Table of Contents

1.0 Proposal

1.1 Introduction

The music streaming industry has significantly evolved over the past decade, offering instant access to millions of songs, yet many platforms fall short in providing a truly personalized and engaging experience. In this crowded landscape, music streaming often becomes a routine activity, with users struggling to connect deeply due to generic recommendations and limited artist interaction. Therefore, Meloverse seeks to address this gap by prioritizing a user-centric experience that revolves around individual preferences and favorite artists. Unlike traditional services, it enhances the connection between listeners and musicians by offering exclusive content such as behind-the-scenes footage, real-time updates on concerts, new releases, and personalized recommendations. By fostering a space for connection, discovery, and celebration, Meloverse aims to redefine music streaming, setting itself apart as a unique platform where music is not just heard but felt, creating an immersive musical universe for each user.

1.2 Objective of Meloverse

The primary goal of Meloverse is to deliver a music streaming platform that emphasizes personalization, user engagement, and artist-centric features, setting itself apart from traditional services by putting the user experience at the forefront. By allowing users to select their favorite artists and tailoring content around those choices, Meloverse ensures that each interaction feels unique and relevant, transforming the listening experience into a more meaningful journey. With exclusive content such as behind-the-scenes footage, artist interviews, early access to releases, and real-time updates on concerts and merchandise, users can connect more deeply with the musicians they love. Beyond streaming, the platform aims to build a community where users feel valued through personalized recommendations and user-generated playlists, while also providing an inclusive environment for emerging artists to gain exposure. Ultimately, Meloverse seeks to set a new standard in music streaming by continuously evolving its features to elevate the experience, making it a place not only for listening but for discovering, connecting, and experiencing music in a more immersive and meaningful way.

1.3 Logo

Figure 1.1 Meloverse logo

The Meloverse web application follows the theme of starry night. Just as how the galaxy hosts vast numbers of stars, our web application caters towards a vast number of users, we use it to symbolize the vast possibilities we provide in enhancing a user's experience. Therefore, our logo is a music vinyl that depicts the musical aspect of the web application whilst the star signifies the quantity of personalisations features. The color theme focuses on a darker palette that is soothing and easy on the eyes and replicates the colors of a night sky.

1.4 Functionalities of Meloverse

The Meloverse platform caters to two types of users: Regular Users (Listeners) and Administrators (Admins). Each type of user is offered a set of functionalities tailored to their needs.

1.4.1 For Regular Users (Listeners):

1. **User Account Management:**
    a. Users can sign up and log in using their username and password.
    b. Personal profiles allow users to manage preferences, favourite artists, and playlists.
2. **Music Library Access:**
    a. Search, sort, and filter songs, albums, and artists.
    b. Play, pause, skip, and adjust volume with high-quality streaming.
    c. Users can create, modify, and delete custom playlists and mark favourites.
3. **Playback Features:**
    a. Standard playback controls, including shuffle, repeat, and volume adjustments.
    b. Playback queues to view, edit, and rearrange upcoming tracks.

1.4.2 Special and Innovative Features for Regular Users:

1. **Artist News and Updates:**
    a. Real-time updates on favorite artists, including concert dates, new releases, and exclusive merchandise.
    b. Access to exclusive artist content like behind-the-scenes footage and interviews.
2. **Points Reward System:**
    a. Earn points by engaging with the platform (streaming, playlist creation, following artists).
    b. Redeem points for exclusive content, early releases, or artist-related experiences.
3. **Social Interaction:**
    a. Follow artists and users to stay updated on new music and shared playlists.
    b. Leave comments and reviews on songs, albums, and playlists.

4. **Personalized Recommendations:**
    a. Tailored playlists and recommendations based on user preferences.
    b. Curated artist-centric content to enhance user engagement.

1.4.3 For Administrators (Admins):

1. **Content Management:**
    a. Admins can upload songs, manage artist profiles, and curate playlists.
2. **User Management:**
    a. Supervision of user accounts, monitoring engagement, and addressing issues.
3. **Security and Privacy:**
    a. Encryption of user data and compliance with global privacy laws.
4. **Analytics and Reporting:**
    a. Track song popularity, user activity, and engagement metrics.

