**ISTANBUL TECHNICAL UNIVERSITY**

**COMPUTER AND INFORMATION FACULTY**

**RECOMUSE**

**ISE-305 PROJECT REPORT**

**BARIŞ ARAN**

**150140223**

**DEPARTMENT: INFORMATION SYSTEMS ENGINEERING**

**LECTURER: AHMET CUNEYD TANTUĞ**

**JUNE 2018**

**RECOMUSE PROJECT REPORT**

**INTRODUCTION**

My project for this term is called RecoMuse. RecoMuse is basically a database driven website that recommends music artists to people according to their taste. It provides sign up & login system, has a dashboard that shows most liked artists, has a library that shows the artists that the user liked and has a recommendation page that shows new artist the user may like. In order to sign up, an email, a username and a password is required, and the user should use the username and password to login to RecoMuse. If a new user is not happy with the most liked artists that shows up in the dashboard when he/she first logged in, he/she may use the search bar in the dashboard to like his/her favorite artists so that he/she can get recommendations based on his/her likes. Also, if a user likes an artist by mistake, he/she can remove the artist for his/her library so that it won’t affect his/her recommendations.

The reason that I wanted to do this project like this is that I have never created a website before, especially a database driven one. So, I wanted to use this project as a chance to dive into this kind of developing. I used HTML, CSS, PHP and MySQL while doing this project. I used ATOM as a text editor to write my code, and MAMP for connecting to the database.

You can find the Use Case Diagram and Data Flow Diagram before the images parts. The database of RecoMuse consists of 7 tables. You can see the tables and relations between them on ERD Diagram under Use Case and Data Flow Diagrams.

**INFORMATION**

Let’s assume that you are a new user. The first thing you are going to see in RecoMuse is the login page (Figure 1). When you take a look at the login page, you can see the username and password input boxes, login button and a register link. I made the username by creating a text input, a password input, a submit input and a hyperlink for registration. Here is the code:

***//Login.php***

***<form name="login" method="post">***

***<div class="login">***

***<input type="text" placeholder="Username" name="username"><br>***

***<input type="password" placeholder="Password" name="password"><br>***

***<input type="submit" value="Login"/>***

***<p class="message">Not Registered?<a href="Register.php">Register</a></p>***

***</div>***

***</form>***

Since you are new to RecoMuse, you probably do not have an account, so you should click to “register” link. After you click to the link, register page should be opened. It is almost the same with the login page, except it has an extra email input and has login link instead of register link (Figure 2). Now, you should enter the username and password that you want to use for this website, and your email. After you’ve entered these all correctly, you can click to register button. What happens when you click that button is that the system takes the information that you’ve entered, then inserts it into the database. To do this, I used this code:

***//Register.php***

***//To connect database***

***$con=mysqli\_connect("localhost","root","root");***

***mysqli\_select\_db($con,"recomuse");***

***//To insert into database***

***$result=mysqli\_query($con,"INSERT INTO account\_info(user\_name,user\_password,user\_email) VALUES('$username','$password','$email')") or die(mysqli\_error($con));***

When the registration is complete, the system directs you to a page where you can see you have been successfully registered, if you’ve entered all the values correctly (Figure 3). You can click the login button there to step into login phase. “Registered.php” is used in this part.

Since you have an account now, you can enter the username and password that you have decided before to login to your account. Click login button after you’ve entered the values correctly. You will get an alert if you enter wrong username or password. When you click to login button, what happens is the system searches for the username and password that you’ve entered. The code is like this:

***//Login.php***

***$result=mysqli\_query($con,"SELECT \* FROM account\_info where user\_name= '$username' and user\_password= '$password'") or die(mysqli\_error($con));***

After you’ve logged in successfully, the first thing you will see is the dashboard (Figure 4). In dashboard, the system recognizes your username, and says you “Hello”. The way I captured the username from “Login.php” is like this:

***//Login.php***

***$\_SESSION['user']=$username;***

***//Dashboard.php***

***echo "<h2>Hello ".$\_SESSION['user'].". Here, you can find the most popular music!</h2>";***

Under the message, you can see the top 5 artists, according to their like numbers. You can also see the number of likes that they have under them. You can click on the artist names to see their profiles, which includes information about them and where you can like the artists and add them into your library. I did this part like this:

***//Dashboard.php***

***//Holding both most liked artist names and how many likes they have***

***$result1=mysqli\_query($con,"SELECT artist\_name,COUNT(\*) as counter FROM account\_libary,artist\_info WHERE artist\_info.id=artist\_id GROUP BY artist\_name ORDER BY `counter` DESC LIMIT 5") or die(mysqli\_error($con));***

***if ($result1->num\_rows > 0) {***

***while($row = $result1->fetch\_assoc()) {***

***?>***

***<a class='one' href="Artist.php?name=<?php echo $row['artist\_name']; ?>"><?php echo $row['artist\_name']; ?></a>***

***<?php echo '<div class="likenum">Has '.$row['counter'].' likes.</div>'; ?>***

***<br>***

***<?php***

***}***

***}***

There are 2 buttons located under these. One of them is “My Library” and the other one is “My Recommendations”. These are actually just 2 links that have been modified with CSS. You can see the artists that you liked in “My Library”, and check RecoMuse’s recommendations for you in “My Recommendations”. But, since you didn’t like any artists, you won’t see any artists in your library or recommendations. Here’s how these two links have been made:

***//Dashboard.php***

***<a class='two' href="Libary.php">My Libary</a><br>***

***<a class='two' href="Recommendations.php">My Recommendations</a>***

And lastly, under these 2 buttons, there is a search bar. It is a text type input which you enter an artist you want to add to your library, then get into that artist’s profile. All you should do is type the artist name, then press Enter. You will be directed to the artist profile if there is such artist in the database, or, get an alert if there is no such artist in the database. This is how the search bar works:

