

VISVESVARAYA TECHNOLOGICAL UNIVERSITY BELGAUM-590014



A Mini-Project Report

On

“MUSIC MANAGEMENT SYSTEM”

*A Mini-project report submitted in partial fulfilment of the requirements for the award of the degree of **Bachelor of Engineering in Computer Science and Engineering** of Visvesvaraya Technological University, Belgaum.*

Submitted by:

AMAN HARIS
AKSHAY KUMAR H.S
DEVRAJ.R
B.R SHASHANK

(1AM18CS016)
(1AM18CS013)
(1AM18CS049)
(1AM18CS037)

Under the Guidance of:

Mrs. Veena Bhat

(Asst. Prof., Dept. of CSE)



Department of Computer Science and Engineering
AMC Engineering College,
18th K.M, Bannerghatta Main Road, Bangalore-560 083
2020-2021

AMC Engineering College,
18th K.M, Bannerghatta Main Road, Bangalore-560 083

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



CERTIFICATE

This is to certify that the mini-project work entitled “**MUSIC MANAGEMENT SYSTEM**” has been successfully carried out by **AMAN HARIS (1AM18CS016), AKSHAY KUMAR H.S (1AM18CS013), DEVRAJ R (1AM18CS049) & B.R SHASHANK (1AM18CS037)** bonafide students of **AMC Engineering College** in partial fulfilment of the requirements for the award of degree in **Bachelor of Engineering in Computer Science and Engineering** of **Visvesvaraya Technological University, Belgaum** during academic year 2020-2021. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental Attendance. The mini project report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

Guide:

Mrs Veena Bhat
Asst. Prof., Dept. of CSE

Dr. LATHA C A
HOD, Dept. of CSE

Dr. A.G NATARAJ
Principal, AMCEC

Examiners:

1.

2.

Signature with Date

ACKNOWLEDGEMENT

The joy and satisfaction that accompany the successful completion of any task would be incomplete without the mention of those who made it possible. We are glad to express our gratitude towards our prestigious institution **AMC ENGINEERING COLLEGE** for providing us with utmost knowledge, encouragement and the maximum facilities in undertaking this project.

We wish to express our sincere thanks to our respected chairman **Dr. K. R Paramahamsa** and beloved principal **Dr. A G NATARAJ** for all their support.

We express our deepest gratitude and special thanks to **Dr. Latha C A, H.O.D, Dept. Of Computer Science Engineering**, for all her guidance and encouragement.

We sincerely acknowledge the guidance and constant encouragement of our mini- project guide, **Mrs. Veena Bhat, Assistant Prof., Dept. Of Computer Science Engineering.**

AMAN HARIS
AKSHAY KUMAR H.S
DEVRAJ.R
B.R SHASHANK

(1AM18CS016)
(1AM18CS013)
(1AM18CS049)
(1AM18CS037)

ABSTRACT

The entitled project “**MUSIC MANAGEMENT SYSTEM**” is made keeping in mind all the aspects of the Events. The system allows only registered users to login and new users are allowed to register on the application. The Music Database project is to categorize and catalog every single piece of music for music info and albums, internet music service, listing record collections. Why did we choose it? The idea of a music database arose out of common interest of project members in Music . The concept seemed different and very interesting right in the first go .The project provides most of the basic functionality required for a music store. Since all the work that is to be done by this software can also be done manually, but this consumes time, man power and energy. So, this software will be a relief to those who have to do all this work manually. The knowledge of computers and programming has become a basic skill needed to survive in present society.

CONTENTS

Title	Page No
INTRODUCTION.....	6
<ul style="list-style-type: none">● Introduction● Objectives● Scope	
SYSTEM SPECIFICATION.....	7
<ul style="list-style-type: none">● Hardware requirements● Software requirements	
DESIGN.....	8
<ul style="list-style-type: none">● Description● Normalisation	
IMPLEMENTATION AND CODING.....	13
<ul style="list-style-type: none">● Source code	
SNAPSHOTS.....	22
CONCLUSION.....	31
REFERENCES.....	31

Chapter 1

INTRODUCTION

This application is used by two users:

Admin

User

The main aim is to provide an easier access to music activities to the User : Logins, new subscriptions, create playlist, add and search songs.

The Admin can login to the login page and can add songs, add albums and manage users.

The Admin updates the albums and songs, as only Admin are authorized.

“MUSIC MANAGEMENT SYSTEM” has been designed to computerize the following functions that are performed by the system:

- Music details
- Subscription details
- Login - Logout details

1.1 OBJECTIVES

The **main objective is** to develop Music Library Management MySQL Project is to overcome the manual errors and make a computerized system. In this project, there are various types of modules available to manage Music, Album and Playlist.

1.2 SCOPE

The scope of the project is a system on which software is installed i.e. the project is developed as a desktop application and it will work for an institute but later the project can be modified to operate online and for various organizations.

Chapter 2

SYSTEM SPECIFICATION

2.1 HARDWARE REQUIREMENTS

PROCESSOR: Intel i7 7th gen

RAM: 8GB

HARD DISK: 1TB

2.2 SOFTWARE REQUIREMENTS

OS: Microsoft Windows 10, 64 bit

CODING LANGUAGE:

- FRONT END: NETBEANS IDE 8.0.2

It provides drivers for ORACLE database servers so as to connect to the databases very easily.

- BACK END: XAMPP v3.2.4

It is a relational database whose components (tables, forms, queries) are linked (related). The linkages between database components are created by making relationship links between them. The relationship can be:

- ◆ One component and another (one-one relationship)
- ◆ One component related to several other components(one-many)
- ◆ Several database components(many-many)

Creation of relationships between the database components reduces data redundancies and enhances ease of access of the information.

Chapter 3

DESIGN

3.1 Description of Music Management System

The system should be designed in such a way that only authorized people should be allowed to access some particular modules. The records should be modified by only administrators and no one else. The user should always be in control of the application and not the vice versa. The user interface should be consistent so that the user can handle the application with ease and speed. The application should be visually, conceptually clear.

STEP 1:

ENTITIES ARE:

Admin

User

Album

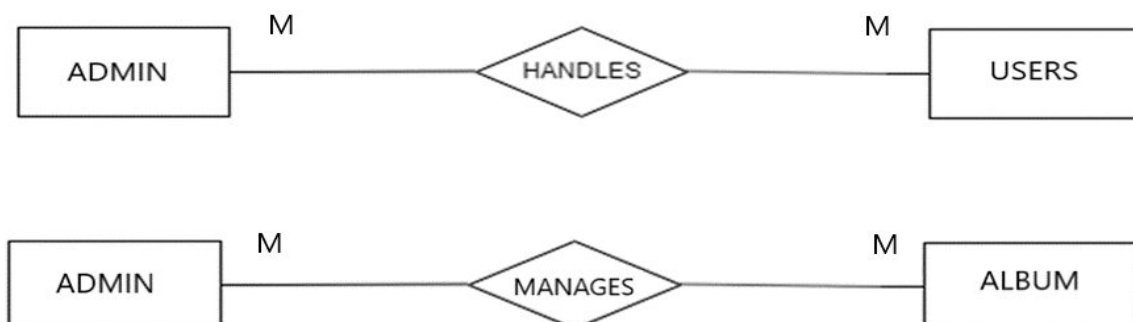
Songs

Playlist

Subscription

STEP 2:

RELATIONS:



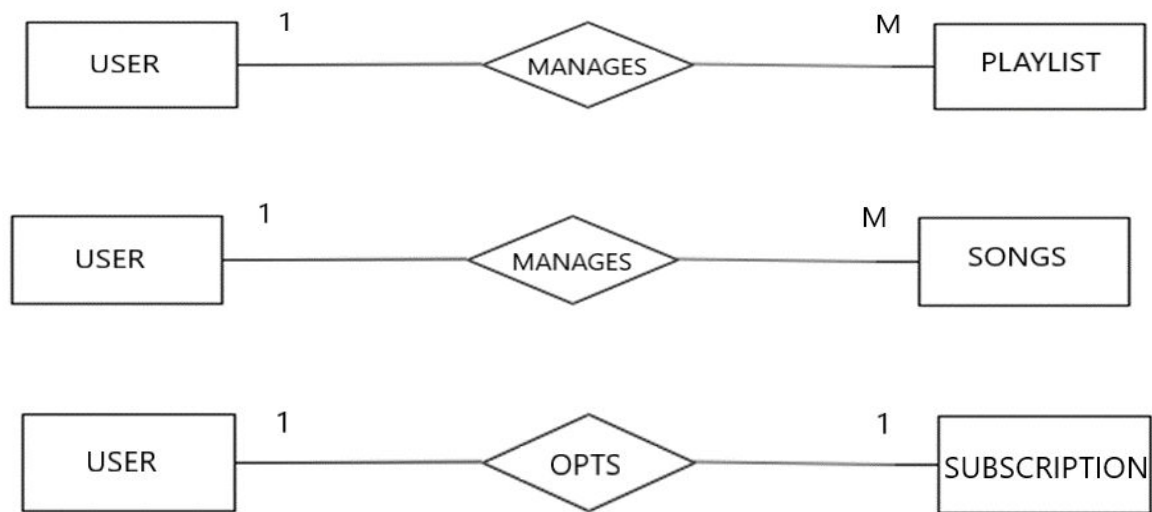


Fig.3.1 Relations

STEP 3:

OTHER ATTRIBUTES:

Admin- admin_id, admin_name, admin_mobile, admin_password

User- user_id, user_name, password, user_mobile, DOB

Album- album_id, album_name, artist, year

songs- song_id, album_id, name, duration

Playlist- user_id, name, song_id, playlist_id, date

Subscription- User_id, payment_id, sub_date, admin_id

SCHEMA DIAGRAM

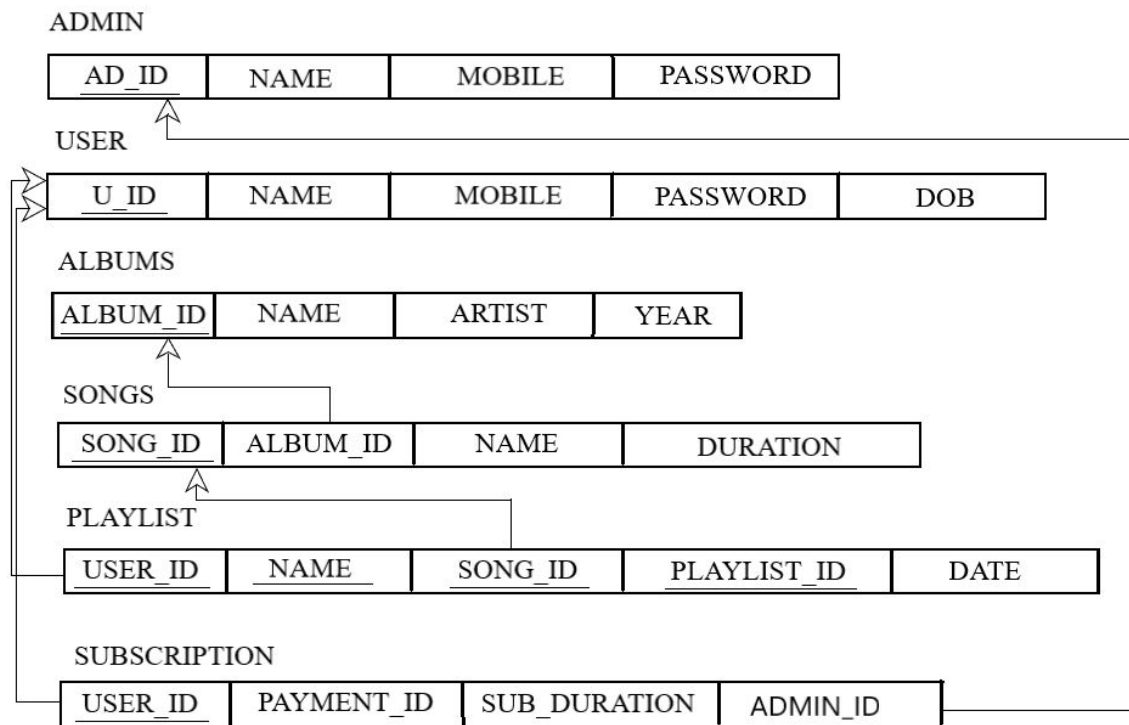


Fig.3.2 Schema diagram of Music Management System

MUSIC MANAGEMENT SYSTEM

STEP 4:

ER DIAGRAM

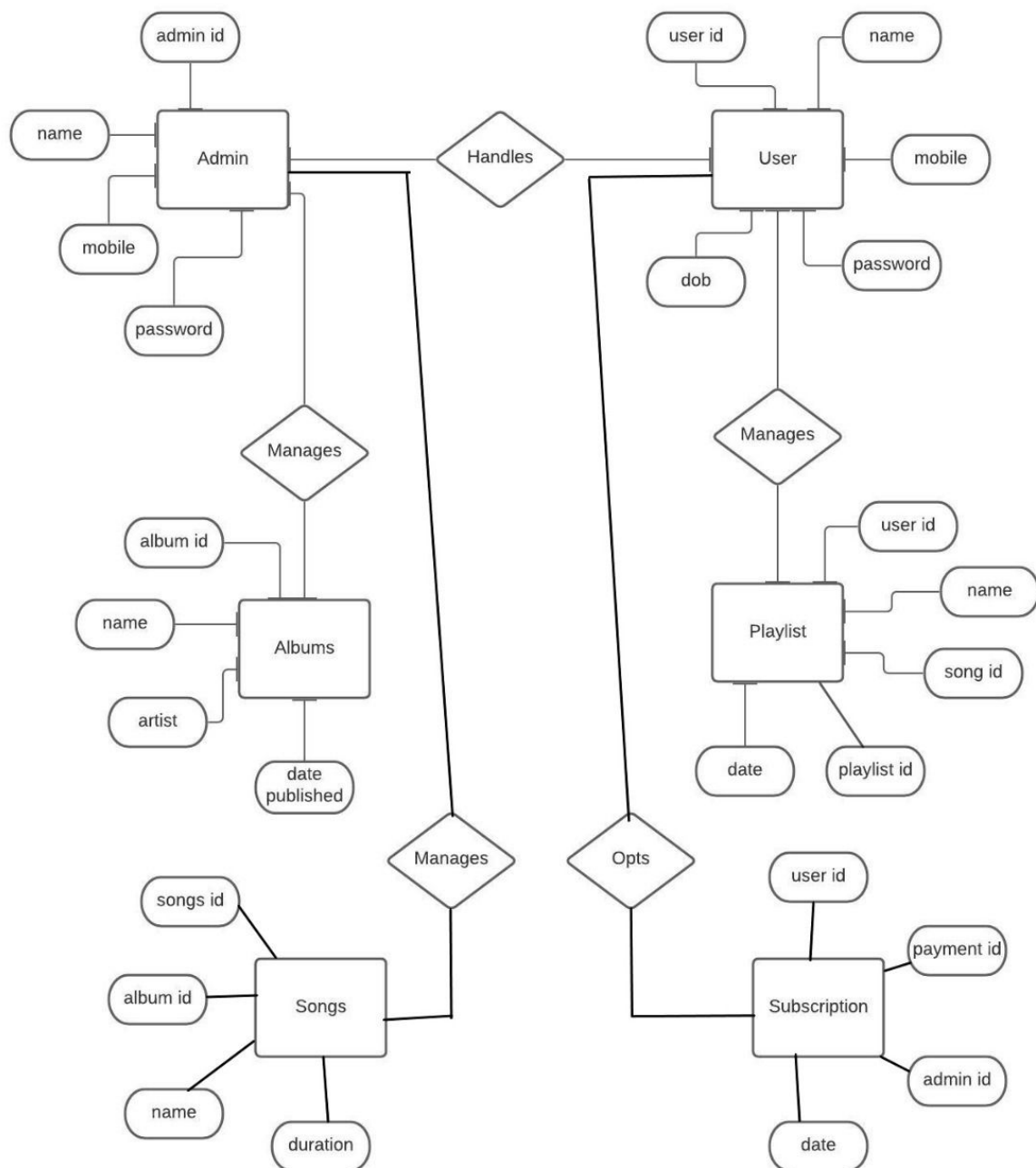


Fig.3.3 ER diagram of Attendance Management System

3.2 NORMALISATION

The basic Objective of normalization is to reduce redundancy, which means that information is to be stored only once. Storing information several times leads to wastage of storage space and increase in the total size of data stored. Relations are normalized so that when relations in the database are to be altered during the lifetime of the database, information is not lost or introduces inconsistencies. The type of alterations normally needed for relation is:

Insertion of new data values to relation. This should be possible without being forced to leave blank fields for some attributes.

Deletion of a tuple, namely, a row of a relation. This should be possible without losing vital information unknowingly.

Functional Dependency:

As the concept of dependency is very important, it is essential that it should be understood first and then proceed to the idea of normalization. There is no fool-proof algorithmic method of identifying dependency.

Properties of normalized relations:

Ideals relation after normalization should have the following properties:

No data values should be duplicated in different rows unnecessarily.

A value must be specified (and required) for every attribute in a row.

Each relation should be self-contained. In other words, if a row from a relation is deleted, important information should not be accidentally lost.

When a row is added to a relation, other relations in the database should not be affected.

A value of an attribute in a tuple may be changed independent of other tuples in the relation and other relations.

Chapter 4

IMPLEMENTATION AND CODING

4.1 SOURCE CODE

CODE FOR ADMIN LOGIN:

```
String user;
    user = jTextField1.getText();
String pwd;
    pwd = new String (jPasswordField1.getPassword());
if("admin".equals(user) && "admin".equals(pwd)){
    new AdminPage().setVisible(true);
    this.setVisible(false); }
else {
String msg;
    msg = jTextField1.getText();
String username;
    username = jTextField1.getText();
String pwsd;
    pwsd = new String (jPasswordField1.getPassword());

PreparedStatement stmt=null;
ResultSet rs=null;
int val = 0;
try {
    Class.forName("com.mysql.jdbc.Driver");
    PreparedStatement ps;
    Connection
conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");
    stmt = conn.prepareStatement("select name from admin");
    rs = stmt.executeQuery();
    while(rs.next()){
        if(username.equals(rs.getString(1))){
            val = 0;
            break;
        }else {
            val=1;
        }
    }

    if(val==1) {
        JOptionPane.showMessageDialog(this,"Enter valid ID and password");
    }
    else if(val==0) {
        stmt = conn.prepareStatement("select password from admin where
name=?");
```

```

stmt.setString(1,username);
rs=stmt.executeQuery();
val=0;
while(rs.next()){
    if(pwsd.equals(rs.getString(1))){
        AdminPage admin;
        admin = new AdminPage();
        admin.setVisible(true);
        this.setVisible(false);
    }else{
        JOptionPane.showMessageDialog(this,"Enter valid id and password");
    }
}
}
}
} catch(ClassNotFoundException | SQLException | HeadlessException e){
JOptionPane.showMessageDialog(null,e);}
}

```

CODE TO ADD ADMIN:

```

try
{
    Class.forName("com.mysql.jdbc.Driver");
    Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");
    String sql= "INSERT INTO
admin(ad_id,name,mobile,password)VALUES(?,?,?,?)";
    PreparedStatement ps=conn.prepareStatement(sql);
    ps.setString(1,jTextField1.getText());
    ps.setString(2,jTextField4.getText());
    ps.setString(3,jTextField5.getText());
    ps.setString(4,jPasswordField2.getText());

    ps.executeUpdate();

    ps.close();
    JOptionPane.showMessageDialog(this, " Admin Added","Admin
Added",JOptionPane.INFORMATION_MESSAGE);

    System.out.println("Admin Added");

}
catch(Exception e)
{
    JOptionPane.showMessageDialog(null,e);
}

```

CODE FOR USER LOGIN:

```

String user;
    user = jTextField1.getText();
String pwd;
    pwd = new String (jPasswordField1.getPassword());
if("users".equals(user) && "users".equals(pwd)){
    new UserPage().setVisible(true);
    this.setVisible(false); }
else {
    String msg;
        msg = jTextField1.getText();
    String username;
        username = jTextField1.getText();
    String pwsd;
        pwsd = new String (jPasswordField1.getPassword());

    PreparedStatement stmt=null;
    ResultSet rs=null;
    int val = 0;
    try {
        Class.forName("com.mysql.jdbc.Driver");
        PreparedStatement ps;
        Connection
conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");
        stmt = conn.prepareStatement("select name from users");
        rs = stmt.executeQuery();
        while(rs.next()){
            if(username.equals(rs.getString(1))){
                val = 0;
                break;
            }else {
                val=1;
            }
        }

        if(val==1) {
            JOptionPane.showMessageDialog(this,"Enter valid ID and password");
        }
        else if(val==0) {
            stmt = conn.prepareStatement("select password from users where
name=?");
            stmt.setString(1,username);
            rs=stmt.executeQuery();
            val=0;
            while(rs.next()){
                if(pwsd.equals(rs.getString(1))){
                    UserPage admin;
                    admin = new UserPage();
                    admin.setVisible(true);
                    this.setVisible(false);
                }else{

