FavoritesOperations(this);
      favoritesOperations.addSongFav(favList);
       setPagerLayout();
        favFlag = false;
   } else{
        item.setIcon(R.drawable.favorite_icon);
        favFlag = true;
     }
    }
    return true;
   }
   return super.onOptionsItemSelected(item);
}
/* Function to handle the click events,@param */
@Override
   public void onClick(View v){
   switch(v.getId()){
   case R.id.img_btn_play:
           if(checkFlag) {
             if(mediaPlayer.isPlaying()) {
                 mediaPlayer.pause();
                 imgBtnPlayPause.setImageResource(R.drawable.play_icon);
} else if(!mediaPlayer.isPlaying()){
          mediaPlayer.start();
          imgBtnPlayPause.setImageResource(R.drawable.pause_icon);
          playCycle();
       }
```

```
} else{
      Toast.makeText(this,"Select the Song ..",
Toast.LENGTH_SHORT).show();
}
break;
case R.id.btn_refresh:
    Toast.makeText(this,"Refreshing",Toast.LENGTH_SHORT).show();
    setPagerLayout();
    break;
case R.id.img_btn_replay:
     if(repeatFlag){
       Toast.makeText(this,"Replaying Removed..",
Toast.LENGTH_SHORT).show();
        mediaPlayer.setLooping(false);
        repeatFlag = false;
} else{
       Toast.makeText(this, "Replaying Added..",
Toast.LENGTH_SHORT).show();
       mediaPlayer.setLooping(true);
       repeatFlag = true;
}
break;
case R.id.img_btn_previous:
  if (checkFlag){
   if (mediaPlayer.getCurrentPosition() > 10) {
    if (currentPosition - 1 > -1) {
       attachMusic(songList.get(currentPosition - 1).getTitle(),
songList.get(currentPosition - 1).getPath());
       currentPosition = currentPosition - 1;
   } else{
```

```
            attachMusic(songList.get(currentPosition).getTitle(),
   songList.get(currentPosition).getPath());
        }
} else{
      attachMusic(songList.get(currentPosition).getTitle(),
   songList.get(currentPosition).getPath());
   }
} else{
       Toast.makeText(this, "Select a Song . .", Toast.LENGTH_SHORT).show();
}
break;
case R.id.img_btn_next:
if(checkFlag){
   if(currentPosition+1<songList.size()){attachMusic(songList.get(currentPosi
tion + 1).getTitle(), songList.get(currentPosition + 1).getPath());
      currentPosition += 1;
   } else{
        Toast.makeText(this, "Playlist Ended", Toast.LENGTH_SHORT).show();
}
} else{
       Toast.makeText(this, "Select the Song ..", Toast.LENGTH_SHORT).show();
}
break;
case R.id.img_btn_setting:
   if (!playContinuseFlag) {
      playContinueFlag = true;
      Toast.makeText(this, "Loop Added", Toast.LENGTH_SHORT).show();
   } else{
      playContinueFlag = false;
      Toast.makeText(this, "Loop Removed", Toast.LENGTH_SHORT).show();
```

```
    }
    break;
    }
}
/* Function to attach the song to the music player, @param name, @param
path*/
       private void attachMusic(String name, String path) {
           imgBtnPlayPause.setImageResource(R.drawable.play_icon);
           setTitle(name);
           menu.getItem(1).setIcon(R.drawable.favorite_icon);
           favFlag = true;
           try {
               mediaPlayer.reset();
               mediaPlayer.setDataSource(path);
               mediaPlayer.prepare();
               setControls();
           } catch (Exception e) {
               e.printStackTrace();
           }
           mediaPlayer.setOnCompletionListener(new
   MediaPlayer.OnCompletionListener() {
@Override
public void onCompletion(MediaPlayer mp) {
      imgBtnPlayPause.setImageResource(R.drawable.play_icon);
           if (playContinueFlag) {
                if (currentPosition + 1 < songList.size()) {
      attachMusic(songList.get(currentPosition        +        1).getTitle(),
   songList.get(currentPosition + 1).getPath());
                           currentPosition += 1;
                       } else {
```