***//Dashboard.php***

***//Searching for the artist***

***$search\_result=mysqli\_query($con,"SELECT name FROM artist\_info where name= '$search'") or die(mysqli\_error($con));***

***$row=mysqli\_fetch\_array($search\_result);***

***if($row[0]==$search){***

***header("Location:Artist.php?name=$row[0]");***

***}else {***

***$message = "There is no such artist in our database";***

***echo "<script type='text/javascript'>alert('$message');</script>";***

***}***

So, no matter if you choose one of the most liked artist or use the search bar to find your favorite artist, you will be directed to the artist profile (Figure 5.1-Figure 5.2). I implemented the “Dashboard.php” in such a way that it can pass the clicked artist’s name to the “Artist.php”, so that helps me to access all the information of an artist and use it in the artist profile. This is how I showed the artist informations:

***//Artist.php***

***$userid=mysqli\_query($con,"SELECT id FROM account\_info where account\_info.user\_name='$user'") or die(mysqli\_error($con));***

***$row1=mysqli\_fetch\_array($userid);***

***$name=mysqli\_query($con,"SELECT name FROM artist\_info WHERE name='$selected\_artist'") or die(mysqli\_error($con));***

***$genre1=mysqli\_query($con,"SELECT genre1 FROM artist\_info WHERE name='$selected\_artist'") or die(mysqli\_error($con));***

***$genre2=mysqli\_query($con,"SELECT genre2 FROM artist\_info WHERE name='$selected\_artist'") or die(mysqli\_error($con));***

***$genre3=mysqli\_query($con,"SELECT genre3 FROM artist\_info WHERE name='$selected\_artist'") or die(mysqli\_error($con));***

***$area=mysqli\_query($con,"SELECT area FROM artist\_info WHERE name='$selected\_artist'") or die(mysqli\_error($con));***

***$year=mysqli\_query($con,"SELECT year\_formed FROM artist\_info WHERE name='$selected\_artist'") or die(mysqli\_error($con));***

***$info=mysqli\_query($con,"SELECT info FROM artist\_info WHERE name='$selected\_artist'") or die(mysqli\_error($con));***

***$rowname=mysqli\_fetch\_array($name);***

***echo 'Artist Name: '.$rowname[0].'<br><br>';***

***$rowgen1=mysqli\_fetch\_array($genre1);***

***echo "Genre: ".$rowgen1[0].", ";***

***$rowgen2=mysqli\_fetch\_array($genre2);***

***echo $rowgen2[0].",";***

***$rowgen3=mysqli\_fetch\_array($genre3);***

***echo $rowgen3[0]."<br><br>";***

***$rowarea=mysqli\_fetch\_array($area);***

***echo "Area: ".$rowarea[0]."<br><br>";***

***$rowyear=mysqli\_fetch\_array($year);***

***echo "Year Formed: ".$rowyear[0]."<br><br>";***

***$rowinfo=mysqli\_fetch\_array($info);***

***echo "Info: ".$rowinfo[0]."<br><br>";***

Under these information, first, there are the albums of the artist. If you click on an album, you will be directed to another page which includes the songs of the album, and if you click to a song you will be directed to Youtube. This is how I implemented it:

***//Artist.php***

***//Showing albums of the artist***

***echo "Albums: <br>";***

***$albums=mysqli\_query($con,"SELECT album\_name FROM album,has\_album,artist\_info WHERE artist\_info.id=has\_album.artist\_id AND has\_album.album\_id=album.album\_id AND artist\_info.name='$selected\_artist'") or die(mysqli\_error($con));***

***if ($albums->num\_rows > 0) {***

***while($row = $albums->fetch\_assoc()) {***

***?>***

***<a href="Album.php?name=<?php echo $row['album\_name']; ?>"><?php echo $row['album\_name']; ?></a><br>***

***<?php***

***}***

***}***

***//Album.php***

***//Showing the songs and directing to Youtube***

***$songs=mysqli\_query($con,"SELECT song\_name FROM song,has\_song,album WHERE album.album\_id=has\_song.album\_id AND has\_song.song\_id=song.song\_id AND album.album\_name='$selected\_album'") or die(mysqli\_error($con));***

***echo "Songs: <br>";***

***?>***

***</div>***

***<?php***

***if ($songs->num\_rows > 0) {***

***while($row = $songs->fetch\_assoc()) {***

***?>***

***<a class="two" href="https://www.youtube.com/results?search\_query=<?php echo $selected\_album; echo " "; echo $row['song\_name']; ?>" target="\_blank"><?php echo $row['song\_name']; ?></a><br>***

***<?php***

***}***

***}***

Under albums, there is either a “Like” button (if you haven’t liked the artist yet), or a “Remove From Library” button (if you have liked the artist and don’t want it to be in your library anymore.). If you can see the “Remove From Library” button, top of it, there is also a text saying “You have already liked this artist”. If you click any of these 2 buttons, you will be directed to a page that will tell you either you have added or removed an artist to/from your library (Figure 6.1-Figure 6.2). Here’s how it is:

***//Artist.php***

***$result=mysqli\_query($con,"SELECT artist\_name FROM account\_libary,account\_info where account\_libary.user\_id=account\_info.id AND account\_info.user\_name='$user' AND account\_libary.artist\_name='$rowname[0]'") or die(mysqli\_error($con));***

***$row=mysqli\_fetch\_array($result);***

***if($row[0]==$rowname[0]){***

***echo '<div class="already"><br>You have already liked this artist.<br></div>';***

***?>***

***<form name="dislike" method="post">***

***<div class="dislike">***

***<input class="dislikebutton" type="submit" name="dislikebutton" value="Remove From Libary"/>***

***</div>***

***</form>***

***<?php***

***if (isset($\_POST['dislikebutton'])){***

***$result2=mysqli\_query($con,"DELETE FROM account\_libary WHERE user\_id='$row1[0]' AND account\_libary.artist\_name='$selected\_artist'") or die(mysqli\_error($con));***