```

```

        JOptionPane.showMessageDialog(this,"Enter valid id and password");
    }
}
}
} catch(ClassNotFoundException | SQLException | HeadlessException e){
JOptionPane.showMessageDialog(null,e);}
}

```

CODE TO ADD USER:

```

try
{
    Class.forName("com.mysql.jdbc.Driver");
    Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");
    String sql= "INSERT INTO
users(u_id,name,mobile,dob,admin_id,password)VALUES(?,?,?,?,?)";
    PreparedStatement ps=conn.prepareStatement(sql);
    ps.setString(1,jTextField1.getText());
    ps.setString(2,jTextField2.getText());
    ps.setString(3,jTextField3.getText());
    ps.setString(4,jTextField4.getText());
    ps.setString(5,jTextField5.getText());
    ps.setString(6,jPasswordField1.getText());
    ps.executeUpdate();

    ps.close();
    JOptionPane.showMessageDialog(this, " User Added","User
Added",JOptionPane.INFORMATION_MESSAGE);

    System.out.println("User Added");

}
catch(Exception e)
{
    JOptionPane.showMessageDialog(null,e);
}

```

CODE TO ADD ALBUM:

```

try
{
    Class.forName("com.mysql.jdbc.Driver");
    Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");

```



```

        String sql= "INSERT INTO
albums(album_id,name,artist,admin_id,date_published)VALUES(?,?,?,?)
";
        PreparedStatement ps=conn.prepareStatement(sql);
        ps.setString(1,jTextField1.getText());
        ps.setString(2,jTextField2.getText());
        ps.setString(3,jTextField3.getText());
        ps.setString(4,jTextField4.getText());
        ps.setString(5,jTextField5.getText());

        ps.executeUpdate();

        ps.close();
        JOptionPane.showMessageDialog(this, "Album Added", "Album
Added",JOptionPane.INFORMATION_MESSAGE);

        System.out.println("Album Added");

    }
    catch(Exception e)
    {
        JOptionPane.showMessageDialog(null,e);
    }

}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
new AlbumsList().setVisible(true);
this.setVisible(false);
}

```

CODE TO ADD SONG:

```

try
{
    Class.forName("com.mysql.jdbc.Driver");
    Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");
    String sql= "INSERT INTO
songs(song_id,song_name,album_id,duration)VALUES(?,?,?,?)";
    PreparedStatement ps=conn.prepareStatement(sql);
    ps.setString(1,jTextField1.getText());
    ps.setString(2,jTextField2.getText());
    ps.setString(3,jTextField3.getText());
    ps.setString(4,jTextField4.getText());

    ps.executeUpdate();

    ps.close();
}

```

```
JOptionPane.showMessageDialog(this, "Song Added", "Song
Added",JOptionPane.INFORMATION_MESSAGE);
```

```
System.out.println("Song Added");
```

```
}
catch(Exception e)
{
    JOptionPane.showMessageDialog(null,e);
}
```

CODE TO SEARCH SONGS:

```
Connection conn;
PreparedStatement ps;
try
{
    ResultSet rs;
    Class.forName("com.mysql.jdbc.Driver");
    conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");
    String ss;
    ss = jTextField1.getText();
    ps=conn.prepareStatement("select * from songs where song_name =
?");
    ps.setString(1,jTextField1.getText());
    rs=ps.executeQuery();
    ResultSetMetaData rsm = rs.getMetaData();
    int c;
    c=rsm.getColumnCount();
    DefaultTableModel df=( DefaultTableModel)jTable1.getModel();
    df.setRowCount(0);
    while(rs.next())
    {
        Vector v2 = new Vector();
        for(int i=1;i<=c;i++)
        {
            v2.add(rs.getString("song_id"));
            v2.add(rs.getString("song_name"));
            v2.add(rs.getString("duration"));
        }
        df.addRow(v2);
    }
}
catch (ClassNotFoundException | SQLException ex) {

}
```

CODE TO CREATE PLAYLIST:

```
try
{
    Class.forName("com.mysql.jdbc.Driver");
    Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");
    String sql= "INSERT INTO
playlist(plist_id,p_name,user_id,song_id,date_created)
VALUES(?,?,?,?,?)";
    PreparedStatement ps=conn.prepareStatement(sql);
    ps.setString(1,jTextField1.getText());
    ps.setString(2,jTextField2.getText());
    ps.setString(3,jTextField3.getText());
    ps.setString(4,jTextField5.getText());
    ps.setString(5,jTextField4.getText());
    ps.executeUpdate();

    ps.close();
    JOptionPane.showMessageDialog(this, " Playlist Created","Playlist
Created",JOptionPane.INFORMATION_MESSAGE);

    System.out.println("Playlist Created");

}
catch(Exception e)
{
    JOptionPane.showMessageDialog(null,e);
}
```

CODE TO SUBSCRIBE:

```
try
{
    Class.forName("com.mysql.jdbc.Driver");
    Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");
    String sql= "INSERT INTO
subscription(payment_id,user_id,sub_date,admin_id)VALUES(?,?,?,?)";
    PreparedStatement ps=conn.prepareStatement(sql);
    ps.setString(1,jTextField1.getText());
    ps.setString(2,jTextField2.getText());
    ps.setString(3,jTextField3.getText());
    ps.setString(4,jTextField4.getText());

    ps.executeUpdate();
```

```

        ps.close();
        JOptionPane.showMessageDialog(this, " Subscribed
        Successfully","Subscribed
        Successfully",JOptionPane.INFORMATION_MESSAGE);

        System.out.println("Subscribed Successfully");

    }
    catch(Exception e)
    {
        JOptionPane.showMessageDialog(null,e);
    }
}

```

CODE TO VIEW PLAYLIST:

```

Connection conn;
PreparedStatement ps;
try
{
    ResultSet rs;
    Class.forName("com.mysql.jdbc.Driver");
    conn =
    DriverManager.getConnection("jdbc:mysql://localhost:3306/music
    store?zeroDateTimeBehavior=convertToNull","root","");
    ps=conn.prepareStatement("SELECT * FROM playlist as A RIGHT
    JOIN songs as B ON A.song_id=B.song_id where p_name = ?");
    ps.setString(1,jTextField1.getText());
    rs=ps.executeQuery();
    ResultSetMetaData rsm = rs.getMetaData();
    int c;
    c=rsm.getColumnCount();
    DefaultTableModel df=( DefaultTableModel)jTable1.getModel();
    df.setRowCount(0);
    while(rs.next())
    {
        Vector v2 = new Vector();
        for(int i=1;i<=c;i++)
        {
            v2.add(rs.getString("song_name"));
            v2.add(rs.getString("duration"));
        }
        df.addRow(v2);
    }
}
catch (ClassNotFoundException | SQLException ex) {
}
}

```

CODE TO VIEW ALBUM:

```

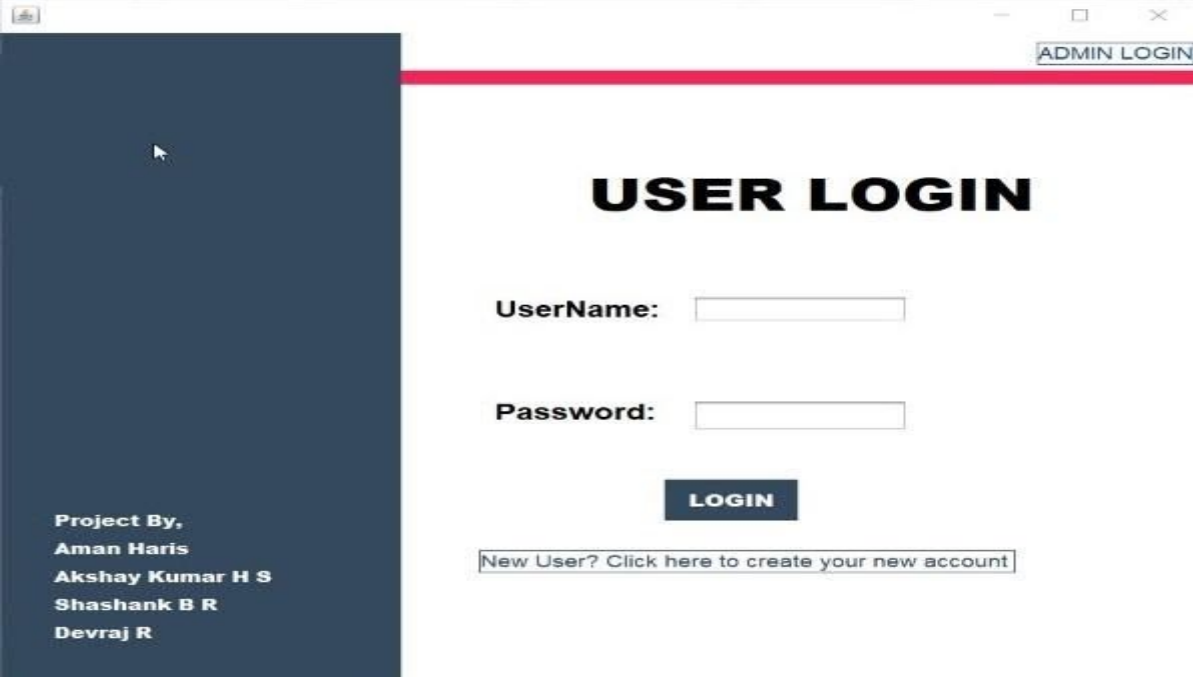
Connection conn;
PreparedStatement ps;
try
{
    ResultSet rs;
    Class.forName("com.mysql.jdbc.Driver");
    conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/music
store?zeroDateTimeBehavior=convertToNull","root","");
    ps=conn.prepareStatement("SELECT * FROM albums as A INNER
JOIN songs as B ON A.album_id=B.album_id where name= ?");
    ps.setString(1,jTextField1.getText());
    rs=ps.executeQuery();
    ResultSetMetaData rsm = rs.getMetaData();
    int c;
    c=rsm.getColumnCount();
    DefaultTableModel df=( DefaultTableModel)jTable1.getModel();
    df.setRowCount(0);
    while(rs.next())
    {
        Vector v2 = new Vector();
        for(int i=1;i<=c;i++)
        {
            v2.add(rs.getString("album_id"));
            v2.add(rs.getString("artist"));
            v2.add(rs.getString("date_published"));
            v2.add(rs.getString("song_name"));
            v2.add(rs.getString("duration"));
        }
        df.addRow(v2);
    }
}
catch (ClassNotFoundException | SQLException ex) {

}