```
                        Toast.makeText(MainActivity.this,        "PlayList
   Ended", Toast.LENGTH_SHORT).show();
                    }
                }
            }
        });
    }
   /* Function to set the controls according to the song*/
    private void setControls() {
        seekbarController.setMax(mediaPlayer.getDuration());
        mediaPlayer.start();
        playCycle();
        checkFlag = true;
        if (mediaPlayer.isPlaying()) {
            imgBtnPlayPause.setImageResource(R.drawable.pause_icon);
       tvTotalTime.setText(getTimeFormatted(mediaPlayer.getDuration()));
        }
   seekbarController.setOnSeekBarChangeListener(newSeekBar.OnSeekBarChangeLis
   tener() {
@Override
public  void  onProgressChanged(SeekBar  seekBar,  int  progress,  boolean
fromUser) {
   if (fromUser) {
                    mediaPlayer.seekTo(progress);
                    tvCurrentTime.setText(getTimeFormatted(progress));
                }
            }
   @Override
      public void onStartTrackingTouch(SeekBar seekBar) {}
   @Override
```

```
      public void onStopTrackingTouch(SeekBar seekBar) {}
           });
       }
/* Function to play the song using a thread*/
   private void playCycle() {
   try{
   seekbarController.setProgress(mediaPlayer.getCurrentPosition());
   tvCurrentTime.setText(getTimeFormatted(mediaPlayer.getCurrentPosition()));
               if (mediaPlayer.isPlaying()) {
                   runnable = new Runnable() {
                       @Override
                       public void run() {
                           playCycle();
                       }
                   };
                   handler.postDelayed(runnable, 100);
               }
           } catch (Exception e) {
               e.printStackTrace();
           }
       }
       private String getTimeFormatted(long milliSeconds) {
           String finalTimerString = "";
           String secondsString;
           //Converting total duration into time
           int hours = (int) (milliSeconds / 3600000);
           int minutes = (int) (milliSeconds % 3600000) / 60000;
           int seconds = (int) ((milliSeconds % 3600000) % 60000 / 1000);
           // Adding hours if any
           if (hours > 0)
```

```
                finalTimerString = hours + ":";
            // Prepending 0 to seconds if it is one digit
            if (seconds < 10)
                secondsString = "0" + seconds;
            else
                secondsString = "" + seconds;
            finalTimerString = finalTimerString + minutes + ":" +
   secondsString;
            // Return timer String;
            return finalTimerString;
        }
/*Function Overrided to receive the data from the fragment, @param name,
@param path*/
@Override
        public void onDataPass(String name, String path) {
            Toast.makeText(this, name, Toast.LENGTH_LONG).show();
            attachMusic(name, path);
        }
        @Override
        public void getLength(int length) {
            this.allSongLength = length;
        }
        @Override
        public void fullSongList(ArrayList<SongsList> songList, int position)
        {
            this.songList = songList;
            this.currentPosition = position;
            this.playlistFlag = songList.size() == allSongLength;
            this.playContinueFlag = !playlistFlag;
        }
```

```
@Override
public String queryText() {
    return searchText.toLowerCase();
}
@Override
public SongsList getSong() {
    currentPosition = -1;
    return currSong;
}
@Override
public boolean getPlaylistFlag() {
    return playlistFlag;
}
@Override
public void currentSong(SongsList songsList) {
    this.currSong = songsList;
}
@Override
public int getPosition() {
    return currentPosition;
}
@Override
protected void onResume() {
    super.onResume();
}
@Override
protected void onPause() {
    super.onPause();
}
@Override
```

```
    protected void onDestroy() {
        super.onDestroy();
        mediaPlayer.release();
        handler.removeCallbacks(runnable);
    }
}
```

- **SongAdapter.java**

```
package com.example.soc_macmini_15.musicplayer.Adapter;
import android.content.Context;
import android.graphics.Movie;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.Filterable;
import android.widget.TextView;
import com.example.soc_macmini_15.musicplayer.Model.SongsList;
import com.example.soc_macmini_15.musicplayer.R;
import java.util.ArrayList;
import java.util.List;
public class SongAdapter extends ArrayAdapter<SongsList> implements
Filterable{
  private Context mContext;
  private ArrayList<SongsList> songList = new ArrayList<>();
  public SongAdapter(Context mContext, ArrayList<SongsList> songList) {
     super(mContext, 0, songList);
     this.mContext = mContext;
```