***header("Location:Disliked.php");***

***}***

***}else{***

***?>***

***<form name="like" method="post">***

***<div class="like">***

***<input class="likebutton" type="submit" name="likebutton" value="Like"/>***

***</div>***

***</form>***

***<?php***

***if (isset($\_POST['likebutton'])){***

***$artistid=mysqli\_query($con,"SELECT id FROM artist\_info where artist\_info.name='$rowname[0]'") or die(mysqli\_error($con));***

***$row2=mysqli\_fetch\_array($artistid);***

***$result2=mysqli\_query($con,"INSERT INTO account\_libary(user\_id,artist\_id,artist\_name) VALUES('$row1[0]','$row2[0]','$rowname[0]')") or die(mysqli\_error($con));***

***header("Location:Liked.php");***

***}***

***}***

Lastly, there 3 other buttons under the button. “My Recommendations”, “Dashboard” and “My Library”. These buttons work exactly the same as in the other pages.

So, since we have liked some artists, we can check our library (Figure 7). The way his page works very similar to the Dashboard. Prints out “hello” with the username, lists the liked artists and gives link to the other pages. But different from the Dashboard, it puts the artist that you like. The way I implemented it is like this:

***//Libary.php***

***$userid=mysqli\_query($con,"SELECT id FROM account\_info where account\_info.user\_name='$temp'") or die(mysqli\_error($con));***

***$row1=mysqli\_fetch\_array($userid);***

***$result1=mysqli\_query($con,"SELECT artist\_name FROM account\_libary WHERE account\_libary.user\_id='$row1[0]' GROUP BY artist\_name") or die(mysqli\_error($con));***

***if ($result1->num\_rows > 0) {***

***while($row = $result1->fetch\_assoc()) {***

***?>***

***<a class="one" href="Artist.php?name=<?php echo $row['artist\_name']; ?>"><?php echo $row['artist\_name']; ?></a><br>***

***<?php***

***}***

***}***

So, it’s time to talk about the most important part of RecoMuse; the recommendation system. The recommendation system based on your likes, and the likes of other people that have liked the same artists with you. To do this, I divided the objective into 3 parts. First, the algorithm searches through the ids of the artists that you’ve liked. Second, it searches through these artist ids to find the ids of the users that liked any of these artists. Third, it searches through these user ids to find the artists that they’ve liked. The recommendation page’s design is same with the library’s design (Figure 8), except it’s content is different. The way I implemented the recommendation system is like this:

***//Recommend.php***

***$temp=$\_SESSION['user'];***

***$userid=mysqli\_query($con,"SELECT id FROM account\_info where account\_info.user\_name='$temp'") or die(mysqli\_error($con));***

***$row\_userid=mysqli\_fetch\_array($userid);***

***$result1=mysqli\_query($con,"SELECT DISTINCT account\_libary.artist\_name FROM account\_libary,(SELECT DISTINCT account\_libary.user\_id FROM account\_libary,(SELECT account\_libary.artist\_id FROM account\_libary WHERE account\_libary.user\_id='$row\_userid[0]') as a WHERE account\_libary.artist\_id=a.artist\_id) as b WHERE account\_libary.user\_id=b.user\_id ORDER BY `account\_libary`.`artist\_name` ASC") or die(mysqli\_error($con));***

***if ($result1->num\_rows > 0) {***

***while($row = $result1->fetch\_assoc()) {***

***?>***

***<a class=one href="Artist.php?name=<?php echo $row['artist\_name']; ?>"><?php echo $row['artist\_name']; ?></a><br>***

***<?php***

***}***

***}***

***?>***

**CLOSING THOUGHTS**

This project was really helpful for me to understand the basic concepts of creating a website, website design, database usage, and connecting a database with a website. The most challenging things in this project were probably connecting the website with database in the beginning and implementing the recommendation system at the end.

