DEVELOP A MOBLIE APPLICATION TO USE THE MOBILE APPLICATION

AIM:

To develop a mobile application to display music application in web view.

PROCEDURE:

1. Create a new android application by choosing the application name,minimum required SDK to be API 19 : android 4.4(Kitkat) and create a blank activity.
2. Design the activity with the music view component .
3. In the main Activity . java file ,write the following code to display images:
   Webview .loadUrl("favorite web app url");
4. To access the internet ,add the following permission in the manifest file.
   < uses-permission android name="android.permission .INTERNET">
5. Start and launch the emulator.
6. Right click the application name in the package explorer window and click
   Run as android application. After successful compilation, the projectname.apk file will be launched and executed on the emulator

OBJECTIVE:

The main focus of Shuffler is to Shuffle a user's music playlist in such a way that the next song that gets played is what the user will most probably want to listen. Various factors are taken into account while shuffling a saved playlist like playback count, Average playing duration, Artist, Genre, Release date and like count from a particular user.

Activity main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.anujsharma.shuffler">


    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"
/>
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />


    <application
        android:allowBackup="true"
        android:icon="@drawable/lg_shuffle"
        android:label="@string/app_name"
        android:roundIcon="@drawable/lg_shuffle"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".activities.MainActivity"
            android:label="@string/app_name"
            android:screenOrientation="portrait"
            android:theme="@style/AppTheme.NoActionBar"
            android:windowSoftInputMode="adjustNothing">


        </activity>
        <activity
            android:name=".activities.ViewSongActivity"
            android:screenOrientation="portrait"
            android:theme="@style/viewSongTheme"
            android:windowSoftInputMode="adjustNothing" />

        <meta-data
            android:name="com.example.anujsharma.shuffler.utilities.MyGlideModule"
            android:value="GlideModule" />

        <service
            android:name=".services.MusicService"
            android:enabled="true" />
        <service
            android:name=".services.ExoPlayerService"
            android:stopWithTask="true" />

        <receiver android:name=".receivers.NotificationBroadcast">
            <intent-filter>
                <action android:name="com.example.anujsharma.shuffler.utilities.nextClick" />
                <action android:name="com.example.anujsharma.shuffler.utilities.previousClick"
/>
                <action android:name="com.example.anujsharma.shuffler.utilities.playPauseClick"
/>
            </intent-filter>
        </receiver>

        <activity
            android:name=".activities.SplashScreenActivity"
            android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

SOURCE CODE:

MainActivity.java:

```
package com.example.anujsharma.shuffler.activities;
import android.annotation.SuppressLint;
import android.content.ComponentName;
import android.content.Context;
import android.content.Intent;
import android.content.ServiceConnection;
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Bundle;
import android.os.IBinder;
import android.support.annotation.NonNull;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentTransaction;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import com.example.anujsharma.shuffler.R;
import com.example.anujsharma.shuffler.dao.TracksDao;
import com.example.anujsharma.shuffler.fragments.HomeFragment;
import com.example.anujsharma.shuffler.fragments.SearchFragment;
import com.example.anujsharma.shuffler.fragments.YourLibraryFragment;
import com.example.anujsharma.shuffler.models.Playlist;
import com.example.anujsharma.shuffler.models.Song;
import com.example.anujsharma.shuffler.services.ExoPlayerService;
import com.example.anujsharma.shuffler.services.MusicService;
import com.example.anujsharma.shuffler.utilities.Constants;
import com.example.anujsharma.shuffler.utilities.FisherYatesShuffle;
import  com.example.anujsharma.shuffler.utilities.SharedPreference;
import com.example.anujsharma.shuffler.volley.RequestCallback;
import  com.example.anujsharma.shuffler.volley.Urls;
import com.google.android.exoplayer2.util.Util;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity implements RequestCallback {

    public static final String TAG = "TAG";
    public static final String HOME_FRAGMENT = "homeFragment";
    public static final String SEARCH_FRAGMENT = "searchFragment";
    public static final String YOUR_LIBRARY_FRAGMENT = "yourLibraryFragment";

    //service
    public static MusicService musicSrv;
    private final int REQUEST_PERMS_CODE = 1;
    SharedPreference pref;
    private boolean initHomeFragment, initSearchFragment, initYourLibraryFragment;
    private HomeFragment homeFragment;
    private SearchFragment searchFragment;
    private YourLibraryFragment yourLibraryFragment;
    //   private MediaPlayer mediaPlayer;
```

```
private Context context;
private ProgressBar mainSongLoader;
private View progressView;
private TextView tvHome, tvSearch, tvMyProfile, tvSongName;
private ImageView ivPlay, ivNext, ivFullView;
private TracksDao tracksDao;
private int currentSongPosition;
private Playlist currentPlaylist;
private Intent playIntent, exoIntent;
//binding
private boolean musicBound = false;
//connect to the service
private ServiceConnection musicConnection = new ServiceConnection() {

  @Override
  public void onServiceConnected(ComponentName name, IBinder service) {
    MusicService.MusicBinder binder = (MusicService.MusicBinder) service;
    //get service
    musicSrv = binder.getService();
    musicBound = true;
    musicSrv.setCallbacks(new MusicService.MusicServiceInterface() {
      @Override
      public void onMusicDisturbed(int state, Song song) {
        switch (state) {
          case Constants.MUSIC_STARTED:
            ivPlay.setClickable(true);
            ivPlay.setImageDrawable(getResources().getDrawable(R.drawable.ic_pause));
            break;
          case Constants.MUSIC_PLAYED:
            ivPlay.setImageDrawable(getResources().getDrawable(R.drawable.ic_pause));
            break;
```

```
                case Constants.MUSIC_PAUSED:
                    ivPlay.setImageDrawable(getResources().getDrawable(R.drawable.ic_play));
                    break;
                case Constants.MUSIC_ENDED:
                    ivPlay.setImageDrawable(getResources().getDrawable(R.drawable.ic_play));
                    break;
                case Constants.MUSIC_LOADED:
                    ivPlay.setClickable(false);
                    tvSongName.setText(song.getTitle());
                    ivPlay.setImageDrawable(getResources().getDrawable(R.drawable.ic_pause));
                    break;
            }
        }

        @Override
        public void onSongChanged(int newPosition) {
            currentSongPosition = newPosition;
        }

        @Override
        public void onMusicProgress(int position) {
            progressView.getBackground().setLevel(position);
        }
    });
}

@Override
public void onServiceDisconnected(ComponentName name) {
    musicBound = false;
}
};
```

```
   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_main);

      /*Intent splashIntent = new Intent(MainActivity.this, SplashScreenActivity.class);
      startActivity(splashIntent);*/

       /*if (playIntent == null) {
           playIntent = new Intent(getBaseContext(), MusicService.class);
           bindService(playIntent, musicConnection, Context.BIND_AUTO_CREATE);
           startService(playIntent);
       }*/

       initialise();
       initialiseListeners();
       if (hasPermissons()) {
           mainStuff();
       } else {
           requestPermissions();
       }

       /*Intent notificationIntent = getIntent();
       boolean fromNotification = notificationIntent.getBooleanExtra(Constants.FROM_NOTIFICATION,
false);
       if (fromNotification) {


           currentPlaylist = notificationIntent.getParcelableExtra(Constants.PLAYLIST_MODEL_KEY);
           currentSongPosition =
notificationIntent.getIntExtra(Constants.CURRENT_PLAYING_SONG_POSITION, 0);
           boolean isPlaying = notificationIntent.getBooleanExtra(Constants.IS_PLAYING, true);
```