```
        this.songList = songList;
      }
@NonNull
@Override
public View getView(int position, @Nullable View convertView, @NonNull
ViewGroup parent) {
        View listItem = convertView;
        if (listItem == null) {
            listItem =
            LayoutInflater.from(mContext).inflate(R.layout.playlist_items,
            parent, false);
         }
         SongsList currentSong = songList.get(position);
         TextView tvTitle = listItem.findViewById(R.id.tv_music_name);
         TextView tvSubtitle = listItem.findViewById(R.id.tv_music_subtitle);
         tvTitle.setText(currentSong.getTitle());
         tvSubtitle.setText(currentSong.getSubTitle());
         return listItem;
        }
     }
```

- **ViewPagerAdapter.java**

```
package com.example.soc_macmini_15.musicplayer.Adapter;
import android.content.ContentResolver;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentPagerAdapter;
import com.example.soc_macmini_15.musicplayer.Fragments.AllSongFragment;
import com.example.soc_macmini_15.musicplayer.Fragments.CurrentSongFragm
```

```
nt;
import com.example.soc_macmini_15.musicplayer.Fragments.FavSongFragment;
public class ViewPagerAdapter extends FragmentPagerAdapter {
    private ContentResolver contentResolver;
    private String title[] = {"All SONGS","CURRENT PLAYLIST","FAVORITES"};
    public ViewPagerAdapter(FragmentManager fm, ContentResolver
    contentResolver){
     super(fm);
     this.contentResolver = contentResolver;
 }
 @Override
 public Fragment getItem(int position) {
     switch (position) {
         case 0:
             return AllSongFragment.getInstance(position,contentResolver);
         case 1:
             return CurrentSongFragment.getInstance(position);
         case 2:
             return FavSongFragment.getInstance(position);
         default:
             return null;
     }
 }
 @Override
 public int getCount() {
     return title.length;
 }
 @Nullable
 @Override
 public CharSequence getPageTitle(int position) {
```

```
        return title[position];
    }
}
```

- **AllSongFragment**

```
package com.example.soc_macmini_15.musicplayer.Fragments;
import android.content.ContentResolver;
import android.content.Context;
import android.content.DialogInterface;
import android.database.Cursor;
import android.graphics.Color;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.support.v4.app.ListFragment;
import android.support.v7.app.AlertDialog;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
import com.example.soc_macmini_15.musicplayer.Adapter.SongAdapter;
import com.example.soc_macmini_15.musicplayer.Model.SongsList;
import com.example.soc_macmini_15.musicplayer.R;
import java.util.ArrayList;
```

```
public class AllSongFragment extends ListFragment {
  private static ContentResolver contentResolver1;
  public ArrayList<SongsList> songsList;
  public ArrayList<SongsList> newList;
  private ListView listView;
  private createDataParse createDataParse;
  private ContentResolver contentResolver;
  public static Fragment getInstance(int position, ContentResolver
  mcontentResolver) {
     Bundle bundle = new Bundle();
     bundle.putInt("pos", position);
     AllSongFragment tabFragment = new AllSongFragment();
     tabFragment.setArguments(bundle);
     contentResolver1 = mcontentResolver;
     return tabFragment;
   }
  @Override
  public void onCreate(@Nullable Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
  }
  @Override
  public void onAttach(Context context) {
     super.onAttach(context);
     createDataParse = (createDataParse) context;
  }
  @Override
  public View onCreateView(LayoutInflater inflater, ViewGroup container,
  Bundle savedInstanceState) {
     return inflater.inflate(R.layout.fragment_tab, container, false);
  }
```

```
    @Override
    public void onViewCreated(@NonNull View view, @Nullable Bundle
    savedInstanceState) {
       listView = view.findViewById(R.id.list_playlist);
       contentResolver = contentResolver1;
       setContent();
    }
/*Setting the content in the listView and sending the data to the Activity*/
    public void setContent() {
       boolean searchedList = false;
       songsList = new ArrayList<>();
       newList = new ArrayList<>();
       getMusic();
       SongAdapter adapter = new SongAdapter(getContext(), songsList);
       if (!createDataParse.queryText().equals("")) {
           adapter = onQueryTextChange();
           adapter.notifyDataSetChanged();
           searchedList = true;
       } else {
           searchedList = false;
       }
      createDataParse.getLength(songsList.size());
      listView.setAdapter(adapter);
      final boolean finalSearchedList = searchedList;
      listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
@Override
public void onItemClick(AdapterView<?> parent, View view, int position, long
id){
// Toast.makeText(getContext(), "You clicked :\n" + songsList.get(position),
Toast.LENGTH_SHORT).show();
```