**USE CASE DIAGRAM**

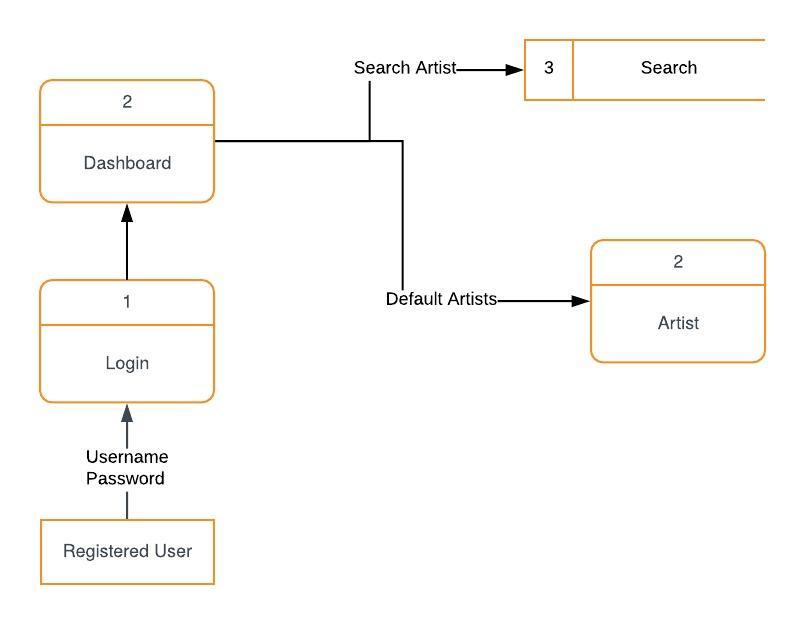
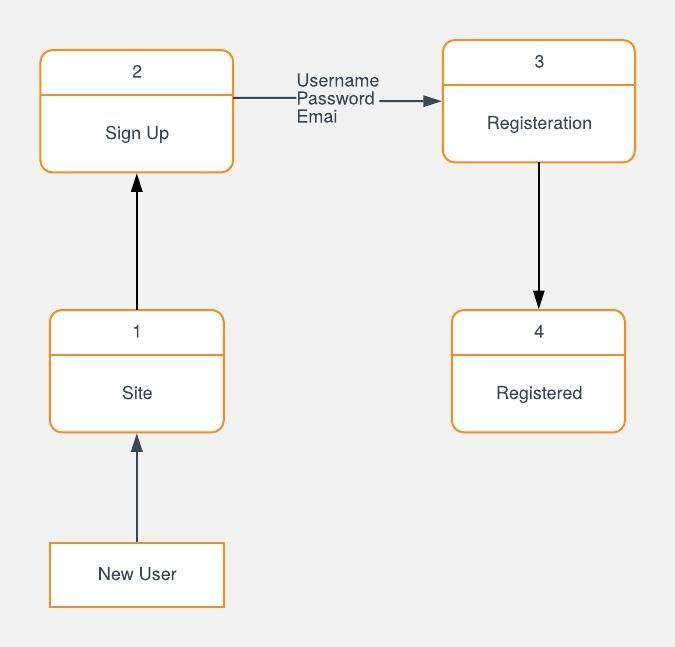
|  |  |  |
| --- | --- | --- |
| Use Case Name: Registeration | ID: UC\_1 | Priority: High |
| Actor: A Person Wants To Sign Up | | |
| Description: The person wants to create an account in order to use the site. | | |
| Trigger: The user wants to explore some new music | | |
| Type: External | | |
| Preconditions:   1. The user must have an email. 2. The user must choose an unique name. 3. The user must determine a password between 8-16 characters. | | |
| Normal Course:   1. The user enters to site. 2. He clickes to the "register" button since he/she doesn't have an account to log in. 3. The user chooses his/her username. 4. The user chooses a password between 8- 16 characters. 5. The user must give his/her email. | Alternative Course: | |
| Exceptions:  \*If the user miss even one of the preconditions, he/she will get an error that telling him/her to fill the missing informations. | | |
| Summary | | |
| Inputs | Outputs | BARIS |
| \*Username | \*Account | ARAN |
| \*Password | 150140223 |
| \*Email |  |

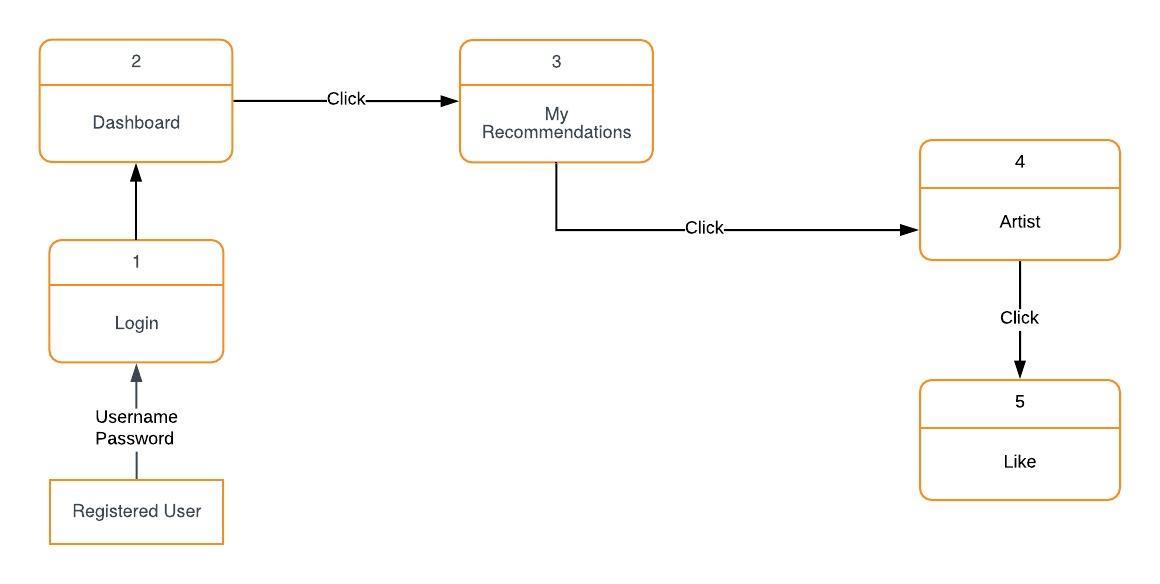
|  |  |  |
| --- | --- | --- |
| Use Case Name: First Login | ID: UC\_2 | Priority: High |
| Actor: A Person That Is Looking For A New Artist To Listen | | |
| Description: The user's first time logging in to the site. | | |
| Trigger: The user wants to explore some new music | | |
| Type: External | | |
| Preconditions:   1. The user must have an account. | | |
| Normal Course:   1. The user logs in. 2. The user checks the default recommendations which are the most liked artists in dashboard. 3. The user can click to an artist to get some information. 4. The user can click "Like" depending on either he/she liked the artist or not. | Alternative Course:   1. The user logs in. 2. If the user is not happy with the default recommendations, he/she can search for his favorite artists and like them in order to get recommendations according to his taste later. | |
| Exceptions:  \*If the user searchs for an artist that doesn't exist in the database, he will recieve a message "There is no such artist" | | |
| Summary | | |
| Inputs | Outputs | BARIS |
| \*Username  \*Password | \*Like | ARAN |
| 150140223 |

