

**A
Project Report
On
“Music Streaming App based-on Cloud Computing”**

Submitted in partial fulfilment of the requirements
For qualifying

M.Sc. (Computer Science) [SEMESTER-IV]

Submitted By:

Rohan Rajendra Khedekar

Under The guidance of:

Prof. Mrs. Vinita Singh

AND

Head of Department

Mr. Mandar Bhawe

Department of Computer Science

D. G. Ruparel College of Arts, Science and Commerce,

Mahim, Mumbai 400 016

UNIVERSITY OF MUMBAI

2020–2021

PREFACE

Over the last few decades, Computer already had a considerable impact on many aspects of our society. Medicine, Government, Banking, Education Institutions, Engineering, Railway and Airline Reservations – these are only some of the fields in which computers are already playing a highly significant role.

Over the next few years, there would be a great increase in computer applications and would also have wide use of computers.

Now the people are mostly attracted to mobile and the use of mobile is increasing day by day. From the last decade, the use of mobile devices increased exponentially over the period of time, this leads to creating an opportunity for a mobile software developer.

In the year ahead, the database system will become increasingly widespread and of increasing importance. The number of systems developed is growing day by day an exponential rate.

This project is also part of the latest development in cloud industry to process the data over server without any restriction also known as Serverless Architecture and helps to reduce the burden on the developer for managing everything to maintaining the code only.

This technique helps to reduce the load of maintaining the traditional cloud computing environment and reduce the cost of deployment. It helps to run multiple microservices with lower cost and higher response time.

Rohan Rajendra Khedekar

ACKNOWLEDGMENT

It is obvious that the development of a project needs good support from many people. The contributions of the people, who made this project a success, are largely acknowledged.

I agreed that the project guide is competitive. I thank our project guide Prof. Mrs. Vinita Singh to be competitive & others supporting teachers who have done a lot to keep this project systematically and on schedule. We are thankful for their invaluable guidance at every stage of this project.

Our sincere thanks to Head of Department, Mr. Mandar Bhave, Department of Computer Science, D. G. Ruparel College of Arts, Science and Commerce, Mahim, Mumbai 400 016 for extending her whole-hearted supports for successfully completing our project.

We also express our sincere thanks to the non-teaching staff of the Department of Computer Science, D. G. Ruparel College of Arts, Science and Commerce, Mahim, Mumbai 400 016 for their invaluable support and co-operation.

And, at last, my sincere thanks to my colleagues for their Moral support during the course of our project. We are also thankful to each and every person who is involved with us in this project; their encouragement and support enabled the project to materialize and contributed to its success.

Rohan Rajendra Khedekar

INDEX

SR NO.		TITLE	PAGE NO.
1	-	TITLE	5
2	-	IMPLEMENTATION DETAILS	6-13
-	2.1	INTRODUCTION	6
-	2.2	LITERATURE SURVEY	7-9
-	2.3	OBJECTIVE	10
-	2.4	FLOWCHART	11
-	2.5	METHODOLOGY	12
-	2.6	REQUIREMENT SPECIFICATION	13
3	-	SYSTEM IMPLEMENTATION	14-199
-	3.1	ACTIVITY	14-121
-	3.2	FRAGMENT	122-164
-	3.3	ADAPTER	165-176
-	3.4	MODEL CLASS	177-183
-	3.5	MISC.	184-199
4	-	CONCLUSION& FUTURE ENHANCEMENT	200
5	-	REFERENCES	201

Music Streaming App based-on Cloud Computing

Introduction

Remember those times when the only source of music was CD collections, and if you wanted to listen to something new, you needed to buy a new CD or exchange it with your classmates? But times had changed. The emergence of Smartphone's and the ensuing digitalization had greatly affected the music industry. Now, you could listen to his favourite tracks on his phone at any time and in any place without having to buy CDs.

But you are frustrated with phone was unable to store a huge music collection. This is the right choice for that. This application is all about streaming music of different type, language, albums, artists, genres.

While streaming music's application provides different advance features like you can download songs for playing offline, playing radio, search songs/track/album/artist and genres that u want, different bitrates (96kbps, 160kbps, 320kbps) and the main feature was **Recommendations**.

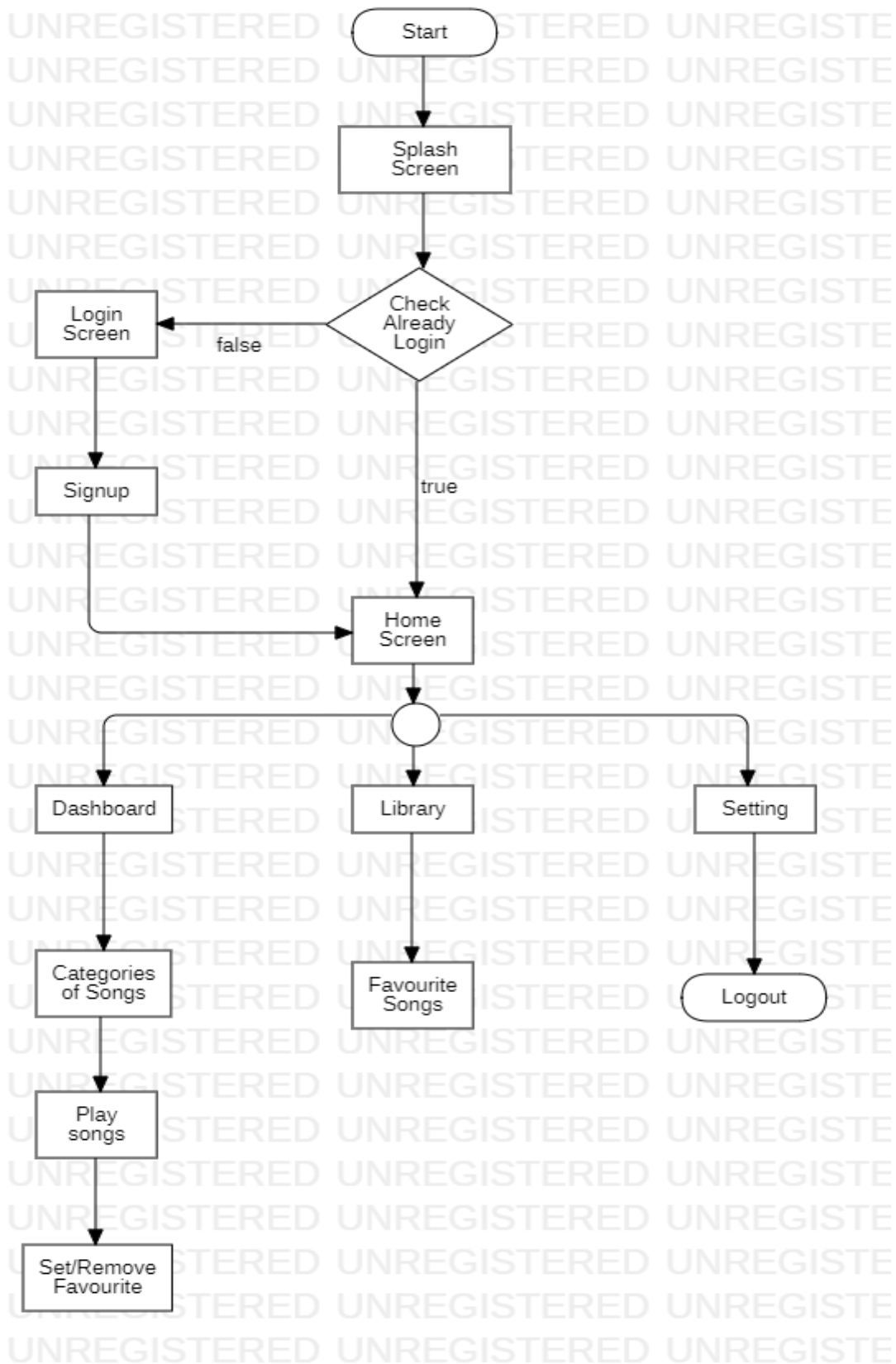
This Application uses **Cloud technology** for Data storing and **MongoDB** as database management tool.

Objective

The main objective of the Project on Music streaming application is to play music seamlessly and get all type of music well categories. In this application music are well categories in Language, Album, Artists, Genres, and Track. Also this application provides features like user can download songs, create your own playlist and like your songs. In this application user can search songs not only search the songs also search playlist, artists and albums for better reach.

Also the main objective of this application was better reach to people. User with all type of internet connection can play the songs with smoothly for that purpose we provide songs of different bits rates with that user can play songs with poor internet connection

Flowchart



Methodology

Agile:

Totally depends on an agile methodology which is iterative, an incremental method of managing the design and build activities that aim to provide new product or service development in a highly flexible and interactive manner.

What is Agile Methodology?

AGILE methodology is a practice that promotes continuous iteration of development and testing throughout the software development life cycle of the project.