```
if (!finalSearchedList) {
      createDataParse.onDataPass(songsList.get(position).getTitle(),
      songsList.get(position).getPath());
       createDataParse.fullSongList(songsList, position);
       } else {
       createDataParse.onDataPass(newList.get(position).getTitle(),
                  newList.get(position).getPath());
       createDataParse.fullSongList(songsList, position);
                 }
                }
            });
      listView.setOnItemLongClickListener(new
      AdapterView.OnItemLongClickListener() {
@Override
public boolean onItemLongClick(AdapterView<?> parent, View view, int
position, long id) {
                 showDialog(position);
                 return true;
              }
           });
        }
 public void getMusic() {
 Uri songUri = MediaStore.Audio.Media.EXTERNAL_CONTENT_URI;
 Cursor songCursor = contentResolver.query(songUri, null, null, null, null);
 if(songCursor != null && songCursor.moveToFirst()) {
            int songTitle =
             songCursor.getColumnIndex(MediaStore.Audio.Media.TITLE);
             int songArtist =
             songCursor.getColumnIndex(MediaStore.Audio.Media.ARTIST);
             int songPath =
```

```
            songCursor.getColumnIndex(MediaStore.Audio.Media.DATA);
            do {
                songsList.add(new SongsList(songCursor.getString(songTitle),
                songCursor.getString(songArtist),
                songCursor.getString(songPath)));
            } while (songCursor.moveToNext());
            songCursor.close();
           }
     }
     public SongAdapter onQueryTextChange() {
        String text = createDataParse.queryText();
        for (SongsList songs : songsList) {
            String title = songs.getTitle().toLowerCase();
            if (title.contains(text)) {
                newList.add(songs);
            }
        }
        return new SongAdapter(getContext(), newList);
    }
private void showDialog(final int position) {
AlertDialog.Builder builder = new AlertDialog.Builder(getContext());
builder.setMessage(getString(R.string.play_next))
.setCancelable(true)
.setNegativeButton(R.string.no, new DialogInterface.OnClickListener() {
@Override
     public void onClick(DialogInterface dialog, int which) {}
  })
.setPositiveButton(R.string.yes, new DialogInterface.OnClickListener() {
@Override
     public void onClick(DialogInterface dialog, int which) {
```

```
                    createDataParse.currentSong(songsList.get(position));
                    setContent();
                }
            });
        AlertDialog alertDialog = builder.create();
        alertDialog.show();
    }
    public interface createDataParse {
    public void onDataPass(String name, String path);
    public void fullSongList(ArrayList<SongsList> songList, int position);
    public String queryText();
    public void currentSong(SongsList songsList);
    public void getLength(int length);
     }
}
```

- **CurrentSongFragment**

```
package com.example.soc_macmini_15.musicplayer.Fragments;
import android.content.ContentResolver;
import android.content.Context;
import android.content.DialogInterface;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.provider.MediaStore;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.support.v4.app.ListFragment;
import android.support.v7.app.AlertDialog;
```

```
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.Toast;
import com.example.soc_macmini_15.musicplayer.Adapter.SongAdapter;
import com.example.soc_macmini_15.musicplayer.DB.FavoritesOperations;
import com.example.soc_macmini_15.musicplayer.Model.SongsList;
import com.example.soc_macmini_15.musicplayer.R;
import java.util.ArrayList;
public class CurrentSongFragment extends ListFragment {
public ArrayList<SongsList> songsList = new ArrayList<>();
private ListView listView;
private createDataParsed createDataParsed;
public static Fragment getInstance(int position) {
Bundle bundle = new Bundle();
     bundle.putInt("pos", position);
     CurrentSongFragment tabFragment = new CurrentSongFragment();
     tabFragment.setArguments(bundle);
     return tabFragment;
 }
 @Override
 public void onCreate(@Nullable Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
 }
 @Override
 public void onAttach(Context context) {
     super.onAttach(context);
     createDataParsed = (createDataParsed) context;
```