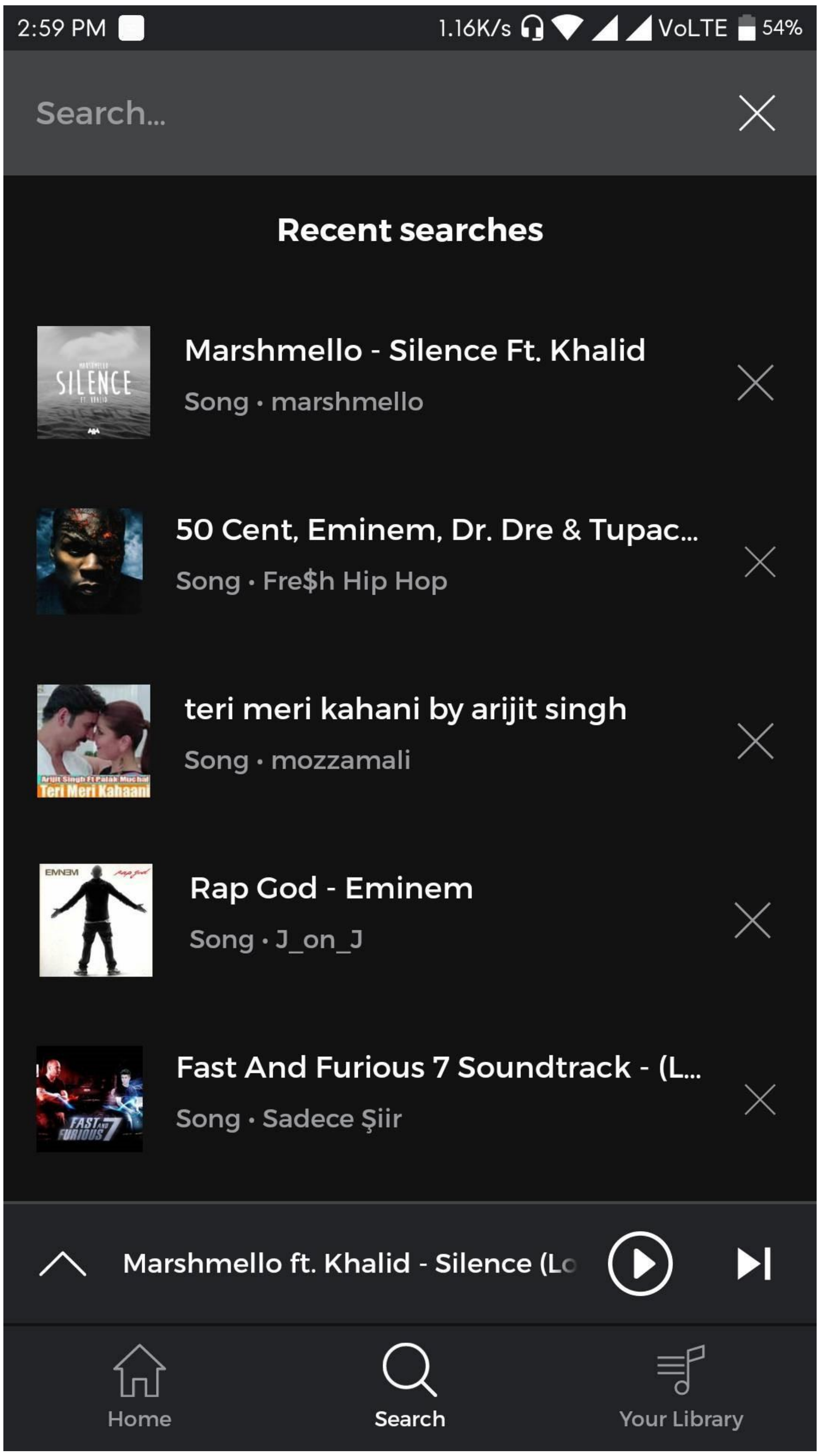
```
            Intent intent = new Intent(context, ViewSongActivity.class);
            intent.putExtra(Constants.PLAYLIST_MODEL_KEY, currentPlaylist);
            intent.putExtra(Constants.CURRENT_PLAYING_SONG_POSITION, currentSongPosition);
            intent.putExtra(Constants.IS_PLAYING, isPlaying);
            context.startActivity(intent);
        }*/

        exoIntent = new Intent(this, ExoPlayerService.class);
        exoIntent.putExtra(Constants.PLAYLIST_MODEL_KEY, pref.getCurrentPlaylist());
        exoIntent.putExtra(Constants.SONG_POSITION_MODEL_KEY,
pref.getCurrentPlayingSongPosition());
        Util.startForegroundService(this, exoIntent);
    }

    public void playSongInMainActivity(int songPosition, Playlist playlist) {
        if (playlist.getSongs() == null || playlist.getSongs().size() == 0) {
            Toast.makeText(context, "Unable to play this Playlist.", Toast.LENGTH_SHORT).show();
            return;
        }
        Song song = playlist.getSongs().get(songPosition);
        currentSongPosition = songPosition;
        pref.setCurrentPlayingSong(song.getId());
        pref.setCurrentPlaylist(playlist);
        pref.setCurrentPlayingSongPosition(songPosition);
        tvSongName.setText(song.getTitle());
        this.currentPlaylist = playlist;
        ArrayList<Integer> shuffleList = new ArrayList<>();
        for (int i = 0; i < playlist.getSongs().size(); i++) shuffleList.add(i);
        pref.setCurrentPlaylistShuffleArray(shuffleList);
        pref.setCurrentShuffleSongPosition(0);
        FisherYatesShuffle.updateShuffleList(context, songPosition);



        String url = song.getStreamUrl() + "?client_id=" + Urls.CLIENT_ID;
        Log.d("TAG", "currently playing " + url);

        /*musicSrv.setSongPosition(songPosition);
        musicSrv.setSongs(playlist.getSongs());
        musicSrv.setPlaylist(playlist);
        musicSrv.startSong();*/
    }

    public void updatePlaylistInMainActivity(Playlist playlist) {
        if (playlist.getPlaylistId() == currentPlaylist.getPlaylistId()) {
            currentPlaylist = playlist;
            pref.setCurrentPlaylist(playlist);
        }
    }

    private void initialiseListeners() {

        ivPlay.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                currentPlaylist = pref.getCurrentPlaylist();
                if (currentPlaylist != null) {
                    musicSrv.setPlaylist(currentPlaylist);
                    if (musicSrv.isPlaying()) {
                        musicSrv.pausePlayer();
                    } else {
                        musicSrv.go();
                    }
                }
            }
        });

        ivNext.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                currentPlaylist = pref.getCurrentPlaylist();
                if (currentPlaylist != null) {
                    musicSrv.setPlaylist(currentPlaylist);
                    musicSrv.playNext();
                }
            }
        });

        ivFullView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (currentPlaylist != null) {
                    Intent intent = new Intent(context, ViewSongActivity.class);
                    intent.putExtra(Constants.PLAYLIST_MODEL_KEY, currentPlaylist);
                    intent.putExtra(Constants.CURRENT_PLAYING_SONG_POSITION, currentSongPosition);
                    intent.putExtra(Constants.IS_PLAYING, musicSrv.isPlaying());
                    context.startActivity(intent);
                }
            }
        });

        tvSongName.setOnClickListener(new View.OnClickListener() {
            @Override
```

```
            public void onClick(View v) {
                if (currentPlaylist != null) {
                    Intent intent = new Intent(context, ViewSongActivity.class);
                    intent.putExtra(Constants.PLAYLIST_MODEL_KEY, currentPlaylist);
                    intent.putExtra(Constants.CURRENT_PLAYING_SONG_POSITION, currentSongPosition);