1.5 Differentiating Features of Meloverse

What truly distinguishes Meloverse from other music streaming platforms is its unwavering commitment to creating a personalized, artist-focused user experience. While many platforms provide vast libraries of songs, they often overlook the potential to make the listening experience more meaningful and relevant to each individual listener. Meloverse addresses this gap by centering its platform around users' favorite artists, delivering content that is meticulously curated to resonate with their preferences.

At its core, Meloverse regularly updates users with artist-specific news, merchandise offers, and concert information, allowing them to stay closely connected with their favorite musicians. This real-time artist interaction fosters a level of engagement and loyalty that other platforms frequently neglect, transforming passive listening into an interactive relationship between fans and artists. Users are not just passive consumers; they become active participants in their musical journey, engaging with the artists they admire in meaningful ways.

Furthermore, the points reward system amplifies the user experience by introducing a fun, gamified element to the platform. Users can collect points through regular engagement—whether by listening to music, sharing playlists, or interacting with content—and redeem these points for exclusive perks, such as early access to concert tickets, unique merchandise, or exclusive behind-the-scenes content. This feature transcends the traditional model of simply offering music; it ensures that users feel valued and involved in the vibrant musical universe they cherish, fostering a deeper sense of community and connection. Ultimately, Meloverse transforms music streaming into an engaging experience that celebrates both the artistry of musicians and the passion of their fans.

1.6 Database Tables

1) Users Table

User_ID	Username	Password	Email	Date_Joined	Points_Balance
U0001	Taylorian	iovewebap	taylorian@gmail.com	17-10-2024	10

Description: Represents the registered users of Meloverse. It stores information about the user, such as login details, profile information, and points balance.

2) Artists Table

Artist_ID	Artist_Name	Genre	Bio	Picture
A0001	Ed Sheeran	Pop	……	…….
A0002	Sabrina Carpenter	Pop	……	……
A0003	Taylor Swift	Country Pop	……	……

Description: Represents musical artists whose songs and merchandise are available on the platform.

3) Songs Table

Song_ID	Title	Artist_ID	Album_ID	Release_Date	Genre	Duration	File_Path
S0001	Photograph	A0001	B0001	11 May 2015	Folk-Pop	4:19	……
S0002	Perfect	A0001	B0002	26 September 2017	Pop	4:23	……
S0003	Love Story	A0003	B0003	15 September 2008	Country Pop	3:55	……

Description: Contains details about the individual songs available for streaming on Meloverse.

4) Albums Table

Album_ID	Album_Name	Artist_ID	Release_Date	Genre	Cover_Image
B0001	X	A0001	20 June 2014	Pop	……
B0002	÷	A0001	3 March 2017	Pop	……
B0003	Fearless	A0003	11 November 2018	Country Pop	……

Description: Represents music albums that consist of multiple songs.

5) Playlists Table

Playlist_ID	User_ID	Playlist_Name	Date_Created
P0001	U0001	Study Playlist	17-10-2024

Description: Represents playlists created by users.

6) Favourites Artist Table

Favourite_A_ID	User_ID	Artist_ID	Date_Added
FA0001	U0001	A0001	17-10-2024
FA0002	U0001	A0003	17-10-2024

Description: Store the users' favourite artists.

7) Favourites Songs Table

Favourite_S_ID	User_ID	Song_ID	Date_Added
FS0001	U0001	S0001	17-10-2024
FS0002	U0001	S0002	17-10-2024

Description: Store the users' favourite songs.

8) Points Table

Points_ID	User_ID	Points_Earned	Points_Spent	Date_Updated
P0001	U0001	10	-	17-10-2024

Description: Tracks points earned and spent by users for rewards and engagement activities.

9) Merchandise Table

Merch_ID	Artist_ID	Item_Name	Description	Price	Stock_Quantity	Image_Path
M0001	A0001	Blue Monster Hot Water Bottle	……	RM 140	15	……
M0002	A0002	Espresso Martini Crewneck	……	Rm 293	49	……
M0003	A0003	TTPD Black Embossed Hoodie	……	RM 323	9	……

Description: Represents merchandise items related to artists, such as clothing, accessories, or collectibles.