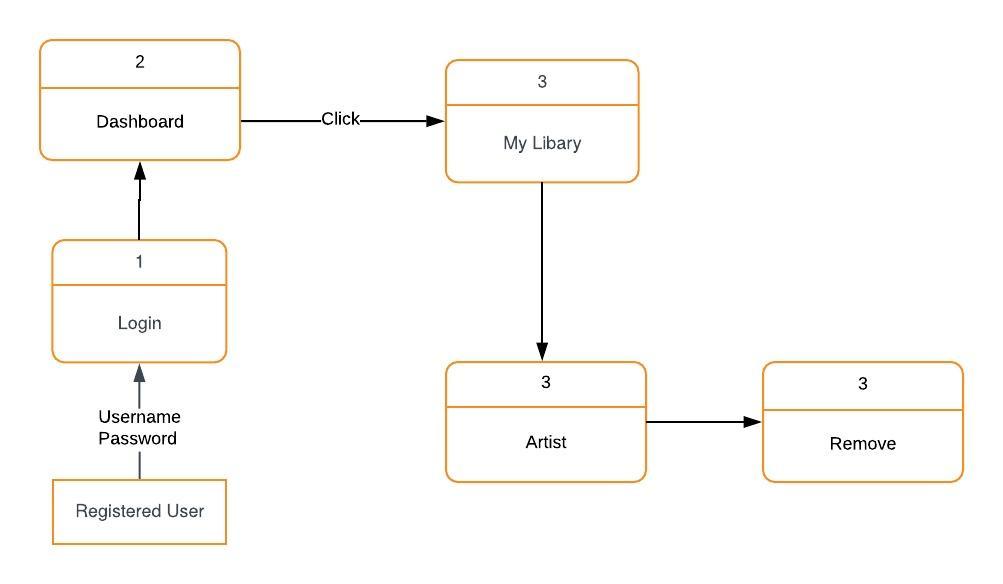
|  |  |  |
| --- | --- | --- |
| Use Case Name: Suggesting A New Artist | ID: UC\_3 | Priority: High |
| Actor: A Person That Is Looking For A New Artist To Listen | | |
| Description: The user gets new artist recommendations according to the artists that he liked before and the other users' preferances. | | |
| Trigger: The user wants to explore some new music | | |
| Type: External | | |
| Preconditions:   1. The user must have an account. 2. The user must be logged in. 3. The user must have liked other artist(s) before. | | |
| Normal Course:   1. The user logs in. 2. The user clicks to "My Recommendations" button. 3. The user can check the recommended artists by clicking on them. 4. In artist profile, the user can click "Like" depending on either he/she liked the artist or not. | Alternative Course: | |
| Exceptions: - | | |
| Summary | | |
| Inputs | Outputs | BARIS |
| \*Username | \*Recommendations | ARAN |
| \*Liked Artists | 150140223 |

|  |  |  |
| --- | --- | --- |
| Use Case Name: Editing Libary | ID: UC\_4 | Priority: High |
| Actor: A Person That Is Not Happy With The Recommendations | | |
| Description: The user somehow liked an artist but is not happy with the recommendations that it is bringing. | | |
| Trigger: The user wants to remove an artist from the liked library. | | |
| Type: External | | |
| Preconditions:   1. The user must have an account. 2. The user must be logged in. 3. The user must have liked other artist(s) before. | | |
| Normal Course:   1. The user logs in. 2. The user clicks to "My Libary" button. 3. The user can click the artist that he/she wants to remove, and then click to “Remove From The Library” button. | Alternative Course: | |
| Exceptions: - | | |
| Summary | | |
| Inputs | Outputs | BARIS |
| \*Username  \*Liked Artist | \*Removal | ARAN |
| 150140223 |