                    if (musicSrv != null)
                        intent.putExtra(Constants.IS_PLAYING, musicSrv.isPlaying());
                    else intent.putExtra(Constants.IS_PLAYING, false);
                    context.startActivity(intent);
                }
            }
        });
    }

    public void initialise() {
        context = getApplicationContext();
        pref = new SharedPreference(context);
        tracksDao = new TracksDao(context, this);

        tvHome = findViewById(R.id.xtvHome);
        tvSearch = findViewById(R.id.xtvSearch);
        tvMyProfile = findViewById(R.id.xtvMyProfile);
        tvSongName = findViewById(R.id.tvSongName);
        mainSongLoader = findViewById(R.id.pbLoadSong);
        tvSongName.setSelected(true);
        ivFullView = findViewById(R.id.ivUpArrow);
        ivNext = findViewById(R.id.ivPlayNext);
        ivPlay = findViewById(R.id.ivPlaySong);
        progressView = findViewById(R.id.progressView);

        currentPlaylist = pref.getCurrentPlaylist();
        currentSongPosition = pref.getCurrentPlayingSongPosition();
        if (currentPlaylist != null) {
            Song currentSong = currentPlaylist.getSongs().get(currentSongPosition);
            tvSongName.setText(currentSong.getTitle());
        }
    }



    public void modifyBottomLayout(int position) {
        switch (position) {
            case 0:
                tvHome.setCompoundDrawablesWithIntrinsicBounds(0, R.drawable.ic_home_selected, 0,
0);
                tvHome.setTextColor(ContextCompat.getColor(context, R.color.white));
                tvSearch.setCompoundDrawablesWithIntrinsicBounds(0, R.drawable.ic_search, 0, 0);
                tvSearch.setTextColor(ContextCompat.getColor(context, R.color.color_unselected));
                tvMyProfile.setCompoundDrawablesWithIntrinsicBounds(0, R.drawable.ic_library, 0, 0);
                tvMyProfile.setTextColor(ContextCompat.getColor(context, R.color.color_unselected));
                break;
            case 1:
                tvHome.setCompoundDrawablesWithIntrinsicBounds(0, R.drawable.ic_home, 0, 0);
                tvHome.setTextColor(ContextCompat.getColor(context, R.color.color_unselected));
                tvSearch.setCompoundDrawablesWithIntrinsicBounds(0, R.drawable.ic_search_selected, 0,
0);
                tvSearch.setTextColor(ContextCompat.getColor(context, R.color.white));
                tvMyProfile.setCompoundDrawablesWithIntrinsicBounds(0, R.drawable.ic_library, 0, 0);
                tvMyProfile.setTextColor(ContextCompat.getColor(context, R.color.color_unselected));
                break;
            case 2:
                tvHome.setCompoundDrawablesWithIntrinsicBounds(0, R.drawable.ic_home, 0, 0);
                tvHome.setTextColor(ContextCompat.getColor(context, R.color.color_unselected));
                tvSearch.setCompoundDrawablesWithIntrinsicBounds(0, R.drawable.ic_search, 0, 0);
            tvSearch.setTextColor(ContextCompat.getColor(context, R.color.color_unselected));
            tvMyProfile.setCompoundDrawablesWithIntrinsicBounds(0, R.drawable.ic_library_selected, 
0, 0);
            tvMyProfile.setTextColor(ContextCompat.getColor(context, R.color.white));
            break;
      }
   }

   public void homeTabClicked(View view) {
      if (!initHomeFragment) {
         initHomeFragment = true;
         homeFragment = new HomeFragment();
      }
      addFragmentToMainFrameContainer(homeFragment, HOME_FRAGMENT);
   }

   public void searchTabClicked(View view) {
      if (!initSearchFragment) {
         initSearchFragment = true;
         searchFragment = new SearchFragment();
      }
      addFragmentToMainFrameContainer(searchFragment, SEARCH_FRAGMENT);
   }

   public void yourLibraryClicked(View view) {
      if (!initYourLibraryFragment) {
         initYourLibraryFragment = true;
         yourLibraryFragment = new YourLibraryFragment();
      }
      addFragmentToMainFrameContainer(yourLibraryFragment, YOUR_LIBRARY_FRAGMENT);