Both development and testing activities are concurrent unlike the Waterfall model

The general principles of the Agile Method

- Satisfy the client and continually develop software.
- Changing requirements are embraced for the client's competitive advantage.
- Concentrate on delivering working software frequently. Delivery preference will be placed on the shortest possible time span.
- Developers and business people must work together throughout the entire project.
- Projects must be based on people who are motivated. Give them the proper environment and the support that they need. They should be trusted to get their jobs done.
- Face-to-face communication is the best way to transfer information to and from a team.
- Working software is the primary measure of progress.
- Agile processes will promote development that is sustainable. Sponsors, developers, and users should be able to maintain an indefinite, constant pace.
- Constant attention to technical excellence and good design will enhance agility.
- Simplicity is considered to be the art of maximizing the work that is not done, and it is essential.
- Self-organized teams usually create the best designs.
- At regular intervals, the team will reflect on how to become more effective, and they will tune and adjust their behaviour accordingly.

Agile software development emphasizes on four core values.

1. Individual and team interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

Requirement Specification

Minimum Hardware Requirement:

- **RAM: 4 GB or Higher**
- **Storage (HDD): 500 GB or Higher**
- **Processor: Intel i3 6th Gen or Later**

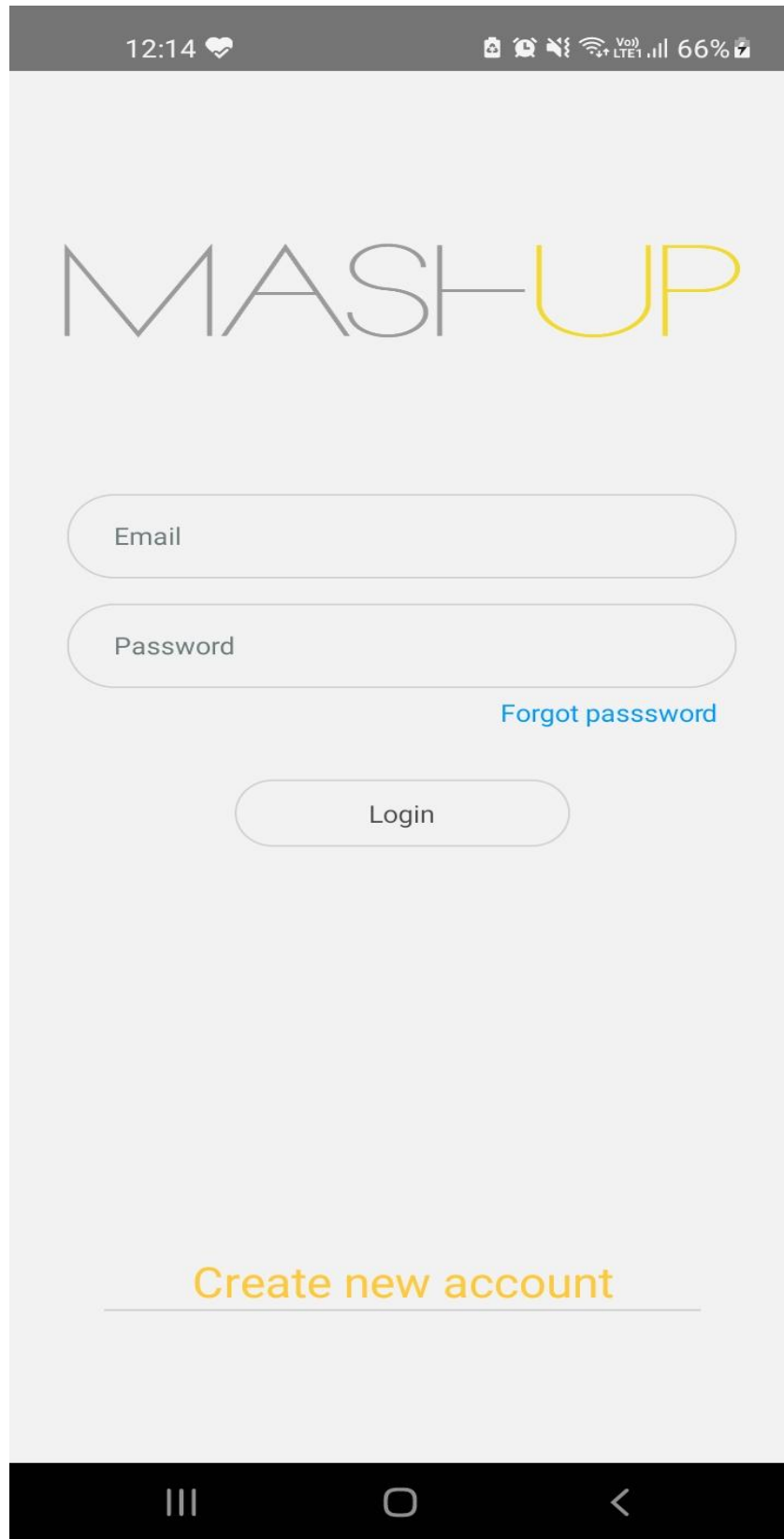
Minimum Software Requirement:

- **Operating System: Windows 10 or any**
- **DATABASE: MongoDB**
- **SOFTWARES:**
 - **ANDROID STUDIO: VERSION 4.1.0.0**
 - **VISUAL STUDIO CODE EDITOR**
 - **LANGUAGE: React-Native, Node.js**

System Implementation

Activity

1. LoginActivity:



The screenshot displays the LoginActivity interface for the MASH-UP application. At the top, a dark grey status bar shows the time 12:14, a heart icon, and various system icons including a battery level of 66%. The main content area has a light grey background. The app's logo, "MASH-UP", is centered at the top in a large, thin, sans-serif font, with "UP" in yellow. Below the logo are two rounded rectangular input fields: the first is labeled "Email" and the second is labeled "Password". To the right of the password field is a blue link that says "Forgot passsword". Below these fields is a rounded rectangular "Login" button. At the bottom of the screen, there is a yellow link that says "Create new account" with a thin horizontal line underneath it. The bottom of the screen features a black navigation bar with three white icons: a hamburger menu, a circle, and a back arrow.

LoginComponent.js

```
import React, { useState } from "react";

import { Image, Keyboard, StyleSheet, Text, TextInput, TouchableOpacity, View } from "react-native";

import { KeyboardAwareScrollView } from "react-native-keyboard-aware-scroll-view";

import { authenticateUser } from "../function/authenticateUser";

const LogoComponent = () => {
  return (
    <View>
      <Image
        style={styles.logoHeaderImage}
        resizeMode="stretch"
        source={require("../assets/LoginIcon.png")}
      />
    </View>
  )
}

const LoginSection = ({ navigation }) => {

  const [textInputs, onChangeText] = useState({ });
  const [errorText, setError] = useState(null);

  const onChangeTextFunction = (name, value) => {
    let newTextInputs = { ...textInputs, [name]: value };
    onChangeText(newTextInputs);
    setError(null);
  }

  const loginPressed = () => {
    const { email = null, password = null } = textInputs;
```

```

let emailVerify = /^\\w+([\\.-]?\\w+)*@\\w+([\\.-]?\\w+)*\\.\\w+\\w+)$/;

if(!email && !password){
    setError("Please enter email and password")
}else if(!email){
    setError("Please enter email")
}else if(!password){
    setError("Please enter password")
}else if (emailVerify.test(email) === false) {
    setError("Please enter valid email")
}else if(password.length < 8){
    setError("Password must be minimum 8 character")
}else{
    authenticateUser(email, password, authenticationResponse)
}
}

const authenticationResponse = (res) => {
    // console.log("authenticationResponse",res);
    //if user was there in database then create otp
    //pass res and otp to the next screen
    //screen will first send otp to node service
    //node service send otp to user via Email
    //after successfully send email validate enter otp
    if(res.data.code === "Email ERROR"){
        setError("Email Id is not registered")
    }else{
        navigation.navigate("Otp",{
            userDetails: res.data
        })
    }
}

```

```
}
```

```
return (
```

```
  <View style={{
```

```
    marginLeft: 20,
```

```
    marginRight: 20
```

```
  }}>
```

```
    <View>
```

```
      <TextInput
```

```
        style={styles.textInputStyle}
```

```
        placeholder="Email"
```

```
        keyboardType="email-address"
```

```
        placeholderTextColor="#6e7c7c"
```

```
        // maxLength={10}
```

```
        textContentType={"emailAddress"}
```

```
        onSubmitEditing={() => {
```

```
          Keyboard.dismiss;
```

```
        }}
```

```
        onChangeText={(text) => {
```

```
          onChangeTextFunction("email", text)
```

```
        }}
```

```
      />
```

```
    </View>
```

```
  <View style={{ marginTop: 15 }}>
```

```
    <TextInput
```

```
      style={styles.textInputStyle}
```

```
      placeholder="Password"
```

```
      placeholderTextColor="#6e7c7c"
```

```
      secureTextEntry={true}
```

```
      onSubmitEditing={() => {
```

```
        Keyboard.dismiss;
```

```

    }}
    onChangeText={(text) => {
        onChangeTextFunction("password", text);
    }}
  />
</View>

{
  errorText ?
    <Text style={{
      fontSize: 14, color: "#ff0000",marginLeft: 20
    }}
    >
      {errorText}
    </Text>
  : null
}

<View style={{ marginTop: 5, marginRight: 10, alignItems: 'flex-end' }}>
  <TouchableOpacity>
    <Text style={{ color: "#0099ff", }}>Forgot passsword</Text>
  </TouchableOpacity>
</View>

<View
  style={{
    marginTop: 30,
    alignItems: 'center'
  }}
>
  <View
    style={{