```
    }
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_tab, container, false);
    }
 @Override
    public void onViewCreated(@NonNull View view, @Nullable Bundle
    savedInstanceState) {
        listView = view.findViewById(R.id.list_playlist);
        //songsList = new ArrayList<>();
        setContent();
     }
/* Setting the content in the listView and sending the data to the Activity*/
    public void setContent() {
        if (createDataParsed.getSong() != null)
            songsList.add(createDataParsed.getSong());
        SongAdapter adapter = new SongAdapter(getContext(), songsList);
        if (songsList.size() > 1)
            if (createDataParsed.getPlaylistFlag()) {
                songsList.clear();
            }
       listView.setAdapter(adapter);
       adapter.notifyDataSetChanged();
       listView.setOnItemClickListener(new AdapterView.OnItemClickListener(){
@Override
public void onItemClick(AdapterView<?>parent,View view,int position,long id){
                // Toast.makeText(getContext(), "You clicked :\n" +
             songsList.get(position), Toast.LENGTH_SHORT).show();
             createDataParsed.onDataPass(songsList.get(position).getTitle(),
```

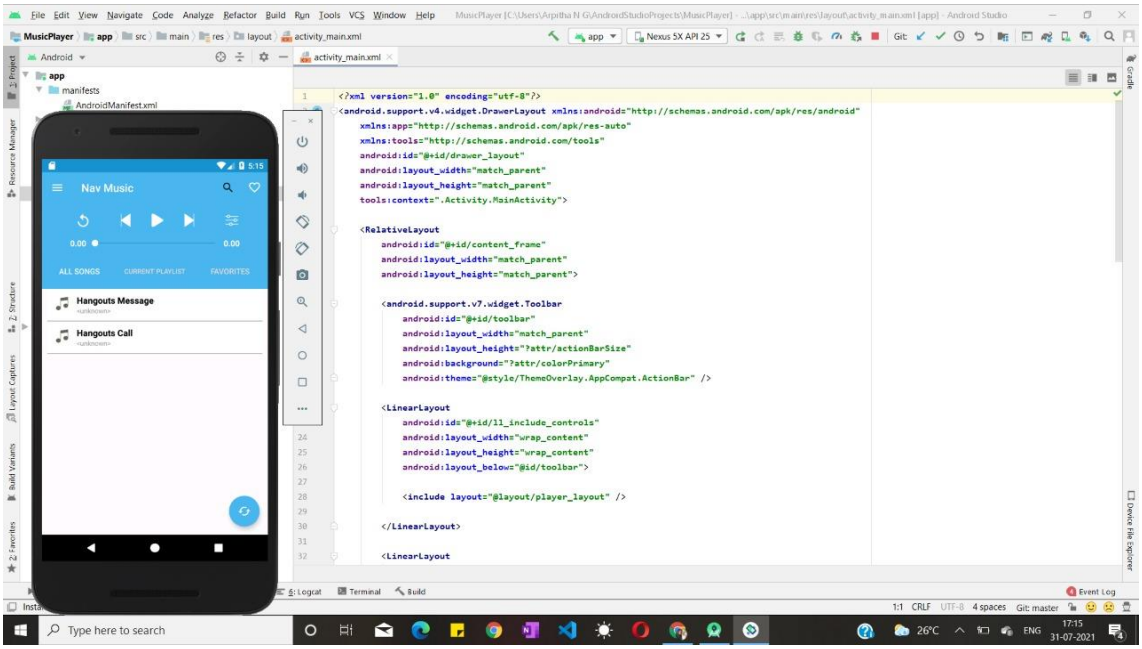
```
            songsList.get(position).getPath());
               createDataParsed.fullSongList(songsList, position);
            }
        });
        listView.setOnItemLongClickListener(new
        AdapterView.OnItemLongClickListener() {
@Override
public boolean onItemLongClick(AdapterView<?> parent,View view,int
position,long id) {
               return true;
            }
        });
     }
 public interface createDataParsed {
 public void onDataPass(String name, String path);
 public void fullSongList(ArrayList<SongsList> songList, int position);
 public SongsList getSong();
 public boolean getPlaylistFlag();
      }
   }
```

- **SongsList**

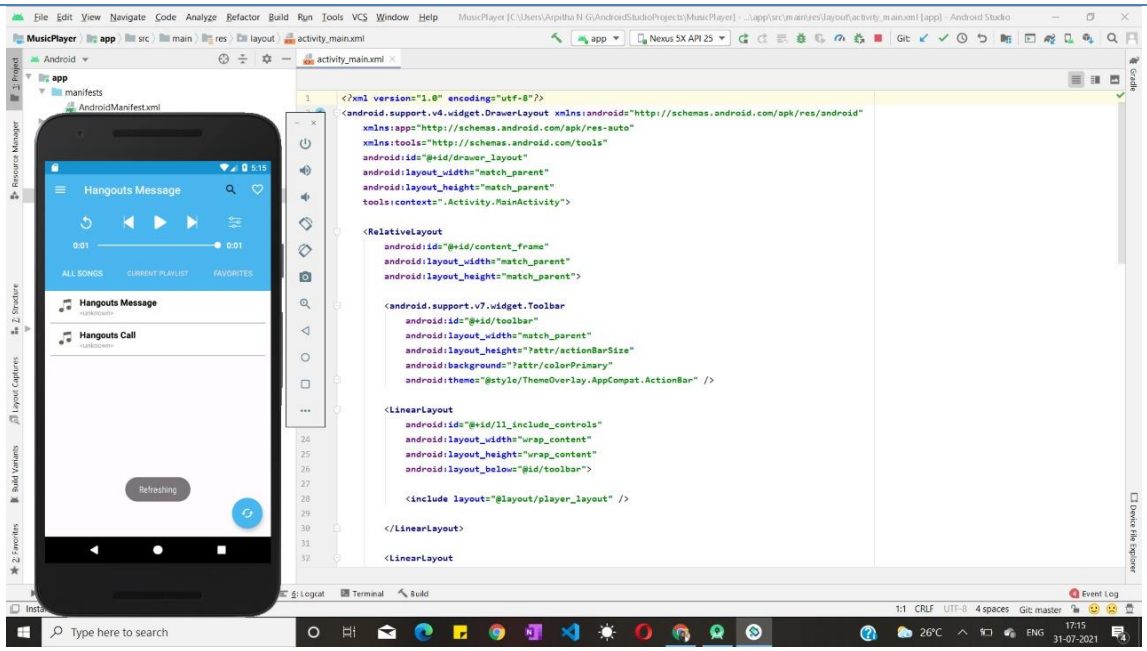
```
package com.example.soc_macmini_15.musicplayer.Model;
public class SongsList {
 private String title;
 private String subTitle;
 private String path;
 public String getPath() {
     return path;
 }
```