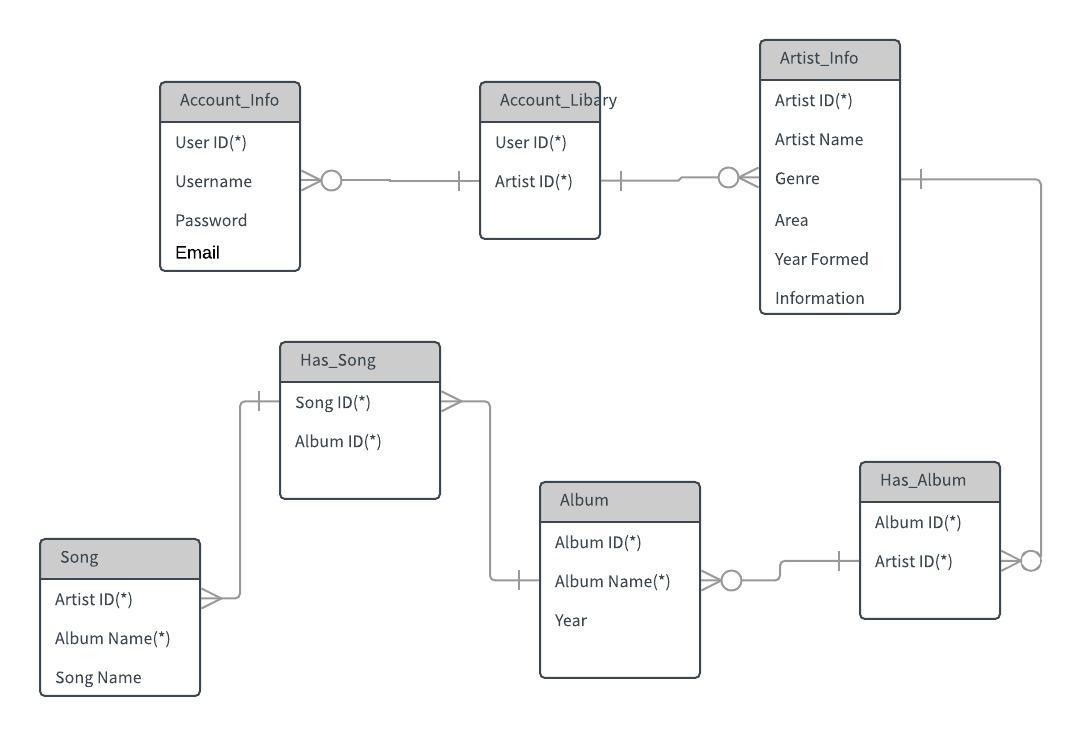
**DATA FLOW DATAGRAM**

****

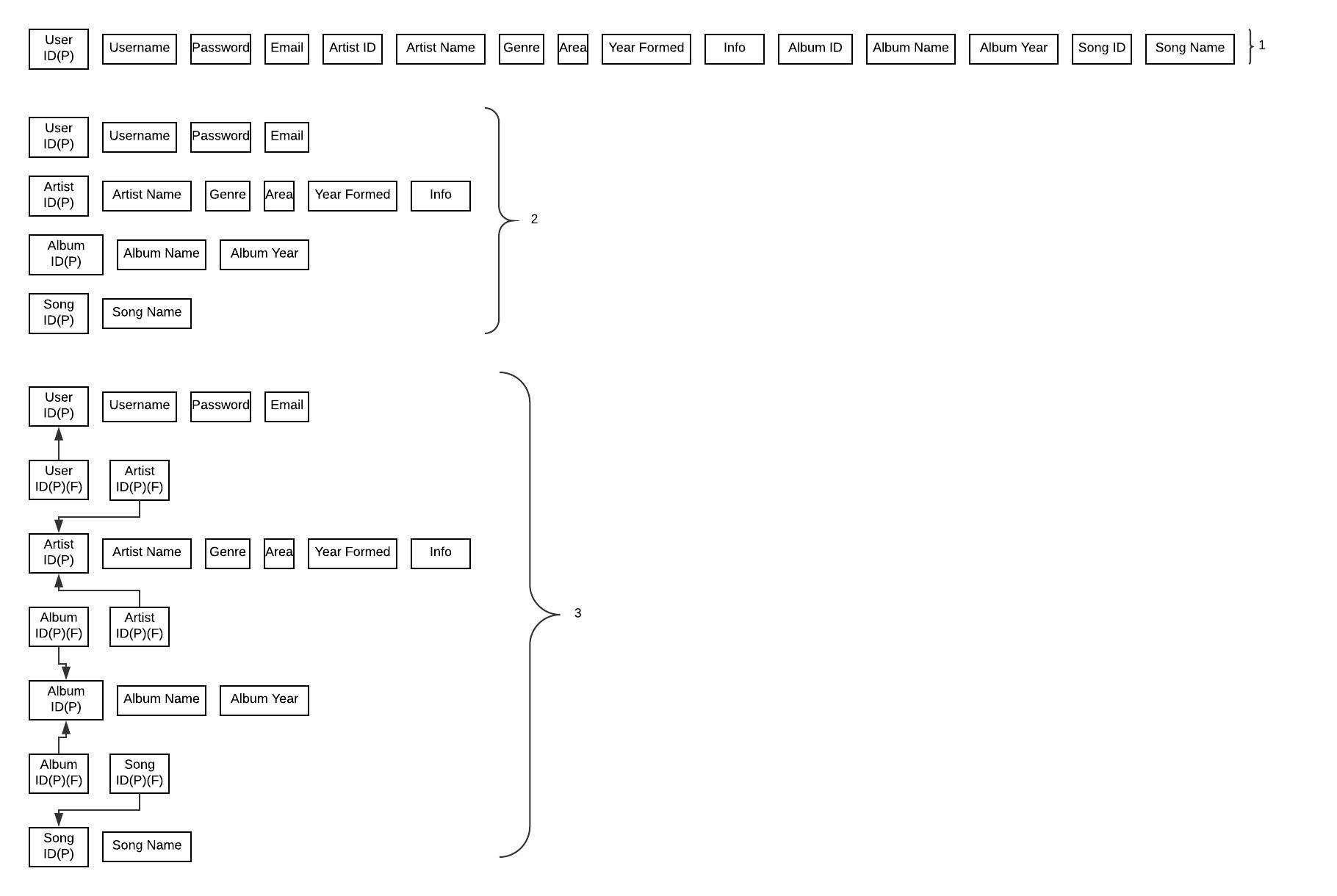
****

****

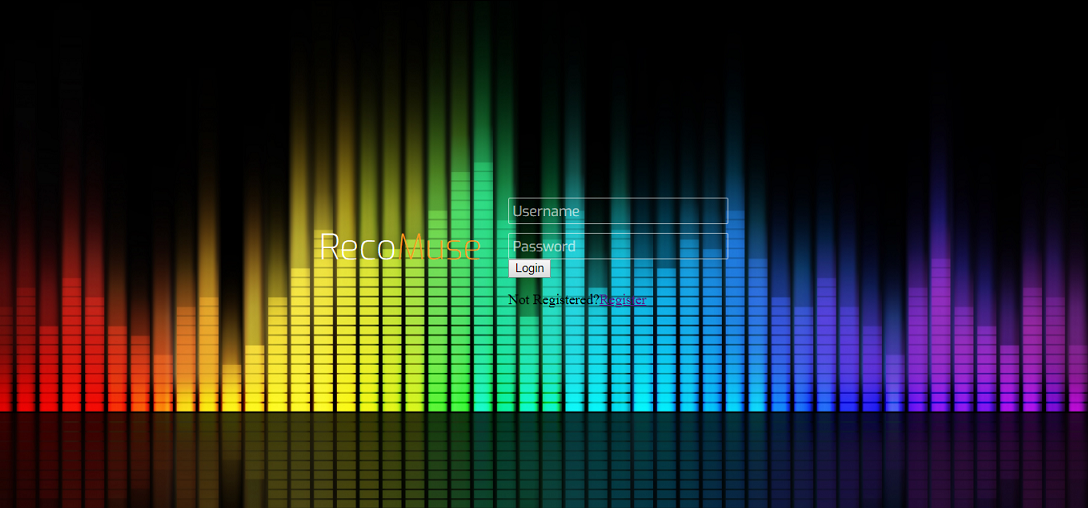