```

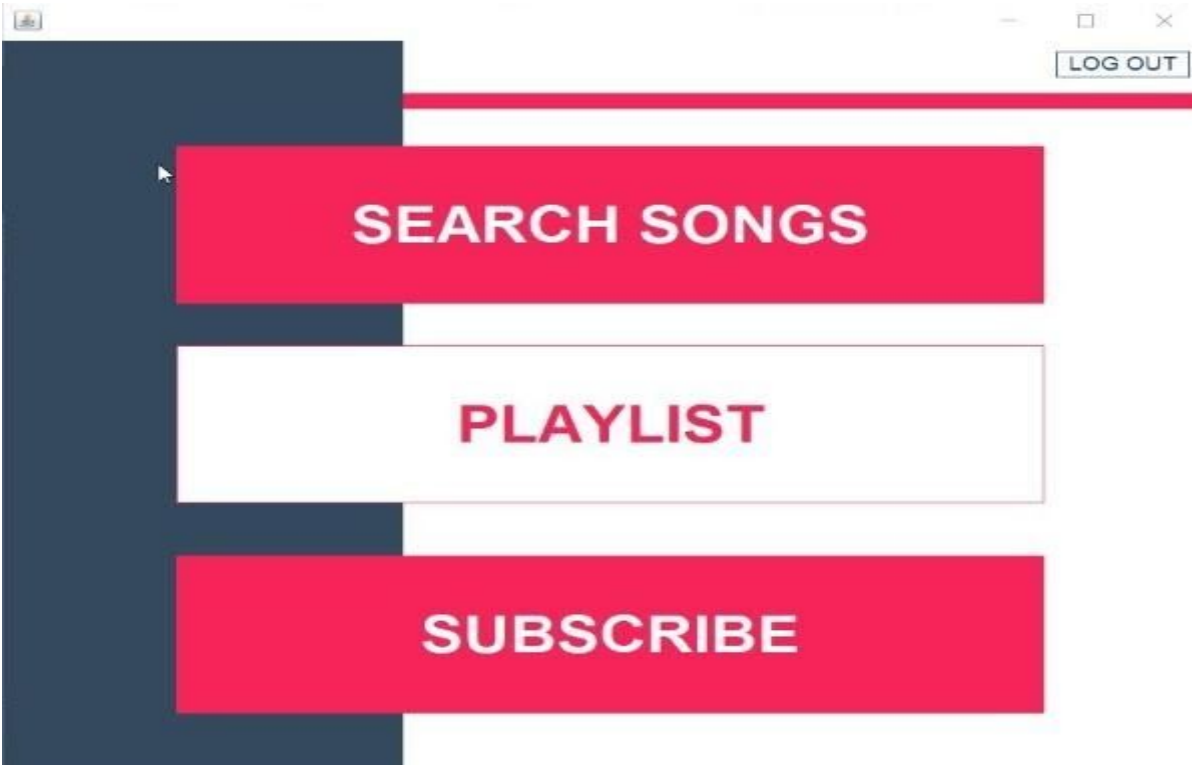
Chapter 5

SNAPSHOTS



The screenshot shows a web browser window with a dark blue sidebar on the left and a white main content area. The sidebar contains the text: "Project By, Aman Haris, Akshay Kumar H S, Shashank B R, Devraj R". The main content area has a red header bar with "ADMIN LOGIN" on the right. Below the header, the title "USER LOGIN" is centered. There are two input fields: "UserName:" and "Password:". Below these is a dark blue "LOGIN" button. At the bottom, there is a link: "New User? Click here to create your new account".

Fig.5.1 User Login



The screenshot shows a web browser window with a dark blue sidebar on the left and a white main content area. The sidebar contains the text: "Project By, Aman Haris, Akshay Kumar H S, Shashank B R, Devraj R". The main content area has a red header bar with "LOG OUT" on the right. Below the header, there are three large buttons: "SEARCH SONGS" (red), "PLAYLIST" (white), and "SUBSCRIBE" (red).

Fig.5.2 User Page

The screenshot shows a web application window titled "MUSIC MANAGEMENT SYSTEM". On the left is a dark blue sidebar with a "SEARCH" button. The main content area has a search bar at the top. Below the search bar is a table with the following headers: "Song ID", "Song Name", and "Duration". The table body is currently empty. At the bottom right of the window is a "BACK" button.

Song ID	Song Name	Duration
---------	-----------	----------

Fig.5.3 Search Songs

The screenshot shows a web application window titled "MUSIC MANAGEMENT SYSTEM". On the left is a dark blue sidebar. The main content area has a heading "CREATE PLAYLIST" followed by a red horizontal line. Below the heading are five form fields: "Playlist ID:", "Playlist Name:", "User ID:", "Song ID:", and "Date:". At the bottom are two buttons: "CANCEL" and "CREATE".

CREATE PLAYLIST

Playlist ID:

Playlist Name:

User ID:

Song ID:

Date:

Fig.5.4 Create Playlist

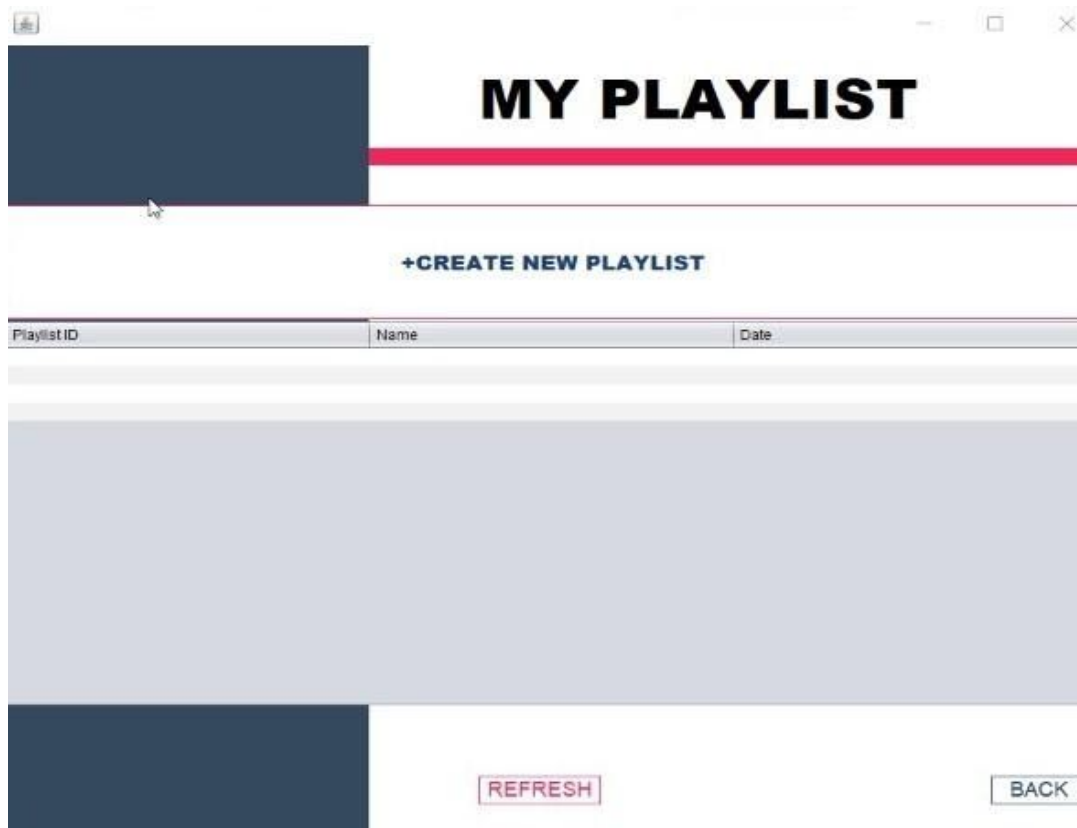


Fig.5.5 My Playlist

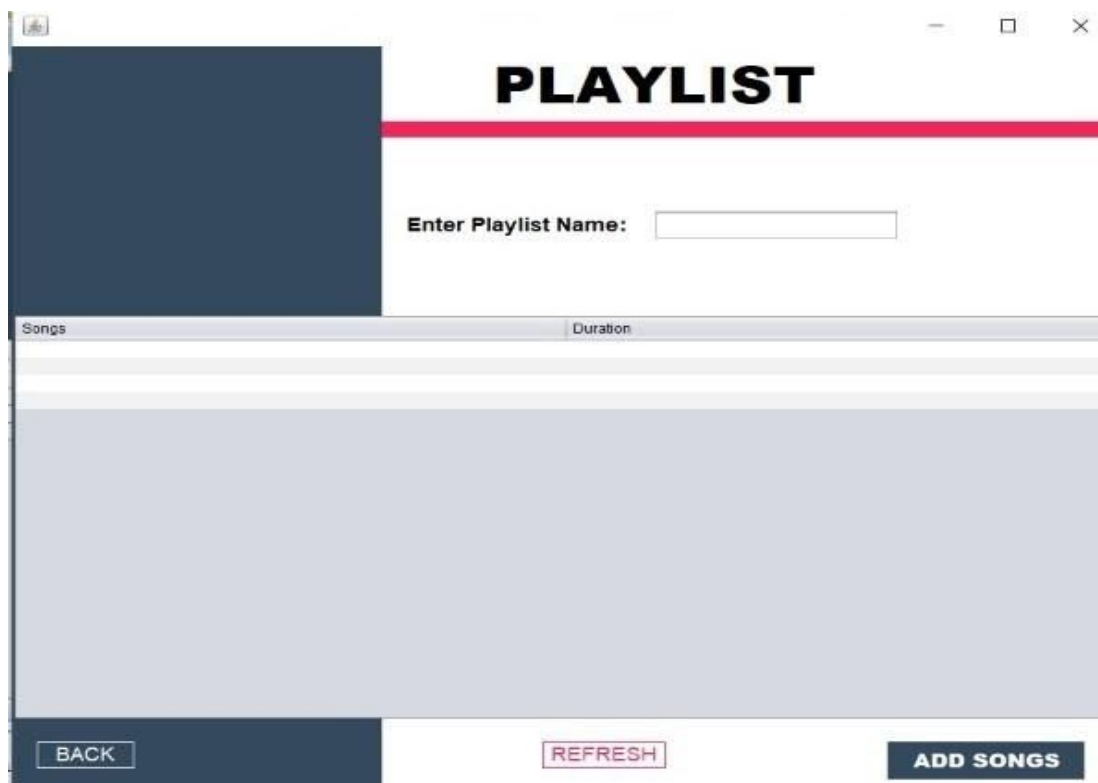
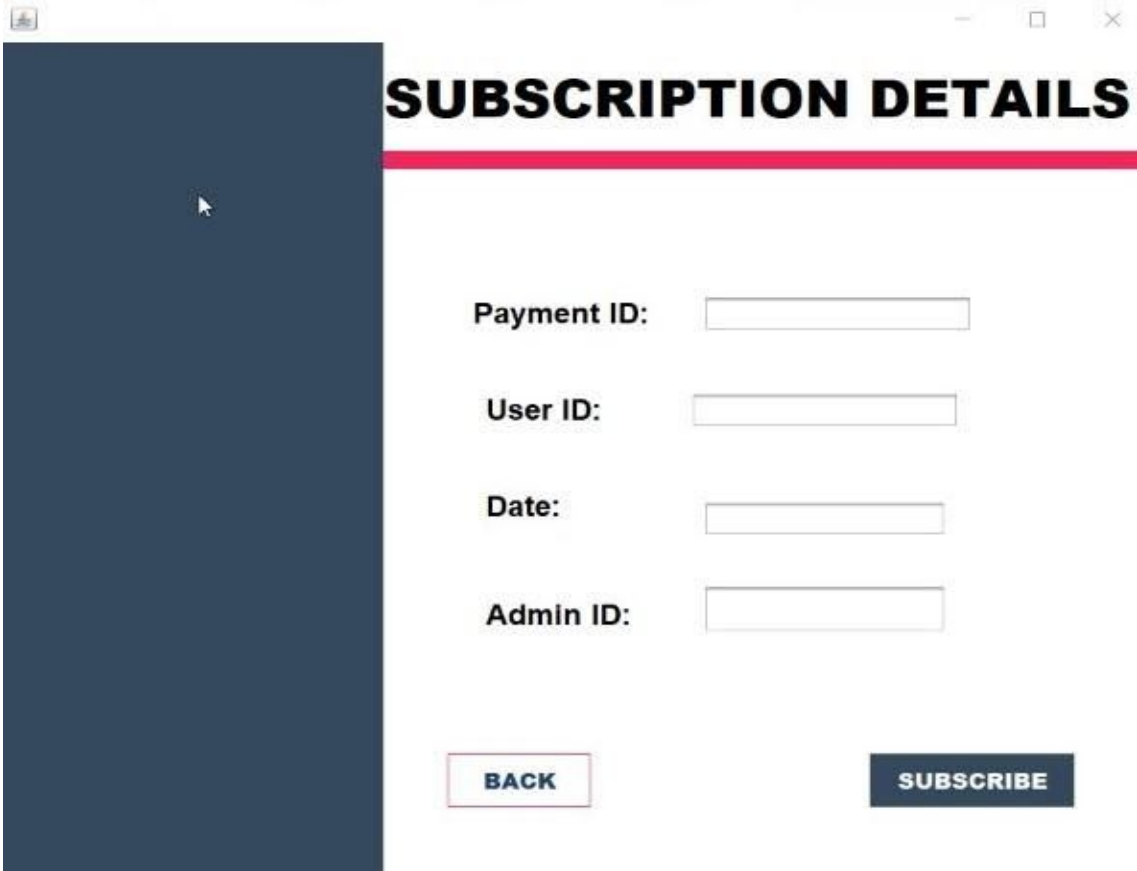


Fig.5.6 Playlist



The screenshot shows a web browser window with a dark blue sidebar on the left. The main content area has a white background with a red header bar. The title "SUBSCRIPTION DETAILS" is in bold black text. Below the title, there are four input fields with labels: "Payment ID:", "User ID:", "Date:", and "Admin ID:". At the bottom, there are two buttons: "BACK" (white with a red border) and "SUBSCRIBE" (dark blue).

SUBSCRIPTION DETAILS

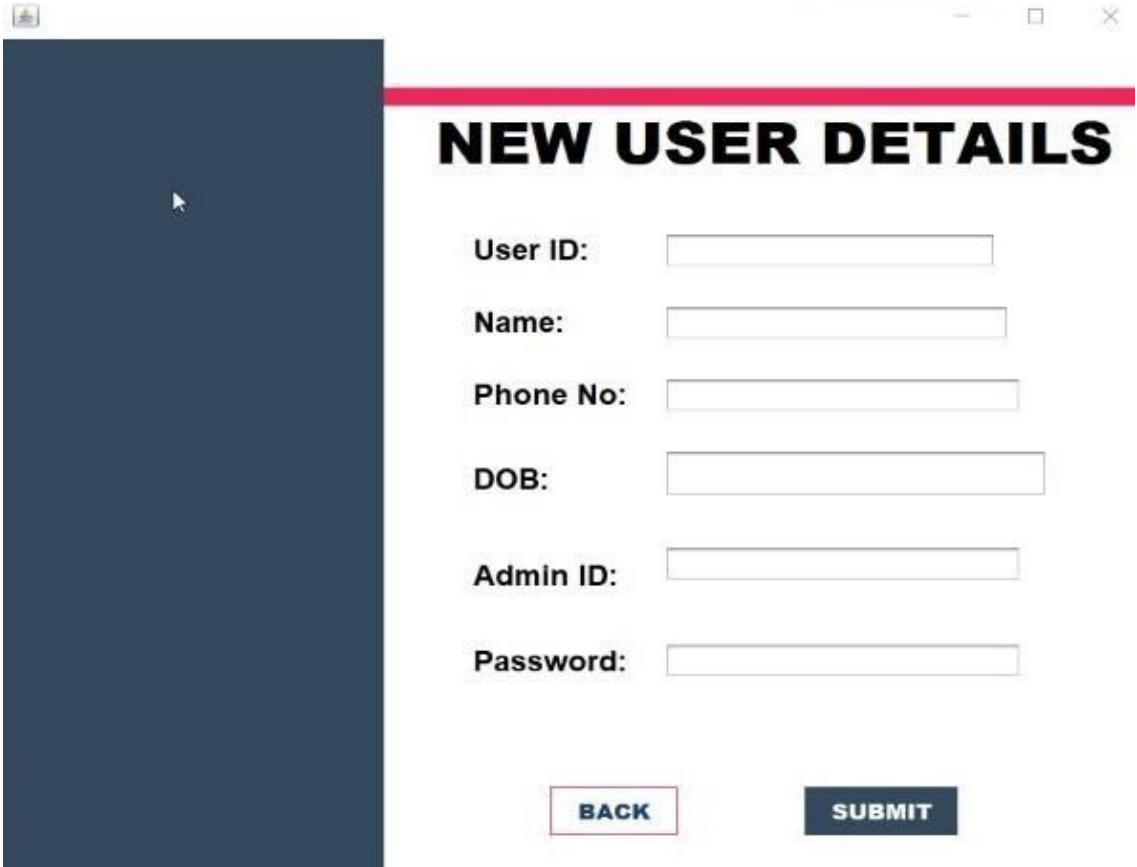
Payment ID:

User ID:

Date:

Admin ID:

Fig.5.7 Subscription



The screenshot shows a web browser window with a dark blue sidebar on the left. The main content area has a white background with a red header bar. The title "NEW USER DETAILS" is in bold black text. Below the title, there are six input fields with labels: "User ID:", "Name:", "Phone No:", "DOB:", "Admin ID:", and "Password:". At the bottom, there are two buttons: "BACK" (white with a red border) and "SUBMIT" (dark blue).

NEW USER DETAILS

User ID:

Name:

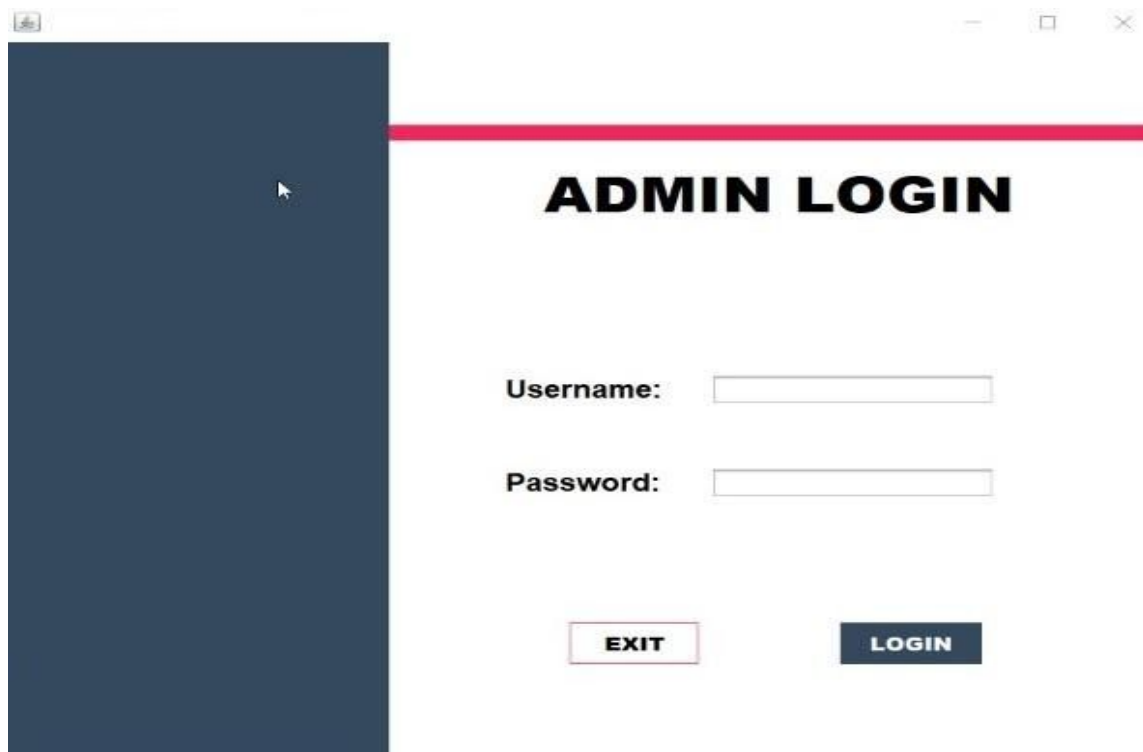
Phone No:

DOB:

Admin ID:

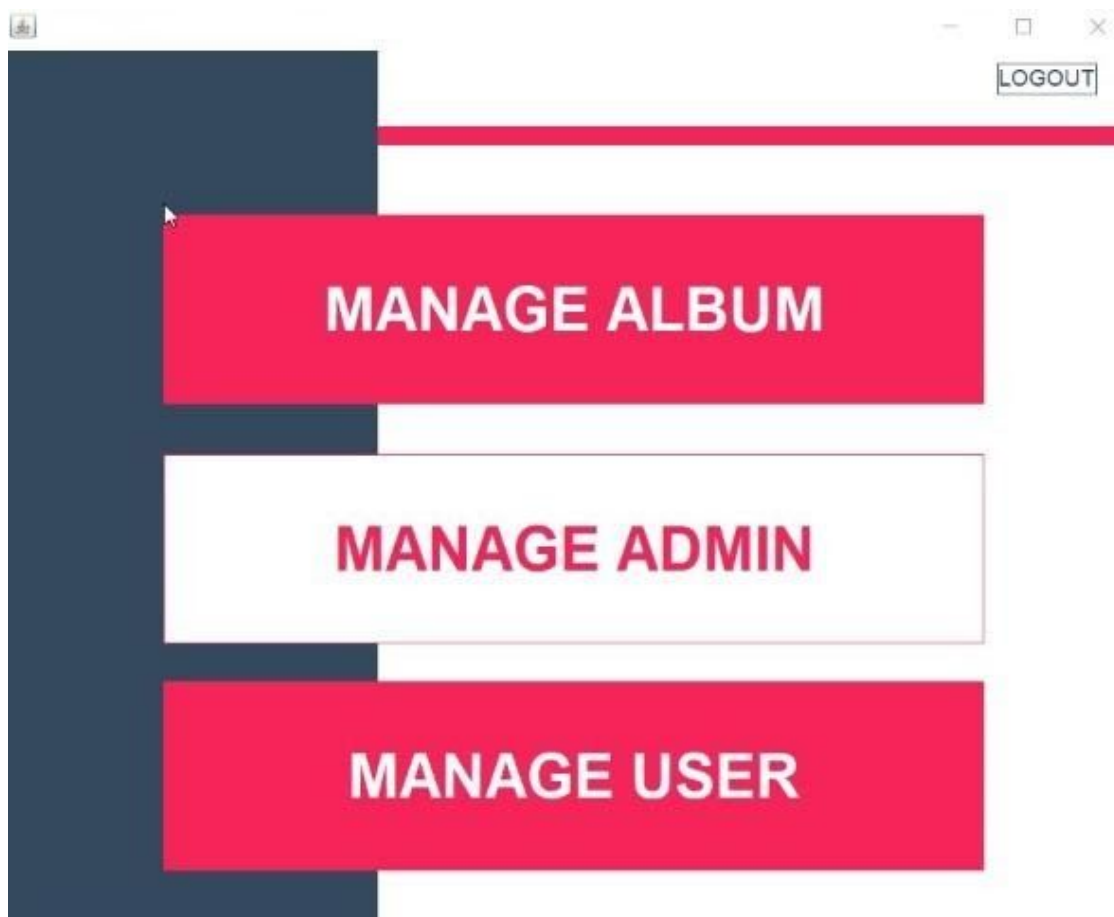
Password:

Fig.5.8 New User



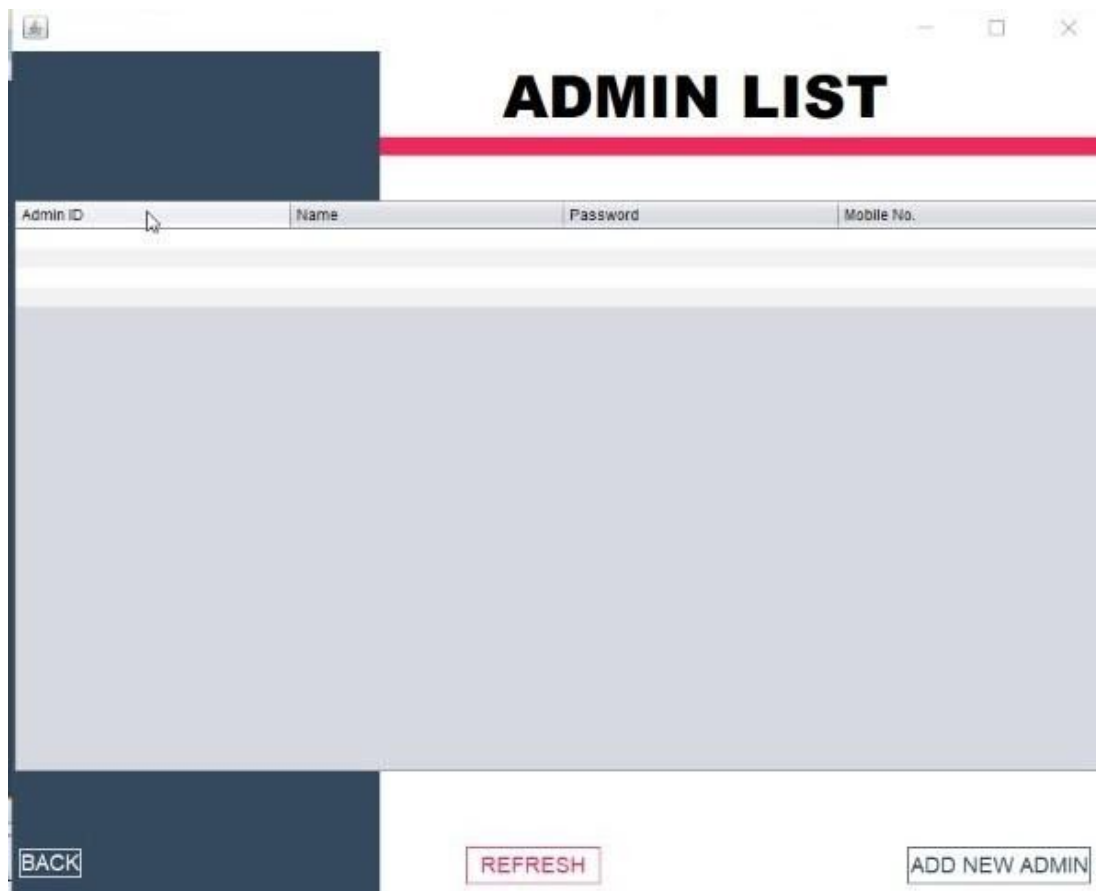
A screenshot of a web browser window displaying the 'ADMIN LOGIN' page. The page has a dark blue sidebar on the left. The main content area has a white background with a red horizontal bar at the top. The title 'ADMIN LOGIN' is centered in bold black text. Below the title are two input fields: 'Username:' and 'Password:'. At the bottom, there are two buttons: 'EXIT' (white with black text) and 'LOGIN' (dark blue with white text).

Fig.5.9 Admin Login



A screenshot of a web browser window displaying the 'Admin Page'. The page has a dark blue sidebar on the left. The main content area has a white background with a red horizontal bar at the top. In the top right corner, there is a 'LOGOUT' button. The main content area contains three large, overlapping rectangular buttons: 'MANAGE ALBUM' (red with white text), 'MANAGE ADMIN' (white with red text), and 'MANAGE USER' (red with white text).

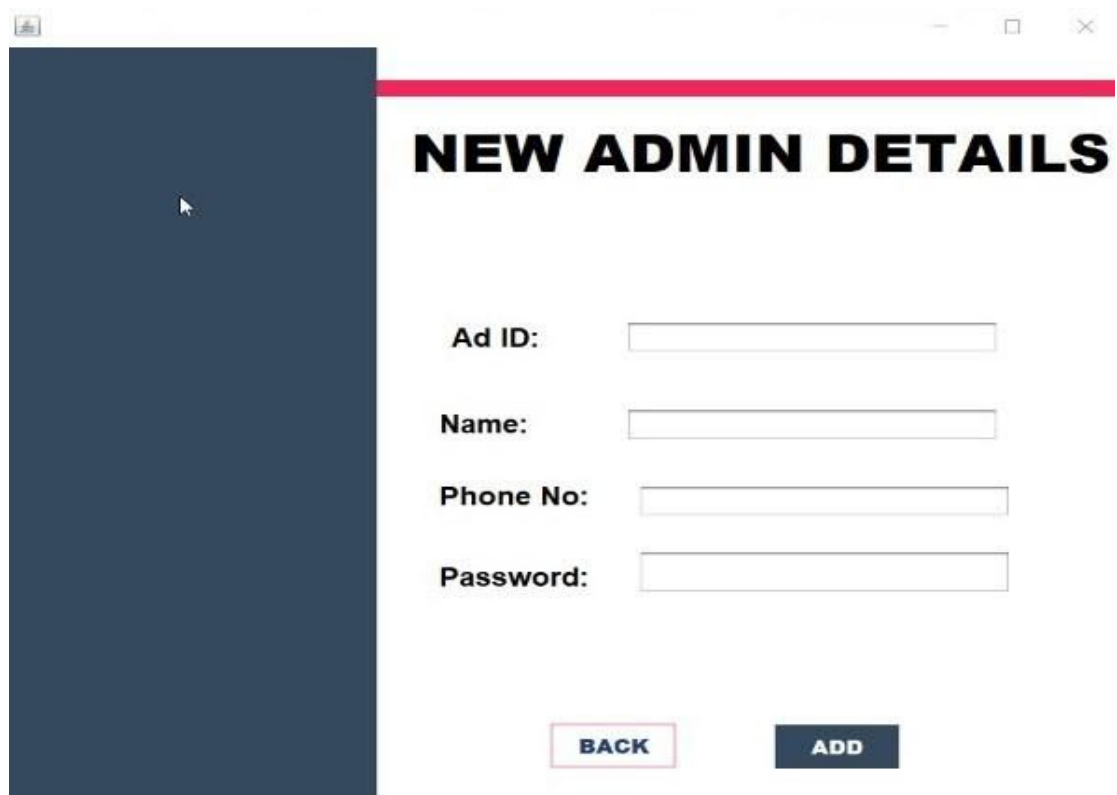
Fig.5.10 Admin Page



The screenshot shows a web application window titled "ADMIN LIST". On the left is a dark blue sidebar. The main content area has a header with the title "ADMIN LIST" in bold black text, underlined by a red horizontal bar. Below the header is a table with four columns: "Admin ID", "Name", "Password", and "Mobile No.". The table body is currently empty. At the bottom of the window, there are three buttons: "BACK" on the left, "REFRESH" in the center, and "ADD NEW ADMIN" on the right.