```

```

        height: 40,
        width: '50%',
        alignItems: 'center',
        justifyContent: 'center',
        borderWidth: 1,
        borderColor: "#d1d1d1",
        borderRadius: 25,
      }}
    >
    <TouchableOpacity
      style={{
        opacity: 0.7
      }}
      // onPress={() => navigation.navigate("Otp")}
      onPress={() => {
        loginPressed()
      }}
    >
    <Text>Login</Text>
  </TouchableOpacity>
</View>
</View>

</View>
)
}

const LoginComponent = ({ navigation }) => {

  console.log("Inside LoginComponent")

  return (

```



```

<KeyboardAwareScrollView
    keyboardShouldPersistTaps={"handled"}
    contentContainerStyle={{ flexGrow: 1 }}
    extraHeight={130}
    bounces={false}
>
  <View style={{ flex: 1, margin: 10 }}>
    <View style={{
      flex: 1,
      justifyContent: 'space-around',
      // marginTop: 30
    }}>
      <LogoComponent />
    </View>
    <View style={{
      flex: 1,

    }}>
      <LoginSection navigation={navigation} />
    </View>

  <View
    style={{
      flex: 1,
      // marginTop: 60,
      justifyContent: 'space-around',
      alignItems: 'center'
    }}
  >
    <TouchableOpacity
      style={{
        borderBottomColor: "#d1d1d1",

```

```

        borderBottomWidth: 1,
        width: '80%',
      }}
      onPress={() => navigation.navigate("SignUp")}
    >
    <Text
      style={{
        color: "#ffc93c",
        fontSize: 25,

        textAlign: 'center',
      }}
    >
      Create new account
    </Text>
  </TouchableOpacity>
</View>
</View>
</KeyboardAwareScrollView>
)
}

```

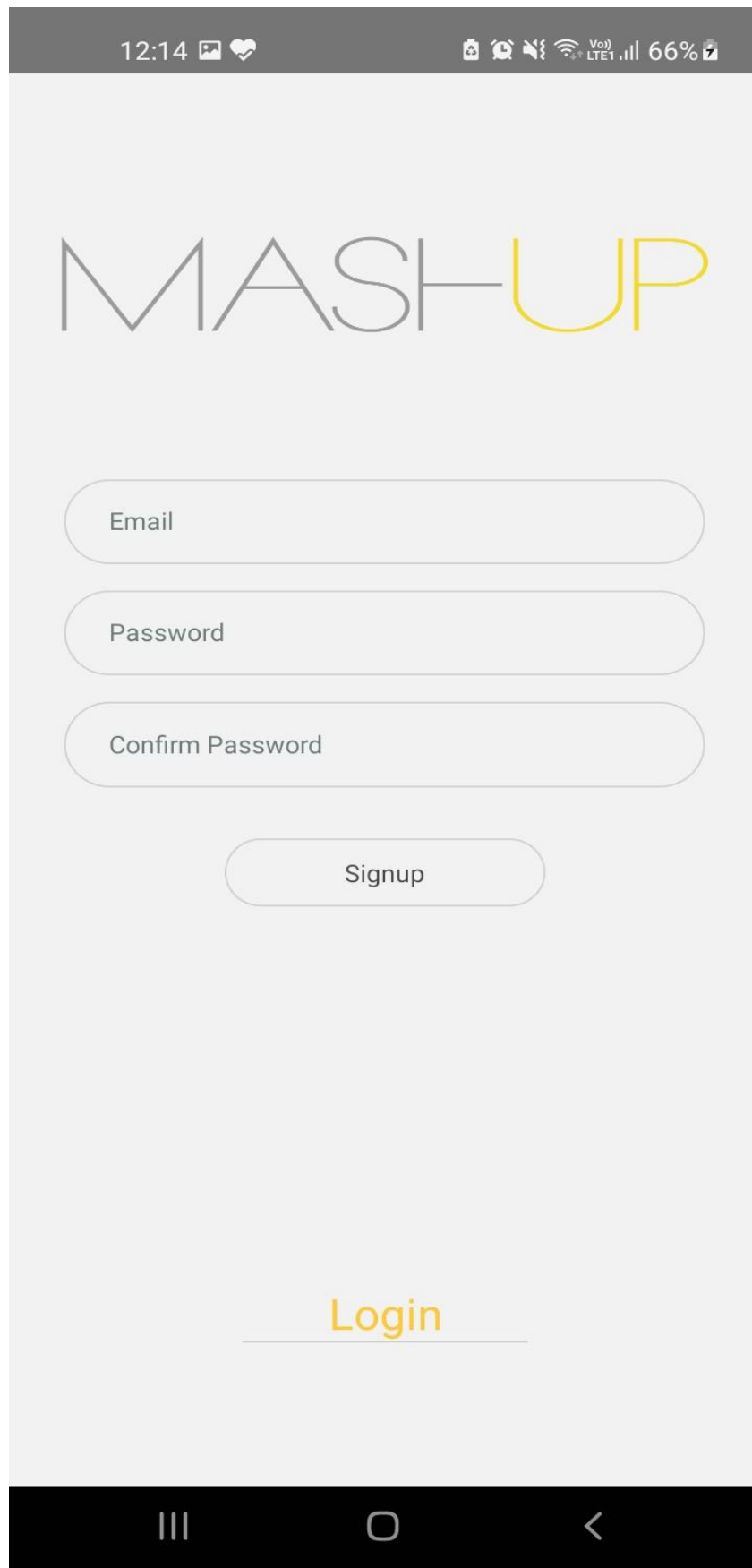
```

const styles = StyleSheet.create({
  logoHeaderImage: {
    alignSelf: "center",
    height: 80,
    width: '100%',
  },
  loginHeader: {
    justifyContent: 'center',
    textAlign: 'center',
    fontSize: 40,

```

```
    },  
    textInputStyle: {  
      borderColor: "#d1d1d1",  
      borderWidth: 1,  
      borderRadius: 25,  
      paddingLeft: 25,  
    }  
  })  
  
export default LoginComponent;
```

2.SignupActivity:



A screenshot of a mobile application's SignupActivity. The screen has a light gray background. At the top is a dark gray status bar with the time 12:14, a heart icon, and various system icons including LTE and 66% battery. The app's logo, "MASH-UP", is centered at the top in a large, thin, sans-serif font, with "UP" in yellow. Below the logo are three rounded rectangular input fields stacked vertically, labeled "Email", "Password", and "Confirm Password". Below these fields is a single rounded rectangular button labeled "Signup". At the bottom of the screen, the word "Login" is written in yellow, with a thin horizontal line underneath it. The very bottom of the screen shows a black Android navigation bar with three white icons: a square, a circle, and a triangle.

12:14

MASH-UP

Email

Password

Confirm Password

Signup

Login

SignupScreen.js

```
import React, { useState } from 'react';

import { Image, Keyboard, StyleSheet, Text, TextInput, TouchableOpacity, View } from 'react-native';

import { KeyboardAwareScrollView } from 'react-native-keyboard-aware-scroll-view';

import { registerUser } from '../function/registerUser';

const LogoComponent = () => {
  return (
    <View>
      <Image
        style={styles.logoHeaderImage}
        resizeMode="stretch"
        source={require("../assets/LoginIcon.png")}
      />
    </View>
  )
}

const SignupSection = ({navigation}) => {
  // console.log("navigation",navigation)

  const [textInputs, onChangeText] = useState({});
  const [errorText, setError] = useState(null);

  const onChangeTextFunction = (name, value) => {
    let newTextInputs = { ...textInputs, [name]: value };
    onChangeText(newTextInputs);
    setError(null);
  }

  const singupPressed = () => {
```

```

const { email = null, password = null, confirmPassword = null } = textInputs;

let emailVerify = /^\\w+([\\.-]?\\w+)*@\\w+([\\.-]?\\w+)*\\.\\w+\\w+)$/;

if (!email && !password && !confirmPassword) {
  setError("Please enter email and password")
} else if (!email) {
  setError("Please enter email")
} else if (!password) {
  setError("Please enter password")
} else if (!confirmPassword) {
  setError("Please enter the password again")
} else if (emailVerify.test(email) === false) {
  setError("Please enter valid email")
} else if (password.length < 8) {
  setError("Password must be minimum 8 character")
} else if (confirmPassword !== password) {
  setError("The password you entered do not match.")
} else {
  registerUser(email, confirmPassword, registerUserCallback)
}
}

```

```

const registerUserCallback = (res) => {
  console.log("registerUserCallback", res)
  if(res.data.code === "Email ERROR"){
    setError("A user already exists with this email address.")
  }else{
    navigation.navigate("Otp",{
      userDetails: {
        userId: res.userId,
        email: res.email,