   }
```

```
  public void addFragmentToMainFrameContainer(Fragment fragment, String TAG) {
    FragmentManager fragmentManager = getSupportFragmentManager();
    Fragment currentFragment = fragmentManager.findFragmentById(R.id.mainFrameContainer);
    if (currentFragment != null && currentFragment.getClass().equals(fragment.getClass())) {

    } else {
      FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
      fragmentTransaction.replace(R.id.mainFrameContainer,
fragment).addToBackStack(null).commit();
    }
  }

  @Override
  public void onBackPressed() {
    if (getSupportFragmentManager().getBackStackEntryCount() == 1) {
      finish();
    } else {
      super.onBackPressed();
    }
  }

  /*@Override
  public boolean onKeyDown(int keyCode, KeyEvent event) {
    if(keyCode == KeyEvent.KEYCODE_HEADSETHOOK){
      Toast.makeText(context, "Headset button clicked", Toast.LENGTH_SHORT).show();
      return true;
    }
    return super.onKeyDown(keyCode, event);
  }*/
```

```
    public void mainStuff() {
        /*File rootFile = new File(Environment.getExternalStorageDirectory().getPath()*//* +
"/SHAREit/files/audios/"*//*);
        fetchSongFilesTask = new FetchSongFilesTask(this);
        fetchSongFilesTask.execute(rootFile);
        layoutManager = new LinearLayoutManager(this);
        mainRecyclerViewAdapter = new MainRecyclerViewAdapter(this, mediaPlayer);
        mainRecyclerView.setLayoutManager(layoutManager);
        mainRecyclerView.setAdapter(mainRecyclerViewAdapter);*/

        HomeFragment fragment = new HomeFragment();
        getSupportFragmentManager().beginTransaction().replace(R.id.mainFrameContainer, fragment,
HOME_FRAGMENT)
                .addToBackStack(null).commit();
    }

    @SuppressLint("WrongConstant")
    private boolean hasPermissons() {
        int res = 0;

        String[] permissions = {android.Manifest.permission.READ_EXTERNAL_STORAGE};
        for (String permission : permissions) {
            res = checkCallingOrSelfPermission(permission);
            if (res != PackageManager.PERMISSION_GRANTED) {
                return false;
            }
        }
        return true;
    }

    private void requestPermissions() {
        String[] permissions = {android.Manifest.permission.READ_EXTERNAL_STORAGE};
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            requestPermissions(permissions, REQUEST_PERMS_CODE);
        }
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {
        boolean allowed = true;

        switch (requestCode) {
            case REQUEST_PERMS_CODE:
                for (int res : grantResults) {
                    allowed = allowed && (res == PackageManager.PERMISSION_GRANTED);
                }
                break;
            default:
                allowed = false;
        }

        if (allowed) {
            mainStuff();
        } else {
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
                if
(shouldShowRequestPermissionRationale(android.Manifest.permission.READ_EXTERNAL_STORAGE))
{
                    Toast.makeText(this, "Storage read permission denied. Music won't be shown if
permission is denied", Toast.LENGTH_SHORT).show();
                    requestPermissions();
                }
            }
        }
    }


    @Override
    public void onListRequestSuccessful(ArrayList list, int check, boolean status) {

    }

    @Override
    protected void onStop() {
        super.onStop();
        if (currentPlaylist != null)
            pref.setCurrentPlayingSong(currentPlaylist.getSongs().get(currentSongPosition).getId());
        if (musicSrv != null) pref.setCurrentPlayingSongPosition(musicSrv.getSongPosition());
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
//        stopService(exoIntent);
        /*musicSrv = null;
        unbindService(musicConnection);*/
    }

    @Override
    public void onObjectRequestSuccessful(Object object, int check, boolean status) {
    }
}
```