10) Login Table

Login_ID	User_ID	Login_Time	Logout_Time
L0001	U0001	10:15 AM	12:15 PM

Description: Records details of user logins, including timestamps for security and monitoring.

11) Register Table

Register_ID	User_ID	Username	Password	Email	Date_Joined
R0001	U0001	Taylorian	ilovewebap	taylorian@gmail.com	17-10-2024

Description: Manage information regarding user registrations, storing details necessary for login and account management.

12) Admin Table

Admin_ID	Admin_Username	Admin_Password
A0001	iamtheadmin	ilovetaylor

Description: Store information about the platform administrators who manage content, user accounts, and system settings.

1.7 Entity-Relationship Diagram

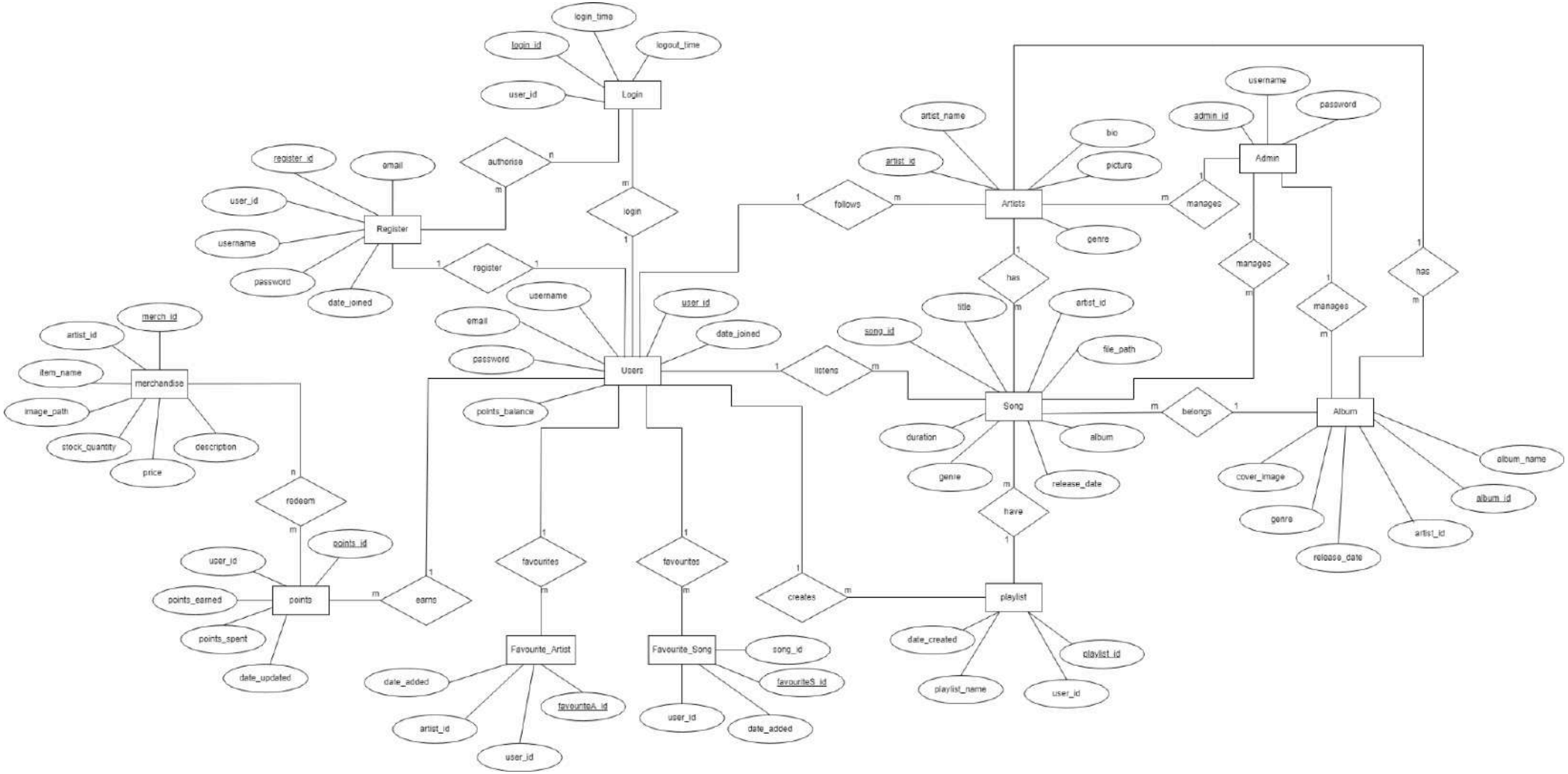


Figure 1.2. Meloverse ER Diagram

1.7.1 ER Diagram Explanation

This ER diagram represents a music streaming platform where users can interact with songs, artists, playlists, and merchandise. Users register, log in, and can earn and redeem points for engaging with the platform. They can favorite songs and artists, create playlists, and follow artists. Artists are managed by admins, release songs and albums, and offer merchandise for users to purchase using points. Songs are organized into albums and playlists, while the platform tracks user activities such as listening to songs and earning points. Admins manage artist and album content, ensuring the platform remains well-curated.

1.7.2 Entities and Their Relationships

1. Users

Attributes: user_id, username, email, password, date_joined, points_balance.

Description: The "Users" table stores information about the users who are registered on the Meloverse platform. It tracks their account details, including username, email, password (securely stored as a hash), the date they joined, and the current balance of reward points.

Relationships:

- ➔ **One-to-Many with Playlists**: Each user can create multiple playlists.
- ➔ **One-to-Many with Points**: Users can earn and spend points, recorded in the Points table.
- ➔ **One-to-Many with Login**: Each user can have multiple login records.
- ➔ **One-to-Many with Favourite Songs and Favourite Artists**: Users can have multiple favorite songs and artists.
- ➔ **One-to-Many with Register**: Each user has one corresponding registration record.

2. Artists

Attributes: artist_id, artist_name, genre, bio, picture.

Description: The "Artists" table holds details about artists whose songs are available on Meloverse, including their name, genre, biography, and picture.