```

```

        password: res.password
    }
  })
}
}

return (
  <View style={{
    marginLeft: 20,
    marginRight: 20
  }}>
    <View>
      <TextInput
        style={styles.textInputStyle}
        placeholder="Email"
        keyboardType="email-address"
        placeholderTextColor="#6e7c7c"
        // maxLength={10}
        textContentType="emailAddress"
        onSubmitEditing={() => {
          Keyboard.dismiss();
        }}
        onChangeText={(text) => {
          onChangeTextFunction("email", text)
        }}
      />
    </View>

    <View style={{ marginTop: 15 }}>
      <TextInput
        style={styles.textInputStyle}
        placeholder="Password"

```

```

placeholderTextColor="#6e7c7c"
secureTextEntry={true}
onSubmitEditing={() => {
  Keyboard.dismiss;
}}
onChangeText={(text) => {
  onChangeTextFunction("password", text);
}}
/>
</View>

<View style={{ marginTop: 15 }}>
  <TextInput
    style={styles.textInputStyle}
    placeholder="Confirm Password"
    placeholderTextColor="#6e7c7c"
    secureTextEntry={true}
    onSubmitEditing={() => {
      Keyboard.dismiss;
    }}
    onChangeText={(text) => {
      onChangeTextFunction("confirmPassword", text);
    }}
  />
</View>

{
  errorText ?
    <Text style={{
      fontSize: 14, color: "#ff0000", marginLeft: 20
    }}
  />

```



```

        {errorText}
      </Text>
      : null
    }

    <View
      style={{
        marginTop: 30,
        alignItems: 'center'
      }}
    >
      <View
        style={{
          height: 40,
          width: '50%',
          alignItems: 'center',
          justifyContent: 'center',
          borderWidth: 1,
          borderColor: "#d1d1d1",
          borderRadius: 25,
        }}
      >
        <TouchableOpacity
          style={{
            opacity: 0.7
          }}
          onPress={() => {
            singupPressed()
          }}
        >
          <Text>Signup</Text>
        </TouchableOpacity>

```

```

        </View>
    </View>

    </View>
)
}

const SignupScreen = ({ navigation }) => {

    console.log("Inside SingupScreen")

    return (
        <KeyboardAwareScrollView
            keyboardShouldPersistTaps={ "handled" }
            contentContainerStyle={{ flexGrow: 1 }}
            extraHeight={ 130 }
            bounces={ false }
        >
            <View style={{ flex: 1, margin: 10 }}>
                <View style={{
                    flex: 1,
                    justifyContent: 'space-around',

                }}>
                    <LogoComponent />
                </View>

                <View style={{ flex: 1 }}>
                    <SignupSection navigation={ navigation }/>
                </View>

                <View

```

```

        style={{
            flex: 1,
            // marginTop: 60,
            justifyContent: 'space-around',
            alignItems: 'center'
        }}
    >
    <View
        style={{
            borderBottomColor: "#d1d1d1",
            borderBottomWidth: 1,
            width: '40%',
        }}
    >
    <TouchableOpacity
        onPress={() => navigation.navigate("Login")}
    >
    <Text
        style={{
            color: "#ffc93c",
            fontSize: 25,

            textAlign: 'center',
        }}
    >
        Login
    </Text>
    </TouchableOpacity>
    </View>
    </View>

</View>

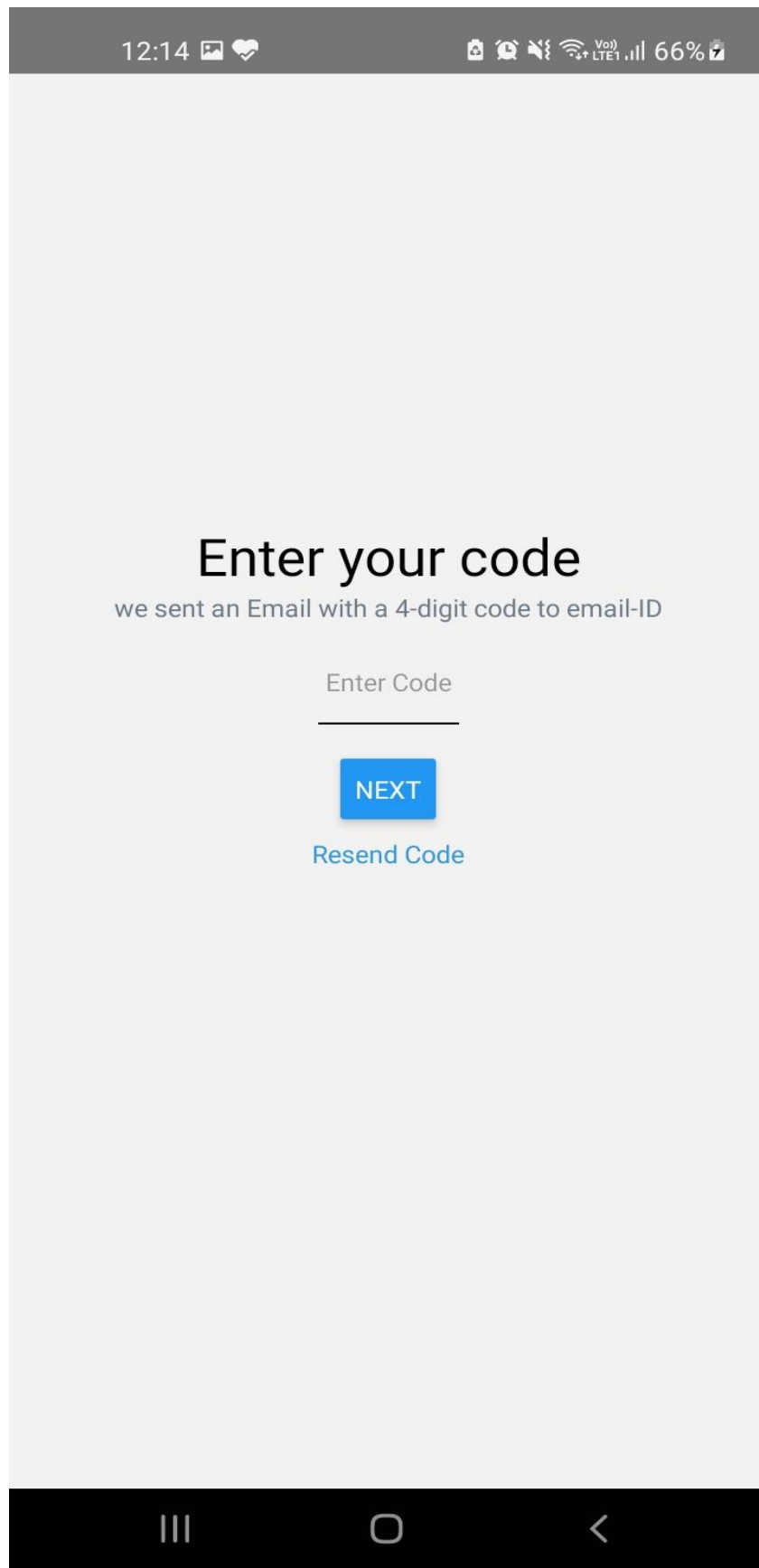
```

```
        </KeyboardAwareScrollView>
    )
}
```

```
const styles = StyleSheet.create({
  logoHeaderImage: {
    alignSelf: "center",
    height: 80,
    width: '100%',
  },
  loginHeader: {
    justifyContent: 'center',
    textAlign: 'center',
    fontSize: 40,
  },
  textInputStyle: {
    borderColor: "#d1d1d1",
    borderWidth: 1,
    borderRadius: 25,
    paddingLeft: 25,
  }
})
```

```
export default SignupScreen;
```

3)Enter OTP:



The image shows a mobile application interface for entering an OTP. At the top, a dark grey status bar displays the time 12:14, a gallery icon, a heart icon, and various system icons including alarm, silent mode, VoLTE, LTE, signal strength, and 66% battery. The main content area has a light grey background. Centered on the screen is the text "Enter your code" in a large, bold, black font. Below it, in a smaller grey font, is the text "we sent an Email with a 4-digit code to email-ID". Further down is a text input field with the placeholder "Enter Code" and a horizontal line underneath. Below the input field is a blue rectangular button with the word "NEXT" in white capital letters. At the bottom of the input section is a blue link that says "Resend Code". The bottom of the screen features a black navigation bar with three white icons: a hamburger menu, a circle, and a back arrow.