Admin ID	Name	Password	Mobile No.
----------	------	----------	------------

Fig.5.11 Admin List



The screenshot shows a web application window titled "NEW ADMIN DETAILS". On the left is a dark blue sidebar. The main content area has a header with the title "NEW ADMIN DETAILS" in bold black text, underlined by a red horizontal bar. Below the header are four form fields, each with a label and an input box: "Ad ID:", "Name:", "Phone No:", and "Password:". At the bottom of the window, there are two buttons: "BACK" on the left and "ADD" on the right.

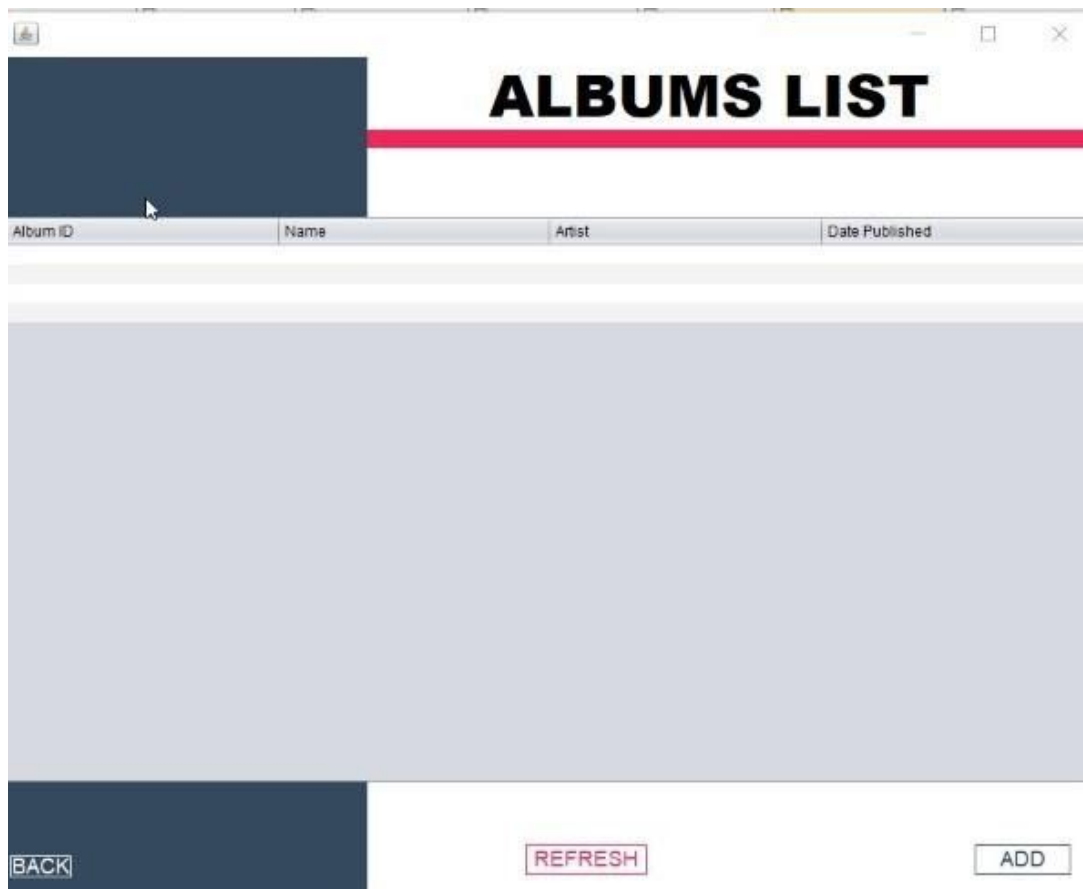
Ad ID:

Name:

Phone No:

Password:

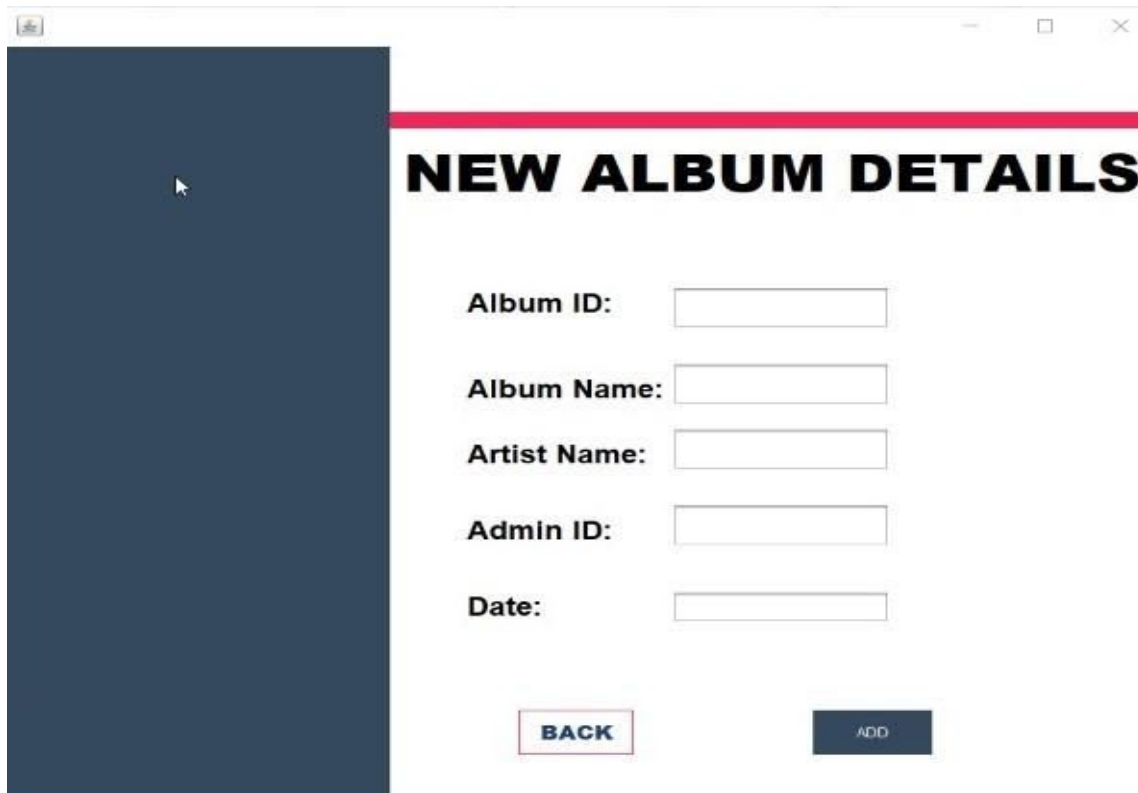
Fig.5.12 New Admin



A screenshot of a web application window titled "ALBUMS LIST". The window has a dark blue sidebar on the left. The main content area has a header with the title "ALBUMS LIST" in bold black text, followed by a thick red horizontal line. Below the header is a table with four columns: "Album ID", "Name", "Artist", and "Date Published". The table body is currently empty, showing a light gray background. At the bottom of the window, there are three buttons: "BACK" (dark blue), "REFRESH" (light blue), and "ADD" (light blue).

Album ID	Name	Artist	Date Published
----------	------	--------	----------------

Fig.5.13 Album List



A screenshot of a web application window titled "NEW ALBUM DETAILS". The window has a dark blue sidebar on the left. The main content area has a header with the title "NEW ALBUM DETAILS" in bold black text, followed by a thick red horizontal line. Below the header are five form fields, each with a label and a text input box: "Album ID:", "Album Name:", "Artist Name:", "Admin ID:", and "Date:". At the bottom of the window, there are two buttons: "BACK" (light blue) and "ADD" (dark blue).

Album ID:

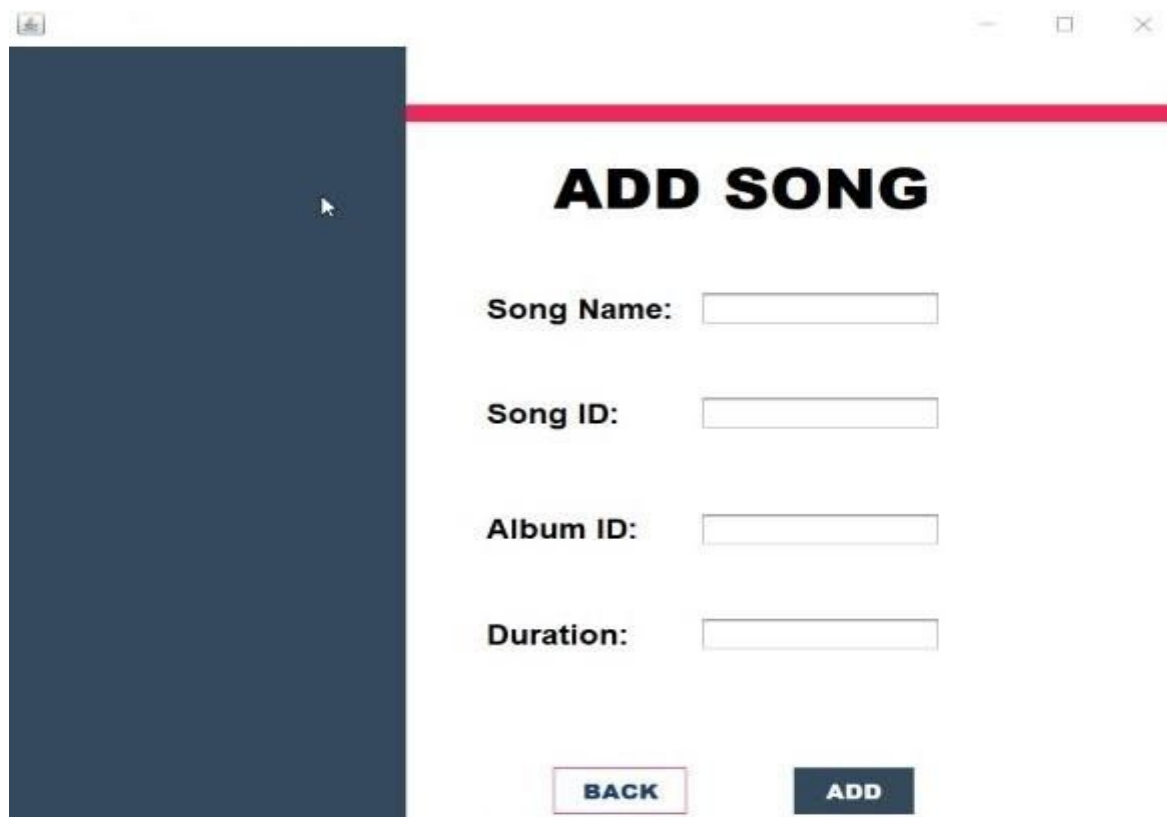
Album Name:

Artist Name:

Admin ID:

Date:

Fig.5.14 New Album



ADD SONG

Song Name:

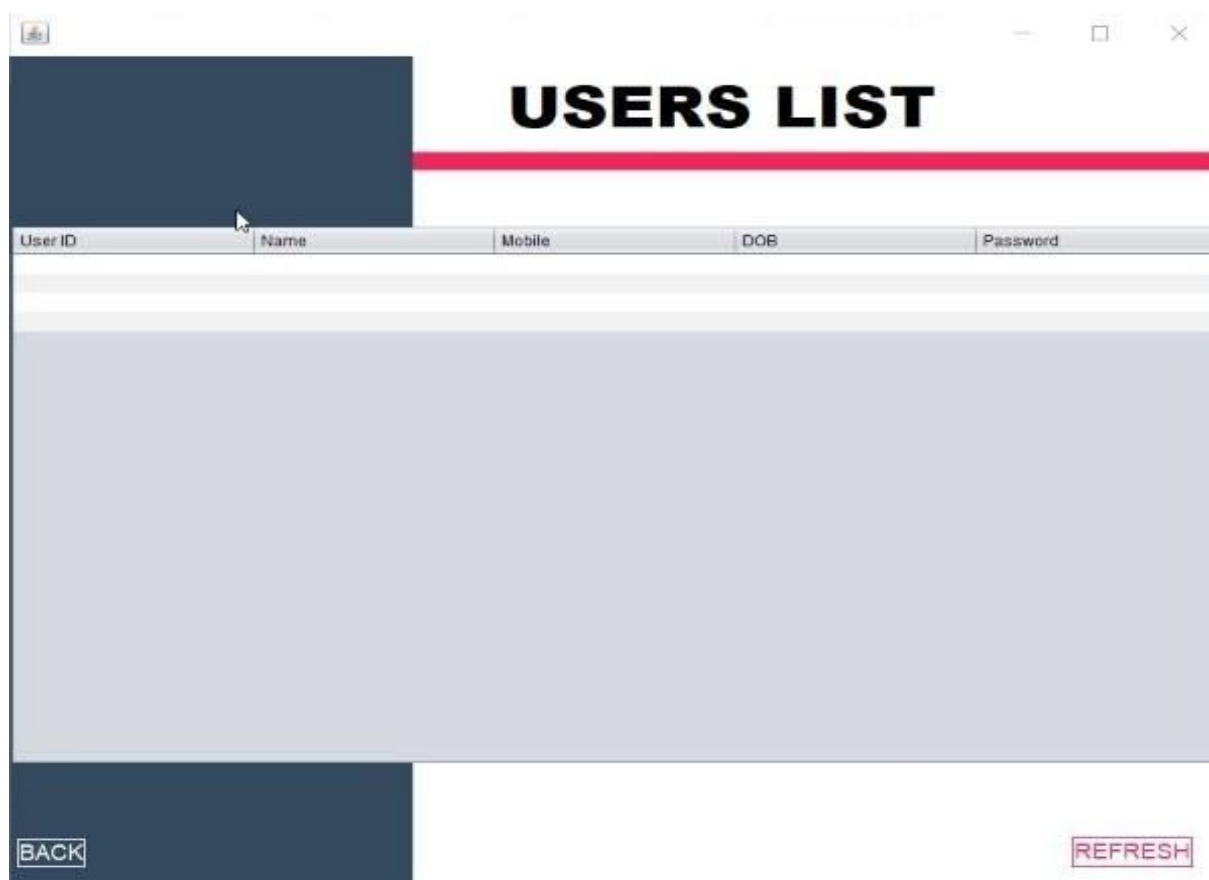
Song ID:

Album ID:

Duration:

BACK **ADD**

Fig.5.15 Add Songs

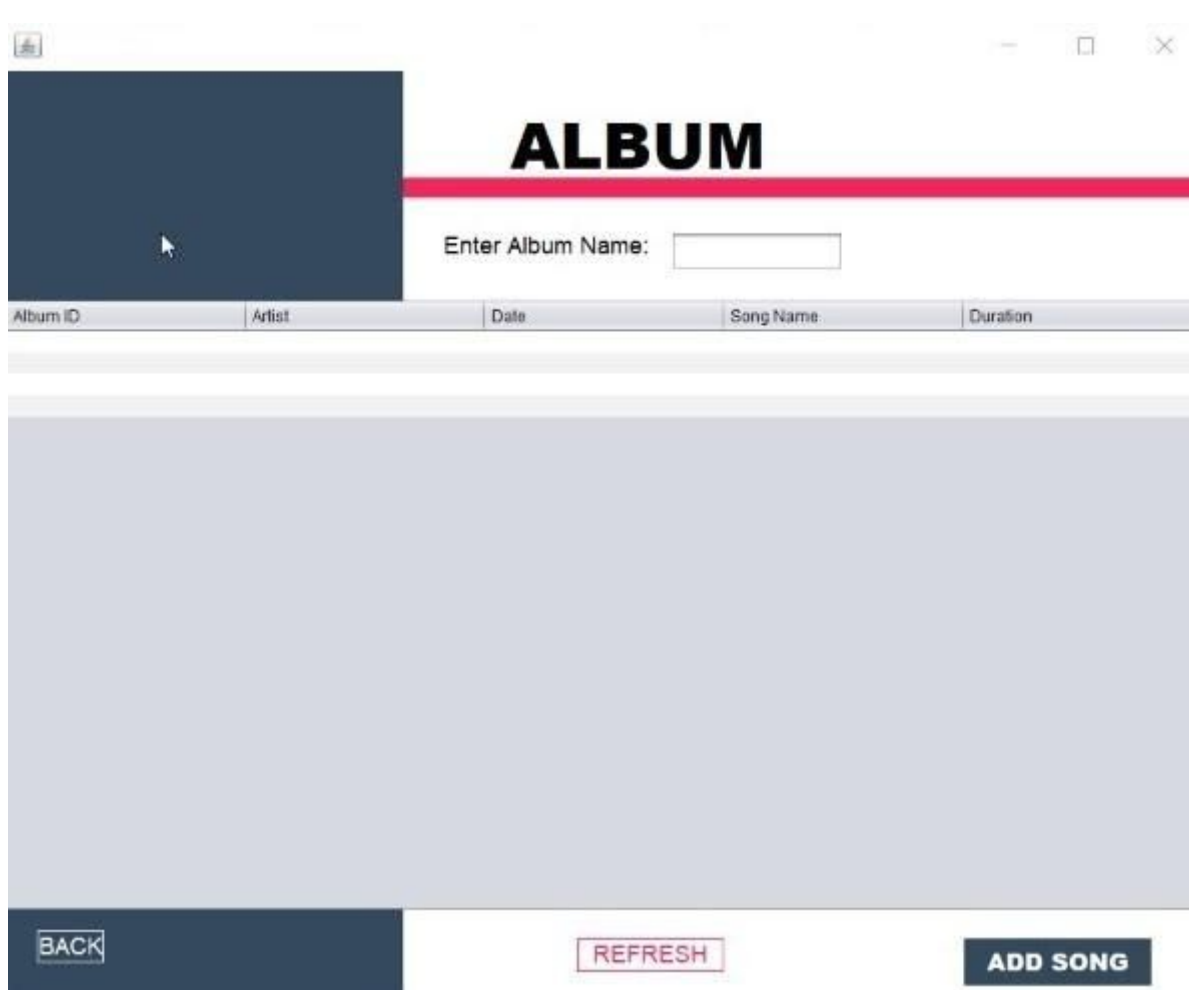


USERS LIST

User ID	Name	Mobile	DOB	Password
---------	------	--------	-----	----------

BACK **REFRESH**

Fig.5.16 User List



The image shows a web application window titled "MUSIC MANAGEMENT SYSTEM". The main heading is "ALBUM" in large, bold, black letters. Below the heading is a red horizontal line. To the left of the heading is a dark blue rectangular area, likely for an album cover. Below the heading is a text input field labeled "Enter Album Name:". Below the input field is a table with five columns: "Album ID", "Artist", "Date", "Song Name", and "Duration". The table is currently empty. At the bottom of the interface are three buttons: "BACK" (dark blue), "REFRESH" (light blue), and "ADD SONG" (dark blue).

ALBUM

Enter Album Name:

Album ID	Artist	Date	Song Name	Duration
----------	--------	------	-----------	----------

[BACK](#) [REFRESH](#) [ADD SONG](#)

Fig.5.17 Album

Chapter 6

CONCLUSION

It gives complete exposure of the requirement of the management for smooth functioning.

There are different forms and tables are used. The data is stored in tables automatically. However, the whole system cannot be changed, but the computerized system designed not only saves time but at the same time reduces labour & expenditures. In traditional systems, there were a lot of irregularities found in generating data to where as in modified and computerized systems in every problem overcome with the press of button. This system provides the security from loss, disclosure, modification and destruction of data. This system provides integrity of proper functioning of programs.

REFERENCES

[1] www.google.com

[2] www.youtube.com