Relationships:

- ➔ **One-to-Many with Songs**: Each artist can have multiple songs associated with them.
- ➔ **One-to-Many with Albums**: An artist can have multiple albums.

3. Songs

Attributes: song_id, title, artist_id, album, release_date, genre, duration, file_path.

Description: The "Songs" table contains information about each song, including the title, artist, album, release date, genre, duration, and the file path for the audio file.

Relationships:

- ➔ **Many-to-One with Artists**: Each song is performed by a specific artist.
- ➔ **Many-to-One with Albums**: Songs can belong to an album.

4. Albums

Attributes: album_id, album_name, artist_id, release_date, genre, cover_image.

Description: The "Albums" table stores details about music albums, including the album name, the associated artist, release date, genre, and cover image.

Relationships:

- ➔ **Many-to-One with Artists**: Each album is created by a specific artist.
- ➔ **One-to-Many with Songs**: Albums contain multiple songs.

5. Playlists

Attributes: playlist_id, user_id, playlist_name, date_created.

Description: The "Playlists" table manages user-created playlists, containing details such as the playlist name, the user who created it, and the date it was created.

Relationships:

- ➔ **Many-to-One with Users**: Each playlist belongs to a specific user.
- ➔ **One-to-Many with Songs:** Playlists contain multiple songs.

6. Points

Attributes: points_id, user_id, points_earned, points_spent, date_updated.

Description: The "Points" table tracks the accumulation and spending of points by users, used for the rewards system in Meloverse.

Relationships:

- ➔ **Many-to-One with Users**: Each record is associated with a specific user.

➔ **Many-to-Many with Merchandise:** Many points can redeem many merch.

7. Merchandise

Attributes: merch_id, artist_id, item_name, description, price, stock_quantity, image_path.

Description: The "Merchandise" table stores details about artist-related merchandise, including item name, description, price, stock, and image.

Relationships:

➔ **Many-to-Many with Points:** Many points can redeem many merch.

8. Login

Attributes: login_id, user_id, login_time, logout_time.

Description: The "Login" table records login activities of users, capturing login and logout times.

Relationships:

➔ **Many-to-One with Users**: Each login record is linked to a specific user.

9. Register

Attributes: register_id, user_id, username, password, email, date_joined.