OUTPUT:


2:59 PM 1.16K/s VoLTE 54%

Search...

Recent searches

Marshmello - Silence Ft. Khalid
Song • marshmello

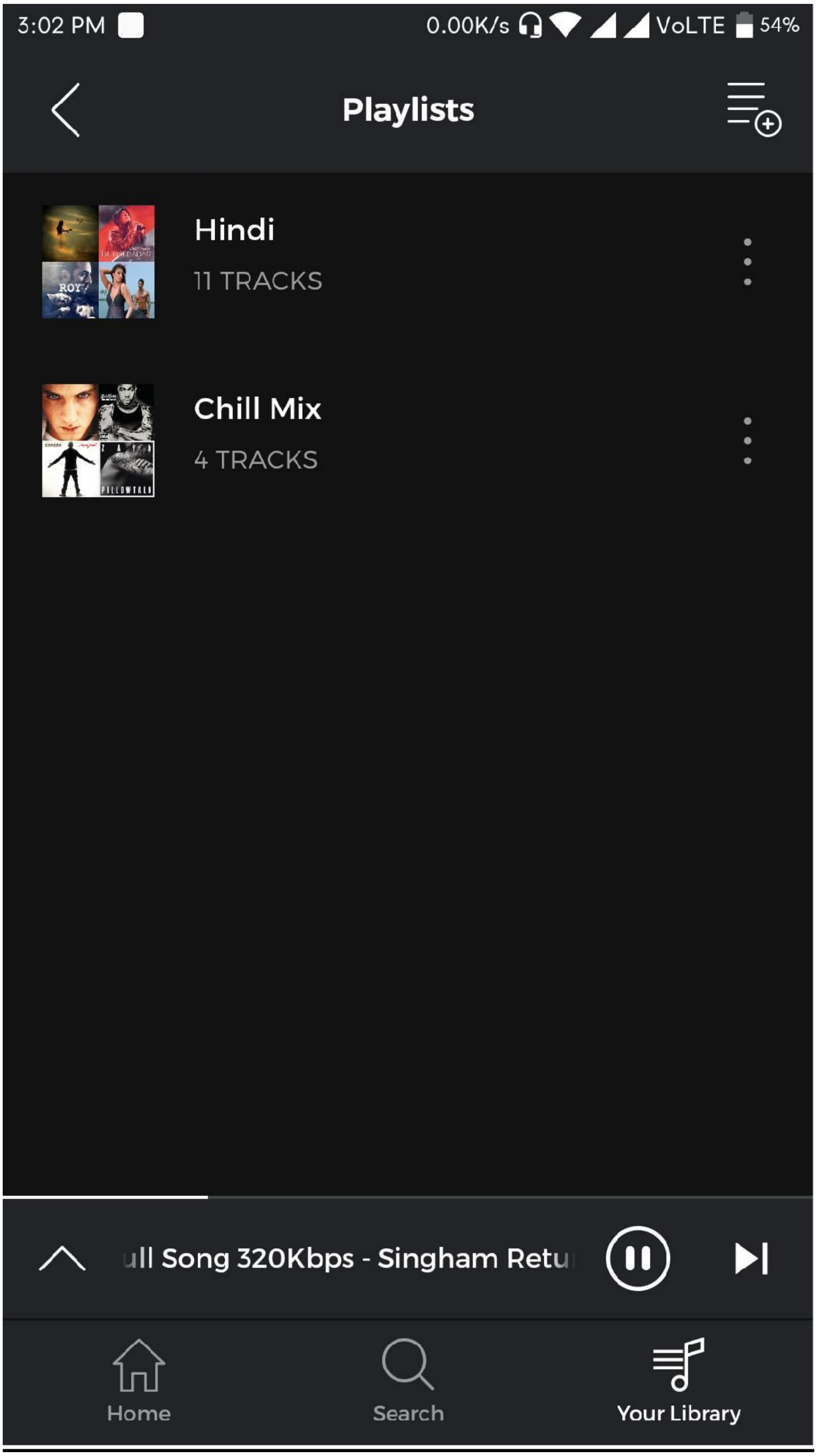
50 Cent, Eminem, Dr. Dre & Tupac...
Song • Fre$h Hip Hop