12:14

Enter your code

we sent an Email with a 4-digit code to email-ID

Enter Code

NEXT

Resend Code

otpScreen.js

```
import React, { useEffect, useState } from 'react';
import { TextInput } from 'react-native';
import { Button } from 'react-native';
import { Text } from 'react-native';
import { View } from 'react-native';
import { KeyboardAwareScrollView } from 'react-native-keyboard-aware-scroll-view';
import { useDispatch } from 'react-redux';
import { setStorage } from '../utility/localutility';
import { sentOtpEmail } from '../function/sentOtpEmail';

const otpScreen = (props) => {
  // console.log("Inside otpScreen", props);

  const redux = useDispatch();

  const [otp, setOtp] = useState();

  const [state, setState] = useState({
    userId: props.route.params.userDetails.userId,
    userEmail: props.route.params.userDetails.email,
    userPassword: props.route.params.userDetails.password,
    otp: "",
    enterOtp: "",
    err: false
  });

  // const [otp, setOtp] = useState("")

  useEffect(() => {
    let otp = (Math.floor(Math.random() * 10000) + 10000).toString().substring(1);
```

```

    setState((prevState) => {
      let newState = {
        ...prevState,
        otp
      }
      return newState;
    })
    sentOtpEmail(state.userEmail, otp, sentOtpEmailCallback)
  }, [])

```

```

const sentOtpEmailCallback = (res) => {
  console.log("sentOtpEmailCallback");
}

```

```

const verifyOtp = async () => {
  if (state.otp !== state.enterOtp) {
    setState((prevState) => {
      let newState = {
        ...prevState,
        err: true
      }
      return newState;
    })
  } else {
    let userCredentials = {
      userId: state.userId,
      email: state.userEmail,
      password: state.userPassword
    }
    console.log("userCredentials", userCredentials)
    await setStorage('userCredentials', userCredentials);
    props.navigation.navigate("MainAppStack")
  }
}

```

```

    }
  }

  return (
    <KeyboardAwareScrollView
      keyboardShouldPersistTaps={"handled"}
      contentContainerStyle={{ flexGrow: 1 }}
      extraHeight={130}
      bounces={false}
    >
      <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
        <View style={{ alignItems: 'center', marginBottom: 100 }}>
          <Text style={{ fontSize: 30 }}>Enter your code</Text>
          <Text style={{ color: '#6c7a89' }}>we sent an Email with a 4-digit code to email-
ID</Text>
          <TextInput
            style={{ borderBottomWidth: 1, marginBottom: 20, marginTop: 10 }}
            placeholder="Enter Code"
            keyboardType="numeric"
            maxLength={4}
            value={state.enterOtp}
            onChangeText={(text) => {
              if(!/^0-9]/g.test(text.toString())){
                setState((prevState)=>{
                  let newState = {
                    ...prevState,
                    enterOtp: text
                  }
                  return newState
                })
              }
            }}
          />

```



```

    {
      state.err ?
        <Text style={{ color: 'red', marginBottom: 20 }}>
          Please enter a valid 4 digit code to login
        </Text>
      : null
    }

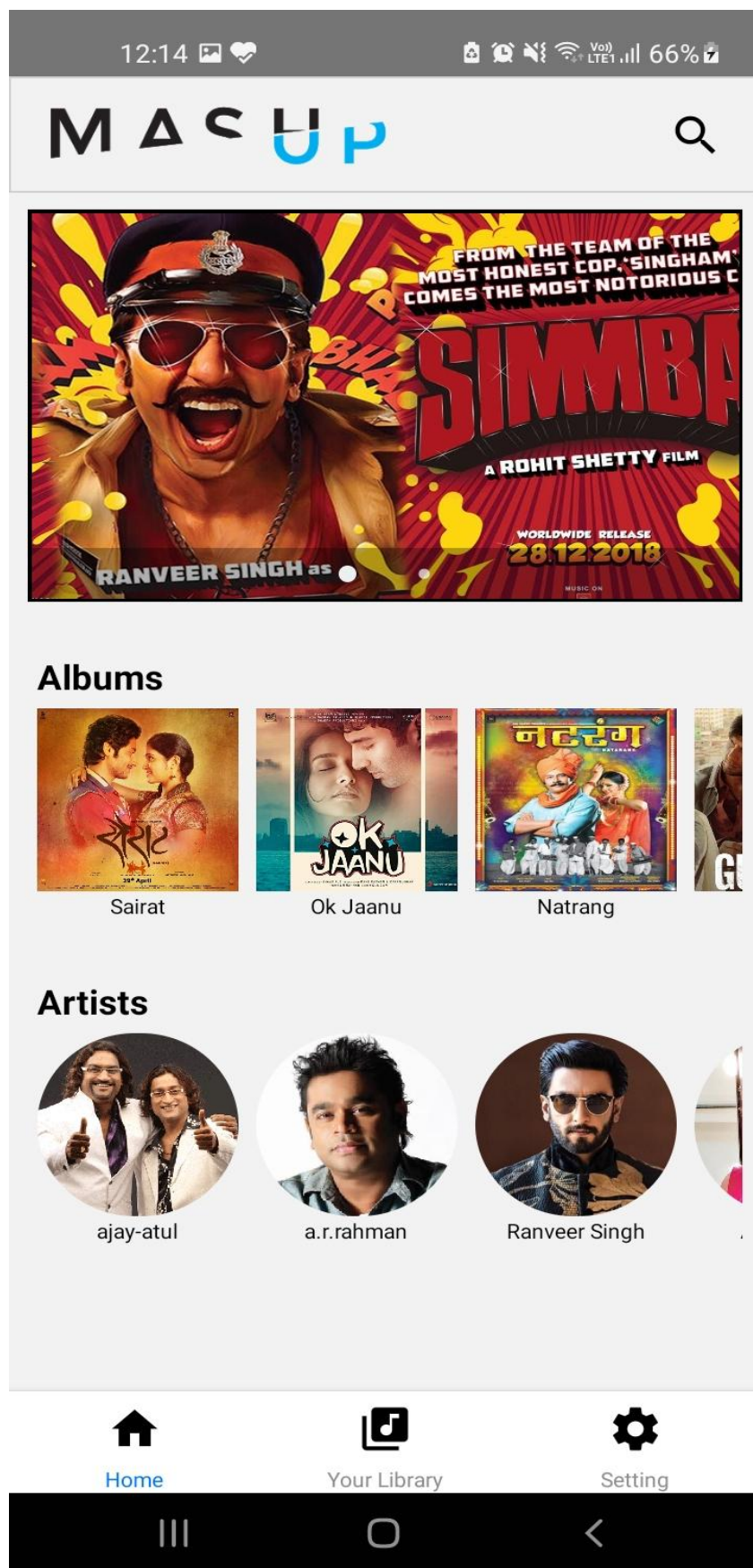
    <Button
      style={{ }}
      title="Next"
      onPress={() => verifyOtp()}
    />

    <Text style={{ marginTop: 10, color: '#3498db' }}>Resend Code</Text>
  </View>
</View>
</KeyboardAwareScrollView>
)
}

export default otpScreen;

```

4)Home Screen:



HomeScreen.js

```
import React from 'react';
import { Image, Modal, Text, TouchableOpacity, View } from 'react-native';
import Icon from 'react-native-vector-icons/MaterialIcons';
import AlbumList from '../AlbumList/AlbumList';
import ArtistList from '../ArtistList/ArtistList';
import Banner from '../Banner/Banner';
import { Overlay } from 'react-native-elements';

const Header = () => {
  return (
    <View style={{ flex: 1, borderColor: "#d1d1d1", borderWidth: 1, flexDirection: 'row', justifyCo
ntent: 'space-evenly' }}>
      <View style={{ flex: 5 }}>
        <Image
          resizeMode="contain"
          style={{
            flex: 1,
            width: '60%',
            marginLeft: 10
          }}
          source={require('../../assets/HomePageHeading.png')}
        />
      </View>
      <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center' }}>
        <TouchableOpacity>
          <Icon name="search" size={30} />
        </TouchableOpacity>
      </View>
    </View>
  )
}
```

```

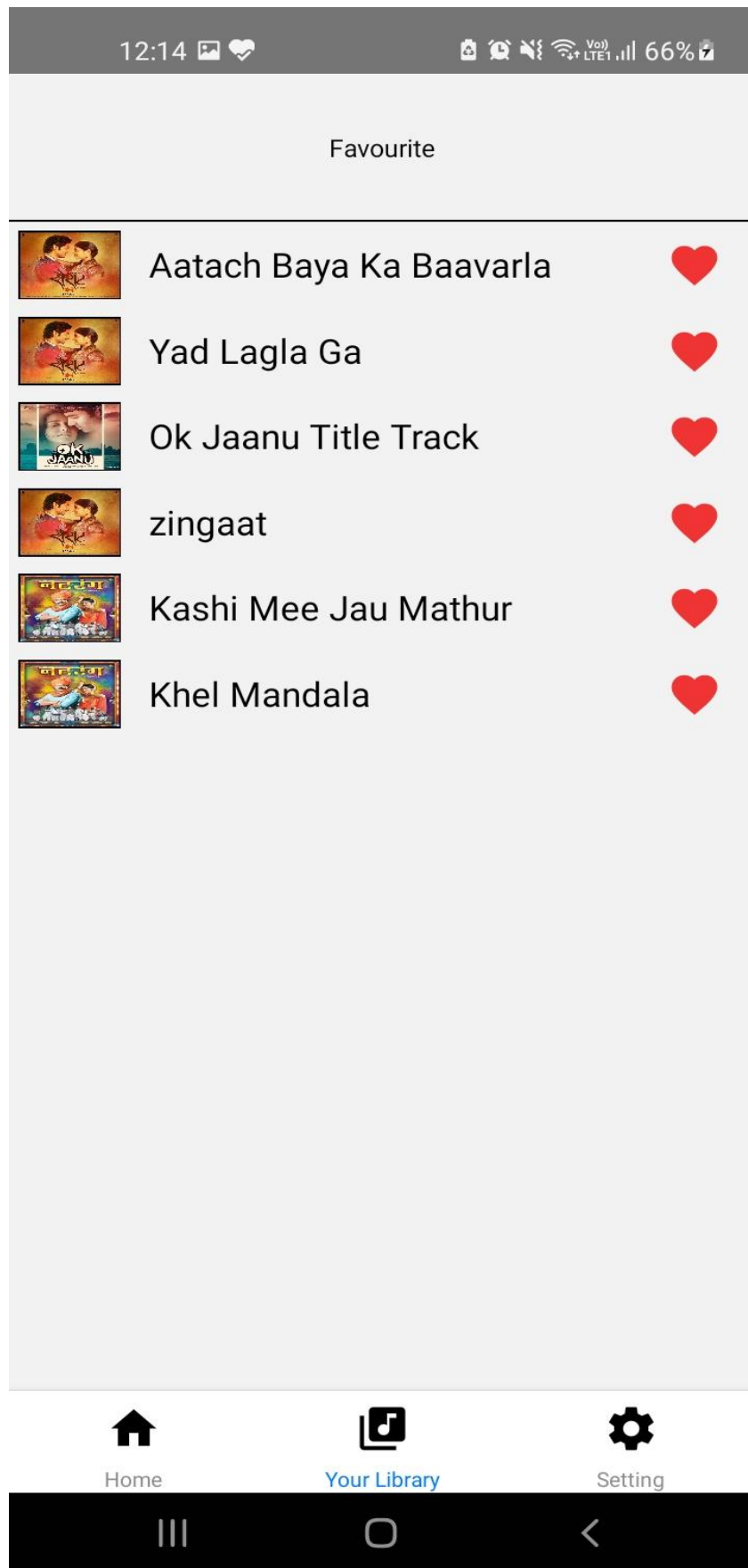
const HomeScreen = (props) => {
  // console.log("HomeScreen",props);

  return (
    <View style={{ flex: 1 }}>
      <View style={{ flex: 1 }}>
        <Header />
      </View>
      <View style={{ flex: 10, margin: 10 }}>
        <View style={{ flex: 1, borderWidth: 2 }}>
          <Banner {...props}/>
        </View>
        <View style={{ flex: 2 }}>
          <View style={{ marginTop: 30,height: 160,/* borderWidth: 1 */}}>
            <AlbumList {...props}/>
          </View>
          <View style={{ marginTop: 30,height: 160,/* borderWidth: 1 */}}>
            <ArtistList {...props}/>
          </View>
        </View>
      </View>
      /* <Overlay fullScreen={false} style={{ position: 'absolute',backgroundColor: 'red'}}>
        <View>
          <Text>Rohan</Text>
        </View>
      </Overlay> */
    </View>
  )
}

export default HomeScreen;

```

5)Favourite Songs:



Favourite.js

```
import React, { useEffect, useState } from 'react'
import { Text } from 'react-native'
import { TouchableOpacity } from 'react-native'
import { FlatList } from 'react-native'
import { Image } from 'react-native'
import { View } from 'react-native'
import Icon from 'react-native-vector-icons/MaterialIcons';
import { DATA_URL } from '../envs/development'
import { getFavourite } from '../function/getFavourite'
import { setFavourite } from '../function/setFavourite'
```

```
const Favourite = () => {
```

```
  const [state, setState] = useState({
    favList: [],
    refreshCount: 0
  })
```

```
  useEffect(() => {
    getFavourite(getFavouriteCallback)
  }, [state.refreshCount])
```

```
  const getFavouriteCallback = (res) => {
    // console.log("getFavouriteCallback",res);
    if (res.data.result === 'success') {
      setState((prevState) => {
        let newState = {
          ...prevState,
          favList: res.data.data
        }
      })
    }
  }
```

```

        return newState;
    })
}
}

const setRemoveFavourite = async (trackId, index) => {
    // setFavourite(trackId, setFavouriteHandler)
    setFavourite(trackId, setFavouriteHandler)
}

const setFavouriteHandler = (res) => {
    // console.log("setFavouriteHandler",res)
    setState((prevState) => {
        let newState = {
            ...prevState,
            refreshCount: prevState.refreshCount + 1,
        };
        return newState;
    });
};

// const renderItem = ({ item, index }) => {
//     return (
//         <View style={{ flex: 1, flexDirection: 'row', height: 50, width: '100%' }}>
//             { /* <View style={{ flex: 1, backgroundColor: 'lightpink' }}>
//                 <View style={{ flex: 1, borderWidth: 1, margin: 5 }}>

//                     </View>
//                 </View> */ }
//             <View style={{ flex: 4, justifyContent: 'center', alignItems: 'flex-
start', marginLeft: 10 }}>
//                 <TouchableOpacity onPress={() => {
//                     // reduxSave({type: SELECT_SONGS, payload: {

```

```

//          //   track: item,
//          //   playList: item
//          //   })
//      }>
//      <Text style={{ fontSize: 20 }}>{item.name}</Text>
//      </TouchableOpacity>
//      </View>
//      <View style={{ flex: 1,/* backgroundColor: 'yellow', */justifyContent: 'center', alignItems
: 'center' }}>
//          <Icon name="favorite-border" size={30} onPress={() => {

//          }} />
//      </View>
//      </View>
//  )
// }

const renderItem = ({ item, index }) => {
  // console.log("renderItem", item);
  return (
    <View style={{ flex: 1, flexDirection: 'row', height: 50, width: '100%' }}>
      <View style={{ flex: 1 }}>
        <View style={{ flex: 1, borderWidth: 1, margin: 5 }}>
          <Image
            source={{ uri: `${DATA_URL}poster/${item.albumImgUrl}` }}
            // source={require('../assets/sairat.jpg')}
            style={{ flex: 1, resizeMode: 'stretch', width: '100%' }}
          />
        </View>
      </View>
      <View style={{ flex: 4, justifyContent: 'center', alignItems: 'flex-start', marginLeft: 10 }}>
        <TouchableOpacity onPress={() => {
          reduxSave({
            type: SELECT_SONGS, payload: {

```

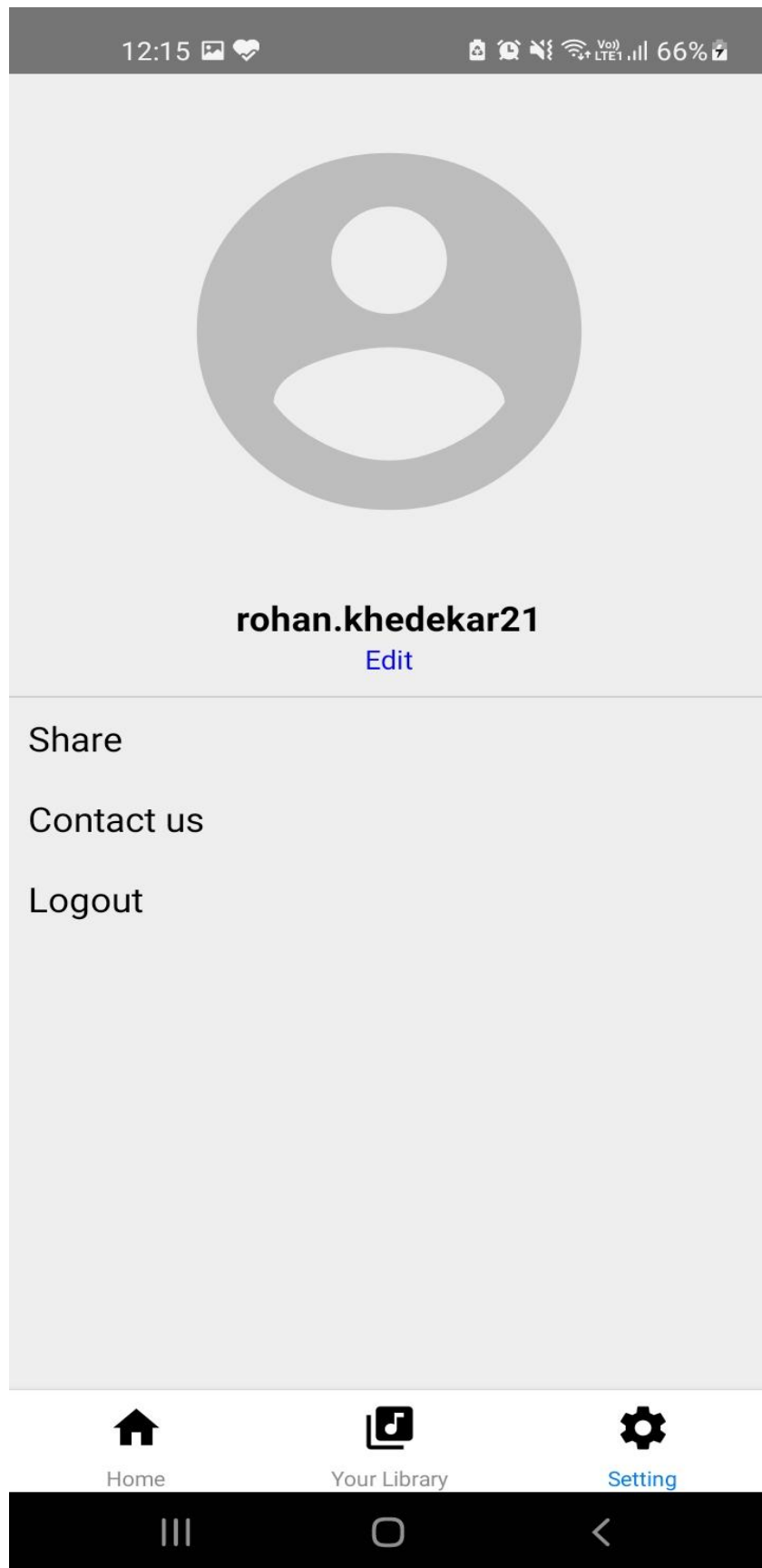


```

        track: item,
        playList: item
      }
    })
  }}>
  <Text style={{ fontSize: 20 }}>{item.trackTitle}</Text>
</TouchableOpacity>
</View>
<View style={{ flex: 1,/* backgroundColor: 'yellow', */justifyContent: 'center', alignItems: '
center' }}>
  <Icon name={item.isFav ? "favorite" : "favorite-
border"} size={30} color={item.isFav ? "#f03434" : 'black'}
    onPress={() => {
      setRemoveFavourite(item.trackId, index)
      // setFavourite(item.trackId, setFavouriteHandler)
    }} />
</View>
</View>
)
}
return (
  <View>
    <FlatList
      data={state.favList}
      keyExtractor={(item, index) => index.toString()}
      renderItem={renderItem}
      showsHorizontalScrollIndicator={false}
    />
  </View>
)
}
export default Favourite;