Description: The "Register" table manages registration details for users, capturing their initial registration information.

Relationships:

➔ **One-to-One with Users**: Each user has one registration record.

10. Favourite Songs

Attributes: favouriteS_id, song_id, user_id, date_added.

Description: The "Favourite Songs" table links users to the songs they have marked as favorites.

Relationships:

- ➔ **Many-to-One with Users**: Each favorite record is linked to a user.
- ➔ **Many-to-One with Songs**: Each favorite record is linked to a song.

11. Favourite Artists

Attributes: favouriteA_id, artist_id, user_id, date_added.

Description: The "Favourite Artists" table links users to the artists they have marked as favorites.

Relationships:

- ➔ **Many-to-One with Users**: Each favorite record is linked to a user.
- ➔ **Many-to-One with Artists**: Each favorite record is linked to an artist.

12. Admin

Attributes: admin_id, username, password.

Description: The "Admin" table stores information about the administrators who manage the Meloverse platform.

Relationships:

- ➔ **One-to-Many with Artists:** One admin manages many artists.
- ➔ **One-to-Many with Albums:** One admin manages many albums.
- ➔ **One-to-Many with Songs:** One admin manages many songs.

1.8 Conclusion

In conclusion, Meloverse is a personalized music streaming platform that puts the user at the center of the experience. With features focused on artist updates, exclusive content, and a reward system, Meloverse offers more than just a place to stream music—it creates a personalized musical journey that reflects each user's preferences. This emphasis on personalization and user engagement sets Meloverse apart from other platforms, providing listeners with a deeper connection to their favorite artists and a richer, more immersive music experience. Through its innovative features and user-focused approach, Meloverse stands poised to redefine the music streaming landscape.

1.9 Prototype

1.9.1 Landing page

The landing page is where users will be directed to when they open our web application. Here, the design is simplistic to capture the user's attention on the main feature of our web app - a user friendly platform catering towards musical artist enthusiasts.

Figure 1.3. Landing page prototype

1.9.2 Home page

Here the users will have immediate access to artists, playlist, songs, and other pages via the top menu.

Figure 1.4. Home page prototype

1.9.3 Discovery page

This page allows for our users to discover new songs that may be of their interest either by showing similar genres of songs they've listened to, or currently trending music. By clicking on an artist's name, a pop-up with information about the artist will appear.

Figure 1.5. Discovery page prototype

1.9.4 Points page

This page displays the current total points garnered by the user as well as possible ways to earn more.