teri meri kahani by arijit singh
Song • mozzamali

Rap God - Eminem
Song • J_on_J

Fast And Furious 7 Soundtrack - (L...
Song • Sadece Şiir

Marshmello ft. Khalid - Silence (Lo


3:02 PM 0.00K/s VoLTE 54%

Playlists

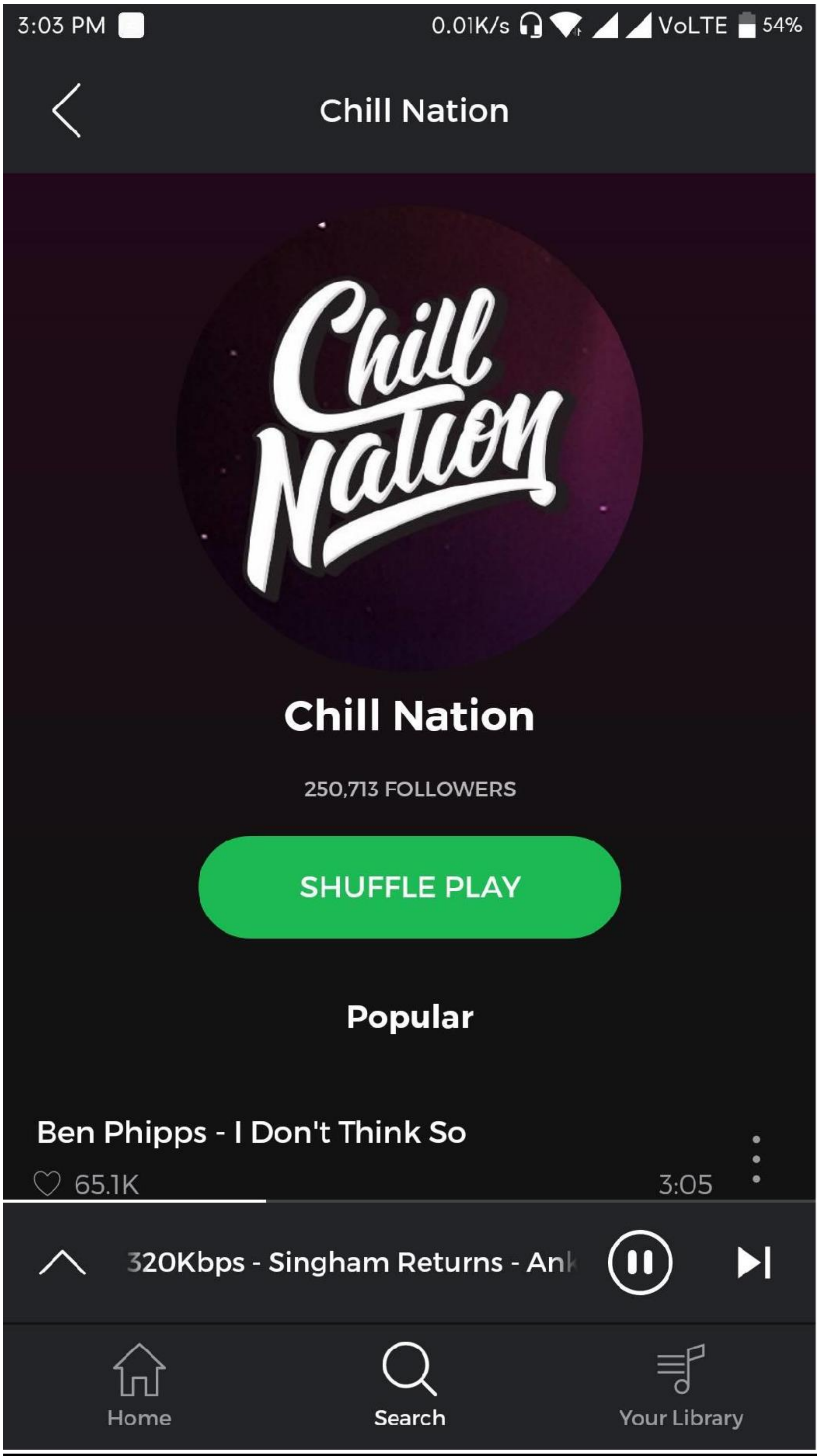
Hindi
11 TRACKS

Chill Mix
4 TRACKS

ull Song 320Kbps - Singham Retu

211520104059

JANAKIRAMAN S



3:03 PM 0.01K/s VoLTE 54%

Chill Nation

Chill Nation