```

6)Setting Menu:



Setting.js

```
import React, { useEffect, useState } from 'react';
import { TouchableOpacity } from 'react-native';
import { Button } from 'react-native';
import { Text, View } from 'react-native';
import Icon from 'react-native-vector-icons/MaterialIcons';
import { clearStorage, getStorage } from '../utility/localutility';

const SettingMain = (props) => {
  console.log("inside settingMain", props)

  const [userCredentials, setUserCredentials] = useState("")

  useEffect(() => {
    getAsyncStorage()
  }, [])

  const getAsyncStorage = async () => {
    const userCredentials = await getStorage('userCredentials')
    setUserCredentials(userCredentials)
  }

  return (
    <View style={{ flex: 1, backgroundColor: '#EEEEEE' }}>
      <View style={{ flex: 1.8, borderBottomWidth: 1, borderBottomColor: '#d1d1d1' }}>
        <View style={{ flex: 5, justifyContent: 'center', alignItems: 'center'/* ,paddingTop: 20 */ }}
        >
          <Icon name="account-circle" size={250} color="#BDBDBD" />
        </View>
        <View style={{ flex: 1, justifyContent: 'center', alignItems: 'center', marginBottom: 5 }}>
          <Text style={{ fontSize: 20, fontWeight: 'bold' }}>
            {userCredentials ?
```



```


        userCredentials.email.substr(0, userCredentials.email.indexOf('@'))
        : "" }
    </Text>
    <TouchableOpacity>
        <Text style={{ fontSize: 15, color: 'blue' }}>Edit</Text>
    </TouchableOpacity>
</View>
</View>
<View style={{ flex: 2 }}>
    <View style={{ flex: 1, margin: 10 }}>
        <TouchableOpacity style={{ marginBottom: 20 }}>
            <Text style={{ fontSize: 20 }}>Share</Text>
        </TouchableOpacity>
        <TouchableOpacity style={{ marginBottom: 20 }}>
            <Text style={{ fontSize: 20 }}>Contact us</Text>
        </TouchableOpacity>
        <TouchableOpacity style={{ marginBottom: 20 }}
            onPress={() => {
                clearStorage();
                props.navigation.navigate("Login")
            }}
        >
            <Text style={{ fontSize: 20 }}>Logout</Text>
        </TouchableOpacity>
    </View>
</View>
)
}

export default SettingMain;

```


7)SongList:

12:15  





Sairat

PLAY





zingaat







Aatach Baya Ka Baavarla







Sairat Zaala Ji










Yad Lagla Ga



Home

Your Library

Setting

songList.js

```
import React, { useEffect, useState } from 'react';
import { Image, Text, View } from 'react-native';
import { FlatList, TouchableOpacity } from 'react-native-gesture-handler';
import Icon from 'react-native-vector-icons/MaterialIcons';
import { useDispatch } from 'react-redux';
import { SELECT_SONGS } from '../actions/actionTypes';
import { setFavourite } from '../component/Favourite/function/setFavourite';
// import nextSong from '../actions/playerAction';
import { DATA_URL } from '../envs/development';
import { getSongsList } from '../functions/getSongsList';

const SongDetails = (props) => {
  // console.log("Inside SongDetails", props.route.params);

  const reduxSave = useDispatch();

  const [state, setState] = useState({
    SongDetails: props.route.params.SongDetails,
    tracks: [],
    refresh: 0
  })

  useEffect(() => {
    getSongsList(state.SongDetails.type, state.SongDetails.Id, getSongsListHandler)
  }, [props.route.params, state.refresh])

  const getSongsListHandler = (res) => {
    // console.log("getSongsListHandler res",res)
    setState((prevState) => {
      let newState = {
```

```

        ...prevState,
        tracks: res.data
      }
      return newState;
    })
  }

// console.log("state", state.tracks[0])

const renderItem = ({ item, index }) => {
  console.log("renderItem", item);
  return (
    <View style={{ flex: 1, flexDirection: 'row', height: 50, width: '100%' }}>
      <View style={{ flex: 1 }}>
        <View style={{ flex: 1, borderWidth: 1, margin: 5 }}>
          <Image
            source={{ uri: `${DATA_URL}poster/${item.albumImgUrl}` }}
            // source={require('../assets/sairat.jpg')}
            style={{ flex: 1, resizeMode: 'stretch', width: '100%' }}
          />
        </View>
      </View>
      <View style={{ flex: 4, /* backgroundColor: 'lightgreen', */ justifyContent: 'center', alignItems: 'flex-start', marginLeft: 10 }}>
        <TouchableOpacity onPress={() => {
          reduxSave({
            type: SELECT_SONGS, payload: {
              track: item,
              playList: [item]
            }
          })
        }}>
          <Text style={{ fontSize: 20 }}>{item.trackTitle}</Text>

```

```

        </TouchableOpacity>

    </View>

    <View style={{ flex: 1,/* backgroundColor: 'yellow', */justifyContent: 'center', alignItems: '
center' }}>

        <Icon name={item.isFav ? "favorite" : "favorite-
border"} size={30} color={item.isFav ? "#f03434" : 'black'}

            onPress={() => {

                setRemoveFavourite(item.trackId, index)

                // setFavourite(item.trackId, setFavouriteHandler)

            }} />

    </View>

</View>

)

}

```

```

const setRemoveFavourite = async (trackId, index) => {

    // setFavourite(trackId, setFavouriteHandler)

```

```

const setFavouriteHandler = (res) => {

    console.log("setFavouriteHandler", res);

```

```

let tempdata = state.tracks

```

```

switch (res.data.code) {

    case "setFav":

        tempdata[index].isFav = true

        break;

    case "removeFav":

        tempdata[index].isFav = false

        break;

    default:

        break;

}

```



```

    setState((prevState) => {
      let newState = {
        ...prevState,
        tracks: tempdata
      }
      return newState;
    })
  }

  setFavourite(trackId, setFavouriteHandler)
}

// const setFavouriteHandler = (res) => {
//   console.log("setFavouriteHandler", res);
// }

return (
  <View style={{ flex: 1 }}>
    <View style={{ flex: 1, /* backgroundColor: 'lightblue', */ paddingBottom: 20 }}>
      <View style={{ flex: 1, width: '100%', justifyContent: 'center', alignItems: 'center', padding
Top: 20 }}>
        <View style={{ flex: 1, borderWidth: 1, width: '50%/* ,margin: 30 */ }}>
          <Image
            source={{ uri: `${DATA_URL}${state.SongDetails.type === 'artist' ? "artist" : "pos
ter"}/${state.SongDetails.imgURL}` }}
            // source={require('../../assets/sairat.jpg')}
            style={{ flex: 1, resizeMode: 'stretch', width: '100%' }}
          />
        </View>
        <Text style={{ fontSize: 20, marginTop: 8 }}>{state.SongDetails.Name}</Text>
        <TouchableOpacity style={{ marginTop: 5, borderWidth: 1, width: 90, height: 30, border
Radius: 50, justifyContent: 'center', alignItems: 'center' }}