**ENTITY RELATIONSHIP DIAGRAM**

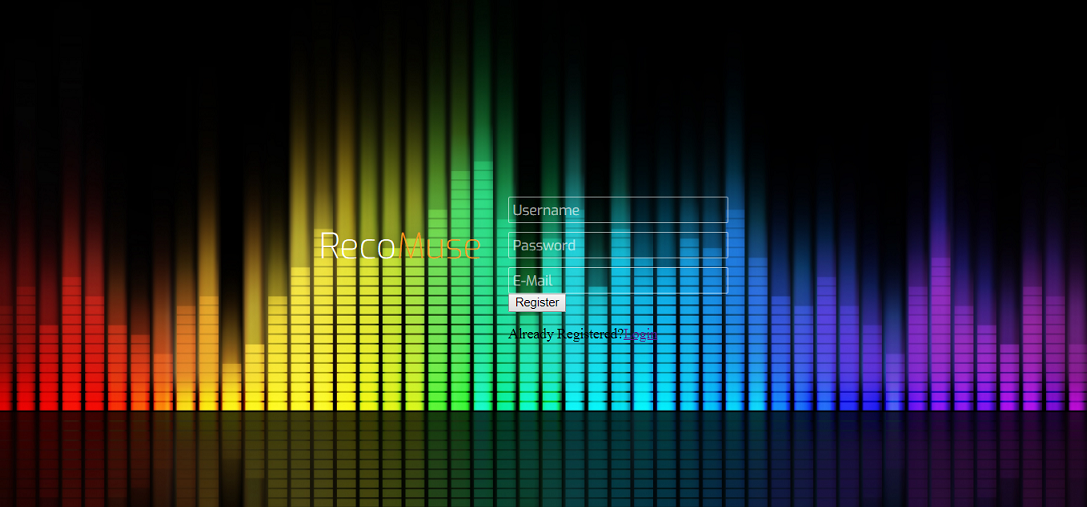
****

**NORMALIZATION**

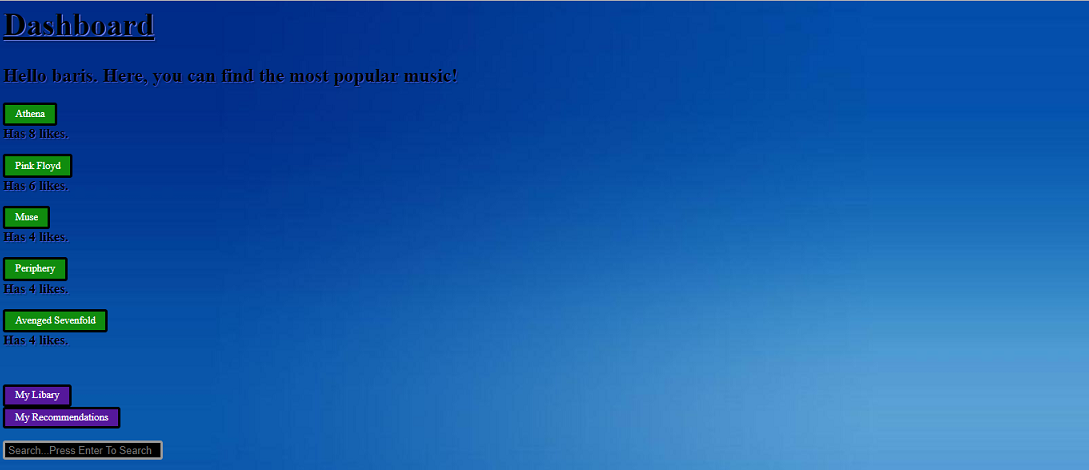
****

**IMAGES**

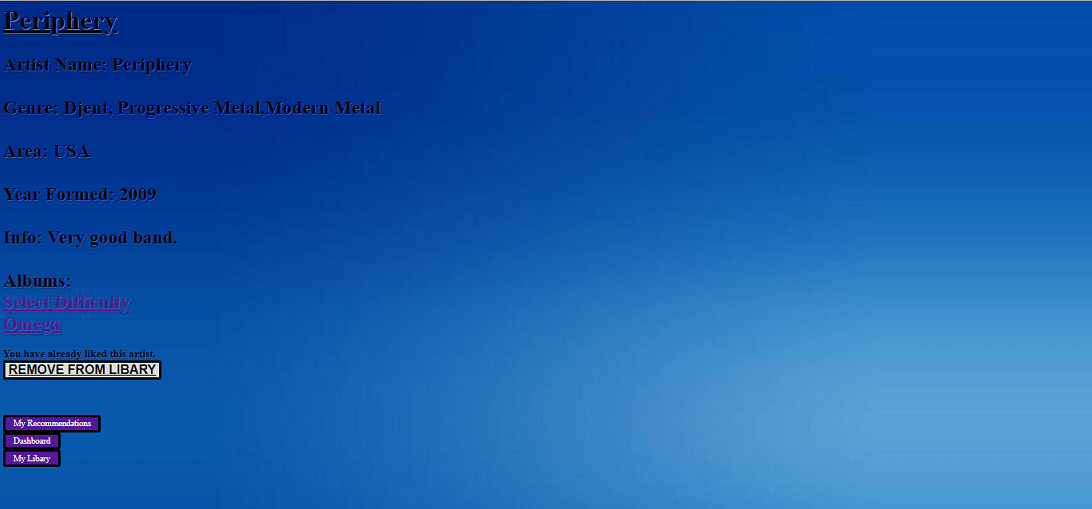
(Figure 1-Login Page)