250,713 FOLLOWERS

SHUFFLE PLAY

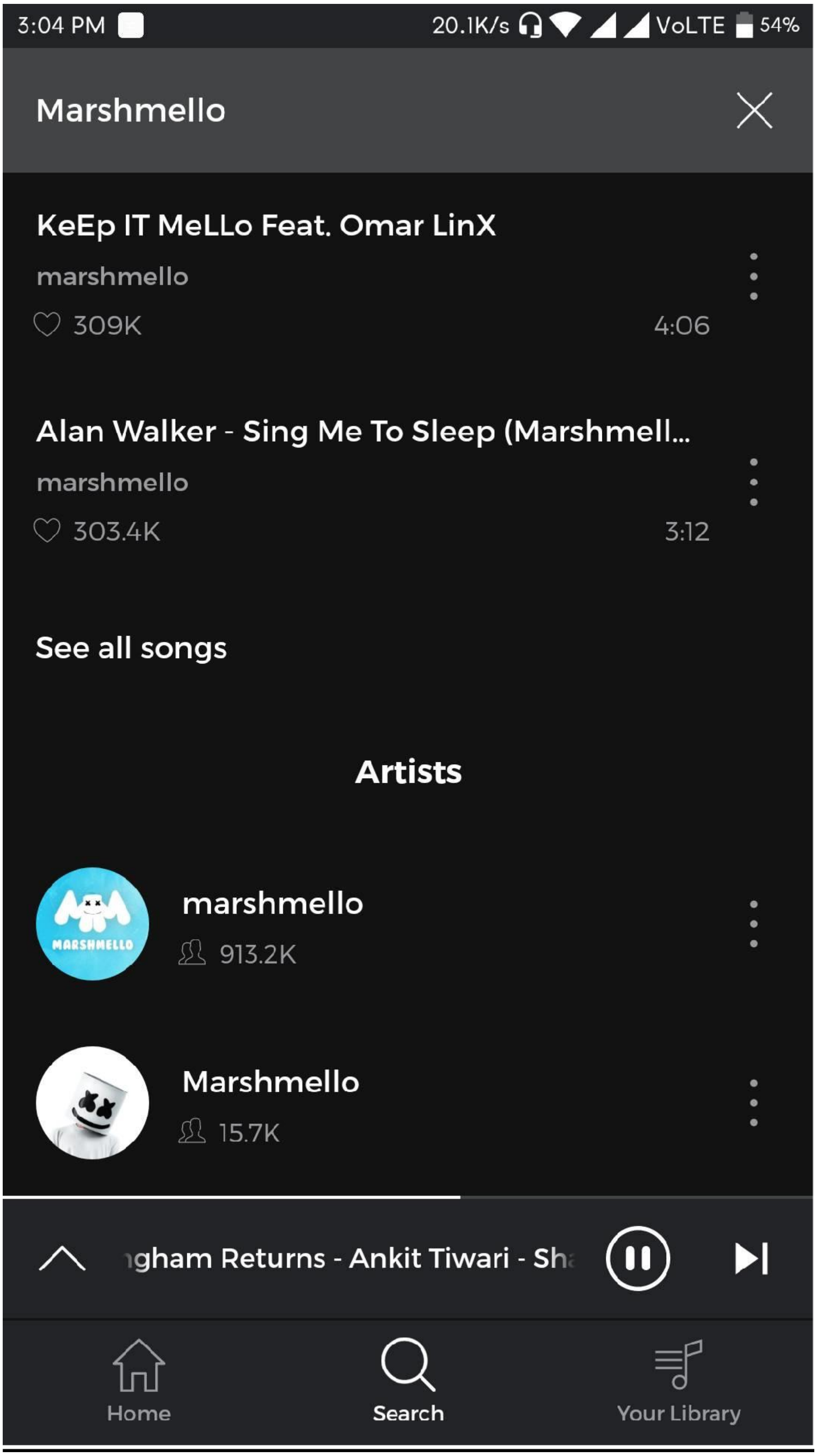
Popular

Ben Phipps - I Don't Think So

65.1K 3:05

320Kbps - Singham Returns - Ank

Home Search Your Library


3:04 PM 20.1K/s VoLTE 54%

Marshmello

KeEp IT MeLLo Feat. Omar LinX
marshmello
309K 4:06

Alan Walker - Sing Me To Sleep (Marshmell...
marshmello
303.4K 3:12

See all songs

Artists

marshmello
913.2K

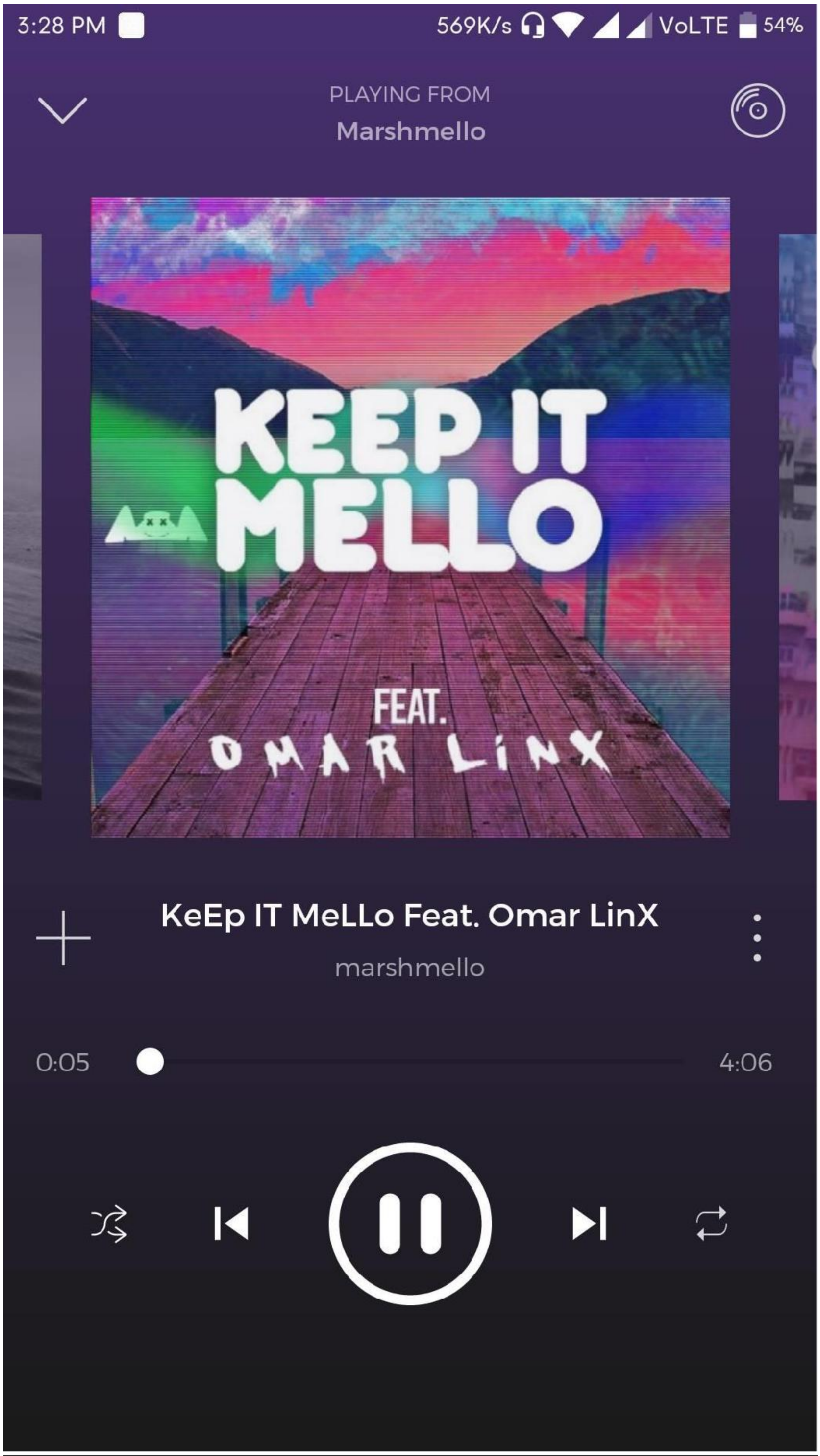