```

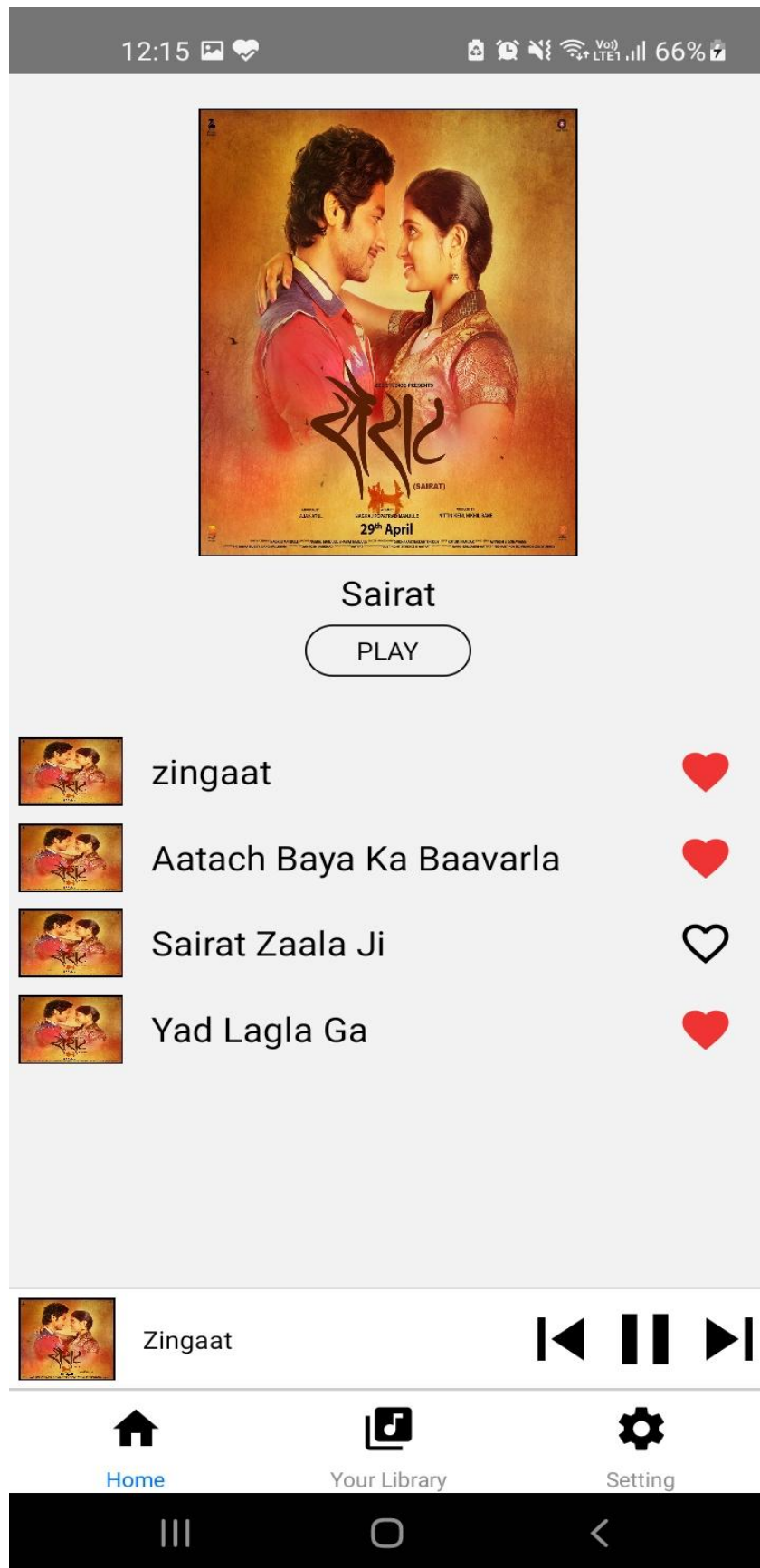
```

        onPress={() => {
          reduxSave({
            type: SELECT_SONGS, payload: {
              track: state.tracks[0],
              playList: state.tracks
            }
          })
        }}
      >
      <Text>PLAY</Text>
    </TouchableOpacity>
  </View>
</View>
<View style={{ flex: 1.1, /* backgroundColor: 'lightgreen', */ paddingTop: 10 }}>
  <FlatList
    data={state.tracks}
    keyExtractor={(item, index) => index.toString()}
    renderItem={renderItem}
    showsHorizontalScrollIndicator={false}
  />
</View>
</View>
)
}

export default SongDetails;

```

8)Song Player:



PlayerWidget.js

```
import React, { useEffect, useRef } from 'react';
import { Image } from 'react-native';
import { TouchableOpacity } from 'react-native';
import { Text, View } from 'react-native';
import Icon from 'react-native-vector-icons/MaterialIcons';
import Video from 'react-native-video';
import { useDispatch, useSelector } from 'react-redux';
import { NEXT_SONGS, PLAY_PAUSE_SONGS, PREV_SONGS } from '../actions/actionTypes';
import { DATA_URL } from '../envs/development';
// import { playpauseSongs } from '../actions/actionTypes';

const PlayerWidget = () => {
  // const player = useRef(null)

  const redux = useSelector(state => state.player)

  console.log("Inside Player redux", redux);

  const reduxSave = useDispatch();

  // console.log("reduxDispatch",reduxSave)

  const nextSong = () => {
    const currentIdx = redux.playlist.findIndex(x => redux.track.trackId === x.trackId)
    console.log("currentIdx",currentIdx)
    console.log("song",redux.playlist[0])
    const nextSong = redux.playlist[currentIdx + 1];
    console.log("nextSong",nextSong);
    if (nextSong && nextSong !== null) {
      reduxSave({ type: NEXT_SONGS, payload: nextSong })
    }
  }

  const prevSong = () => {
```

```

const currentIdx = redux.playlist.findIndex(x => redux.track.trackId === x.trackId)

const prevSong = redux.playlist[currentIdx - 1];

if(prevSong && prevSong !== null){
  reduxSave({type: PREV_SONGS, payload: prevSong})
}
}

return (
  <View style={{ flex: 1, flexDirection: 'row', position: 'absolute', width: '100%', height: 60, bottom: 60, borderBottomWidth: 1, borderBottomColor: '#d1d1d1', borderTopWidth: 1, borderTopColor: '#d1d1d1', backgroundColor: 'white' }}>
    <TouchableOpacity style={{ flex: 4, flexDirection: 'row' }}>
      <View style={{ flex: 1, borderWidth: 1, margin: 5 }}>
        <Image
          source={{ uri: `${DATA_URL}poster/${redux.track.albumImgUrl}` }}
          // source={require('../assets/sairat.jpg')}
          style={{ flex: 1, resizeMode: 'stretch', width: '100%' }}
        />
      </View>
      <View style={{ flex: 4, justifyContent: 'center', marginLeft: 10 }}>
        <Text>{redux.track !== null ? redux.track.trackUrl.substring(0, redux.track.trackUrl.indexOf(".")) : "SONG NAME"}</Text>
      </View>
    </TouchableOpacity>
    <View style={{ flex: 2, flexDirection: 'row', justifyContent: 'center', alignItems: 'center' }}>
      <View style={{ flex: 1 }}>
        <Icon name="skip-previous" size={50} onPress={() => prevSong()} />
      </View>
      <View style={{ flex: 1 }}>
        {
          redux.statusPlaying
            ?
            <Icon name="play-arrow" size={50} onPress={() =>
              reduxSave({ type: PLAY_PAUSE_SONGS, payload: false })
            } />
        }
      </View>
    </View>
  </View>
)

```

```

        } />
      :
      <Icon name="pause" size={50} onPress={() =>
        reduxSave({ type: PLAY_PAUSE_SONGS, payload: true })
      } />
    }
  </View>
  <View style={{ flex: 1 }}>
    <Icon name="skip-next" size={50} onPress={() =>
      nextSong()
    } />
  </View>
</View>
<Video
  source={{ uri: `${DATA_URL}mp3/${redux.track !== null ? redux.track.trackUrl : null}`
}}
  // ref={player}
  // volume={1.0}
  muted={false}
  paused={redux.statusPlaying}
  playInBackground={true}
  playWhenInactive={true}
  // onProgress={this.onPlayProgress}
  onEnd={nextSong}
  resizeMode="cover"
  repeat={false}
/>
</View>
)
}

export default PlayerWidget;

```

Conclusion

- Our project is a humble venture to provide a better tool to enhance user's experience.
- Several user-friendly coding has also been adopted to improve any downsides.
- This package shall prove to be a helpful package to engage a user.
- The objective of software planning is to provide a framework that enables the manager to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses.

Future Enhancement

1. Updating UI for better User Friendly Experience.
2. Enter Lyrics of All Song.
3. Categories songs for better user experience.

Reference

1. <http://www.stackoverflow.com>
2. <http://www.w3school.com>
3. <http://www.tutorialspoints.com>
4. <http://www.Quara.com>