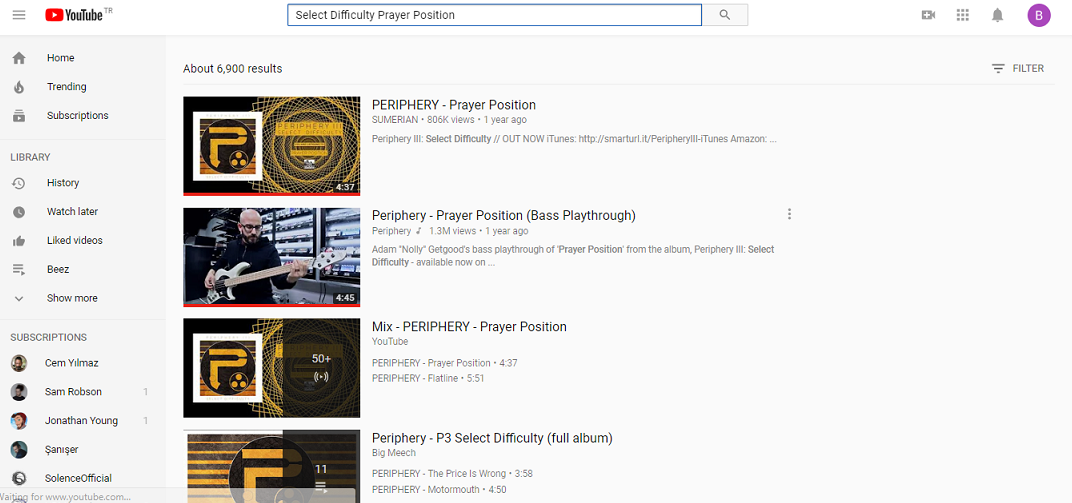
(Figure 2-Register Page)

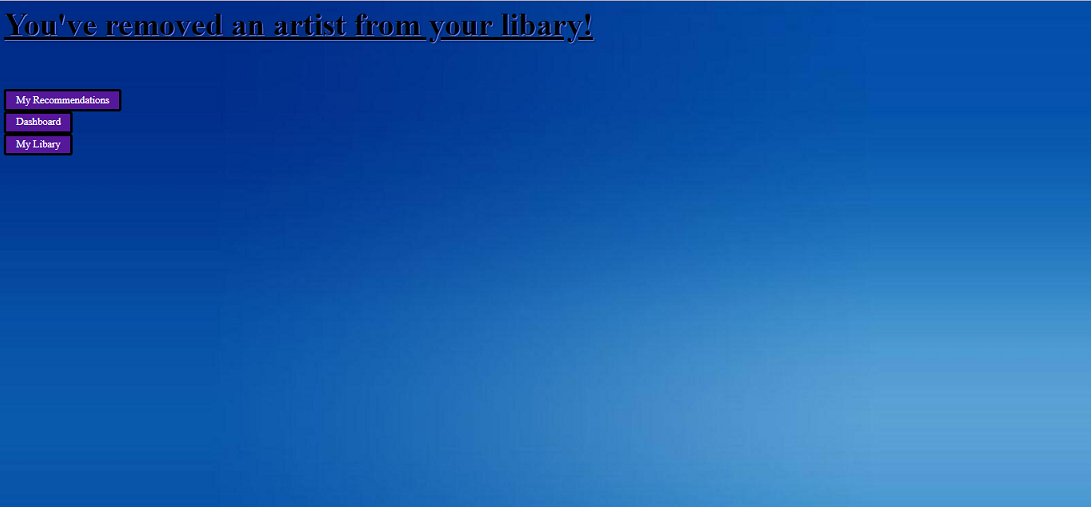
(Figure 3-Register Confirmation)

(Figure 4-Dashboard)

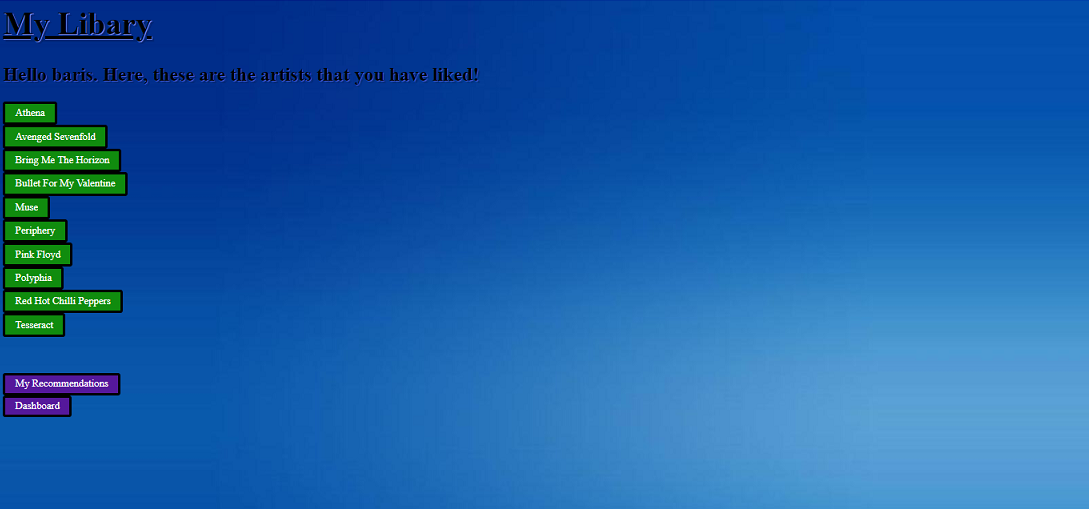
 (Figure 5.1-New Artist Profile)

(Figure 5.2-Liked Artist Profile) (Figure 5.3-Album)

(Figure 5.4-Directing To Youtube)

(Figure 6.1-Removal From Library)

(Figure 6.2-Addition To Library)

(Figure 7-Library)

(Figure 8-Recommendations)