Marshmello
15.7K

ıgham Returns - Ankit Tiwari - Sh

Home Search Your Library

211520104059

JANAKIRAMAN S


3:28 PM 569K/s VoLTE 54%

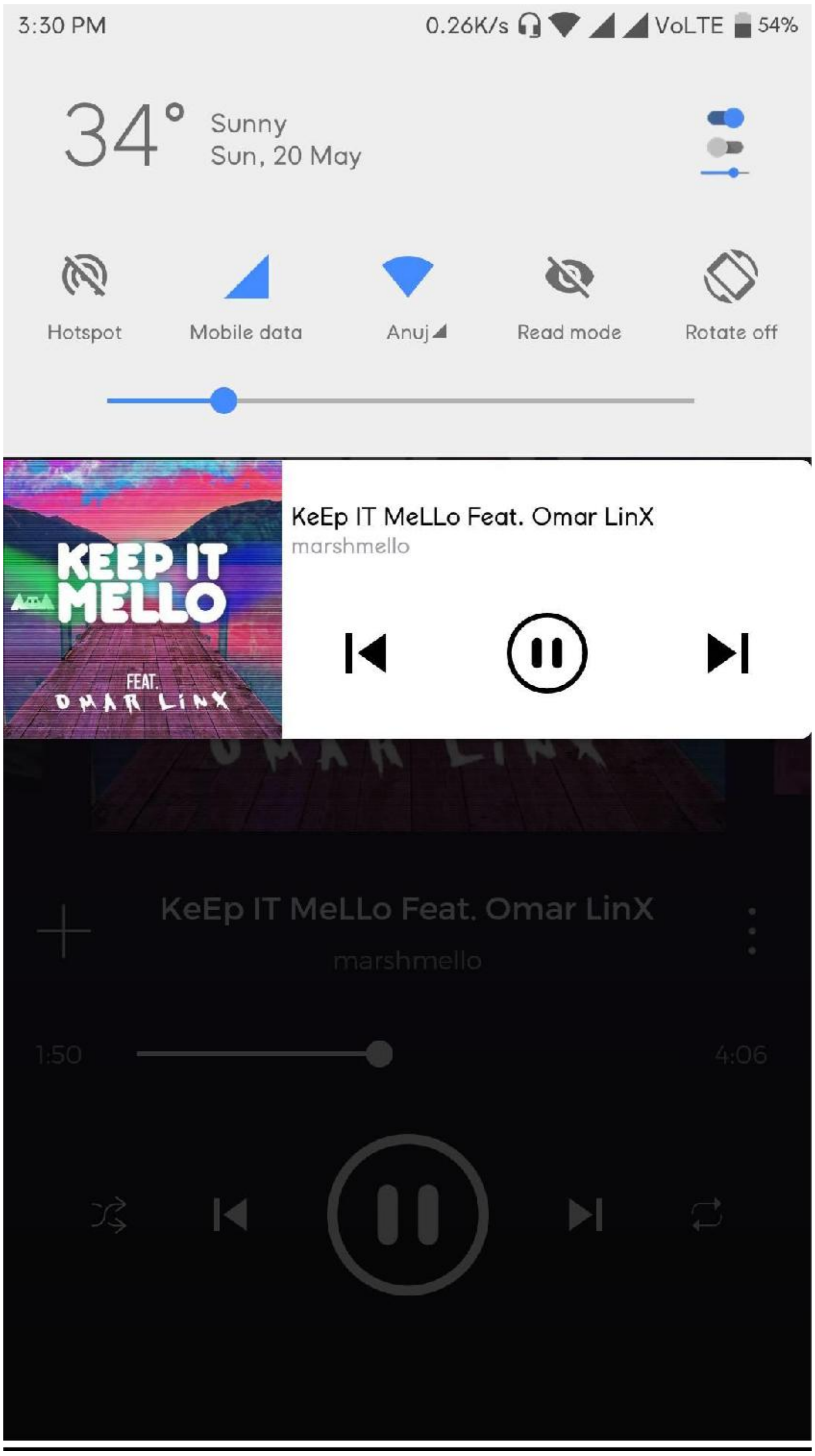
PLAYING FROM
Marshmello

KeEp IT MeLLo Feat. Omar LinX
marshmello

0:05 4:06

211520104059

JANAKIRAMAN S



RESULT:

Thus the mobile application to display music application in web view has been developed ,launched and executed successfully.