```
    public void setPath(String path) {
        this.path = path;
    }
    public SongsList(String title, String subTitle, String path) {
        this.title = title;
        this.subTitle = subTitle;
        this.path = path;
    }
    public String getTitle() {
        return title;
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public String getSubTitle() {
        return subTitle;
    }
   }
```

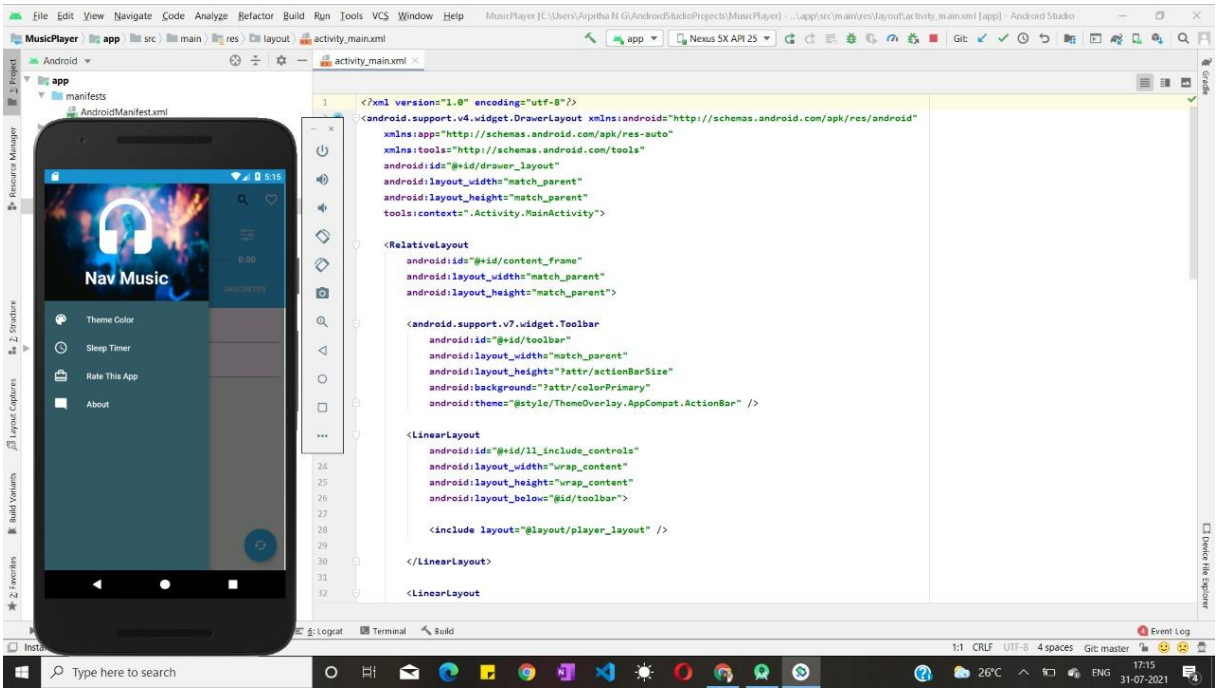
RESULTS



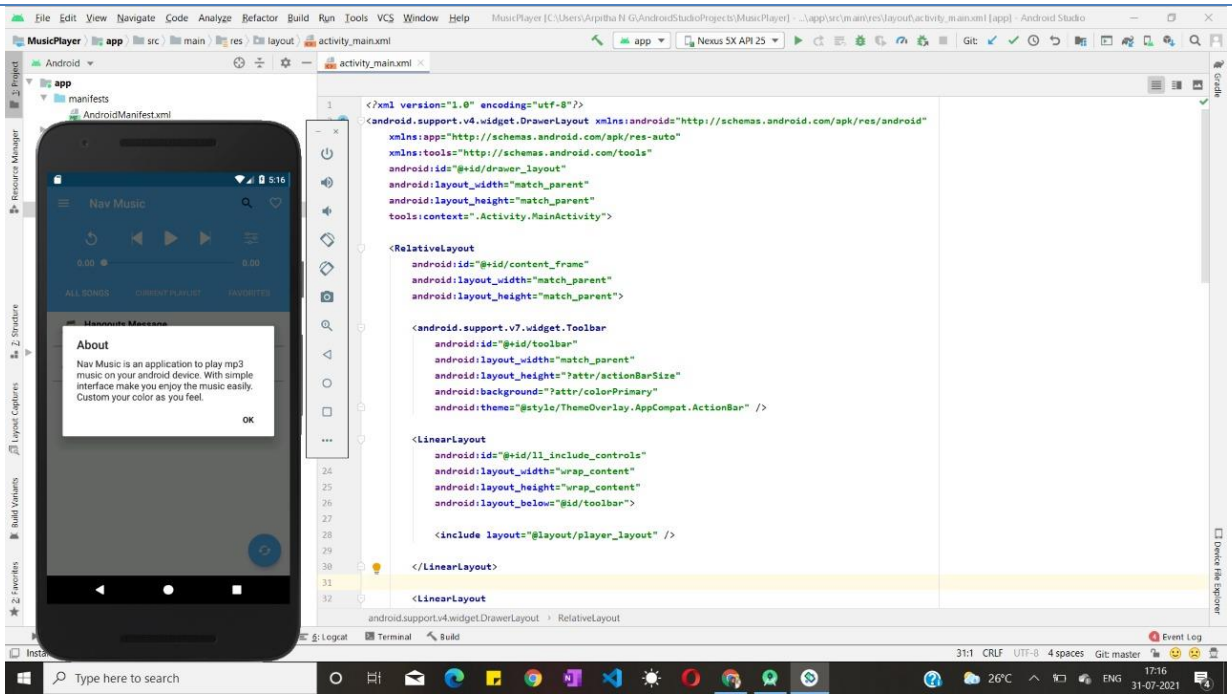
Snapshot 1: PlayList



Snapshot 2 : Refreshing



Snapshot 3 : Dashboard



Snapshot 4 : About pop-up