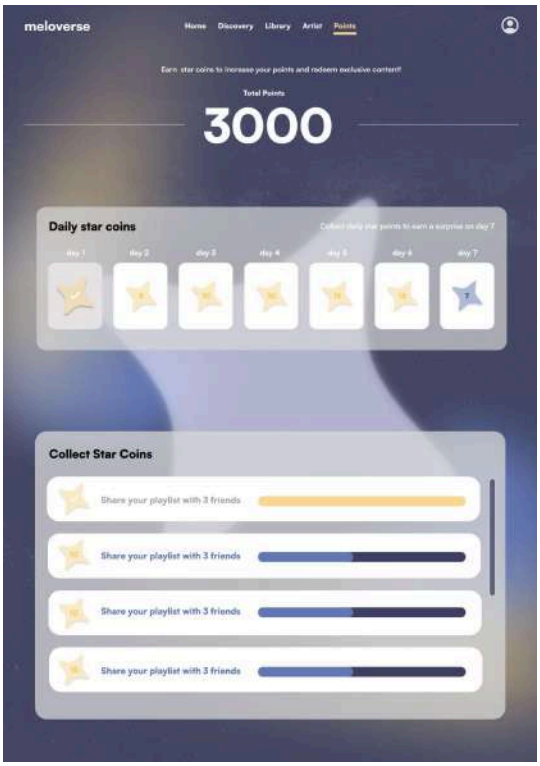


Figure 1.6. Points page prototype

2.0 Introduction - Company Website and Background

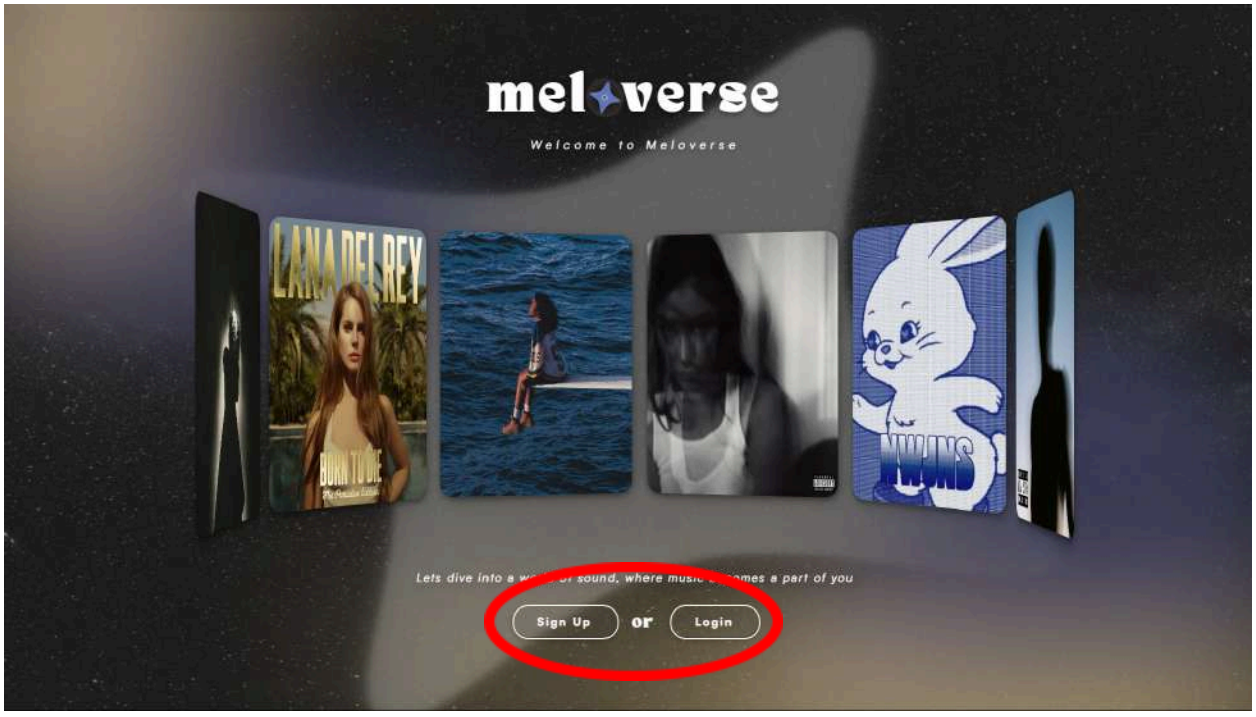
In a world where music is more than just sound, it is a way to connect, express, and inspire, Meloverse emerges as a transformative music streaming application. Designed with a user-first approach, Meloverse goes beyond traditional music platforms by blending personalization, engagement, and an immersive connection between listeners and their favorite artists. With its thoughtfully crafted features, Meloverse ensures that music isn't just heard—it's experienced.

One of the standout features of Meloverse is the Points Page, a rewards-driven system where users can earn points through their interactions on the platform and redeem them for amazing rewards. This gamified approach makes engaging with music even more exciting and rewarding. Additionally, the Artist Page provides a beautifully designed space where users can explore an artist's world. From their popular songs and albums to exclusive merchandise, this page offers a comprehensive introduction to the musicians users love.

These features, combined with a personalized experience that tailors recommendations based on individual preferences, set Meloverse apart from other platforms. By offering real-time updates on concerts and new releases, behind-the-scenes content, and exclusive artist interactions, Meloverse strengthens the bond between users and their favorite musicians. Whether you're discovering new music or deepening your connection with familiar artists, every interaction on Meloverse is crafted to feel meaningful and engaging.

With its innovative features and intuitive design, Meloverse isn't just a music streaming platform—it's a dynamic space where music becomes a lifestyle. It's a platform that rewards passion, celebrates artistry, and redefines what it means to connect with music in the digital age. Welcome to Meloverse, where every song, every artist, and every moment matters.

3.0 User Guide



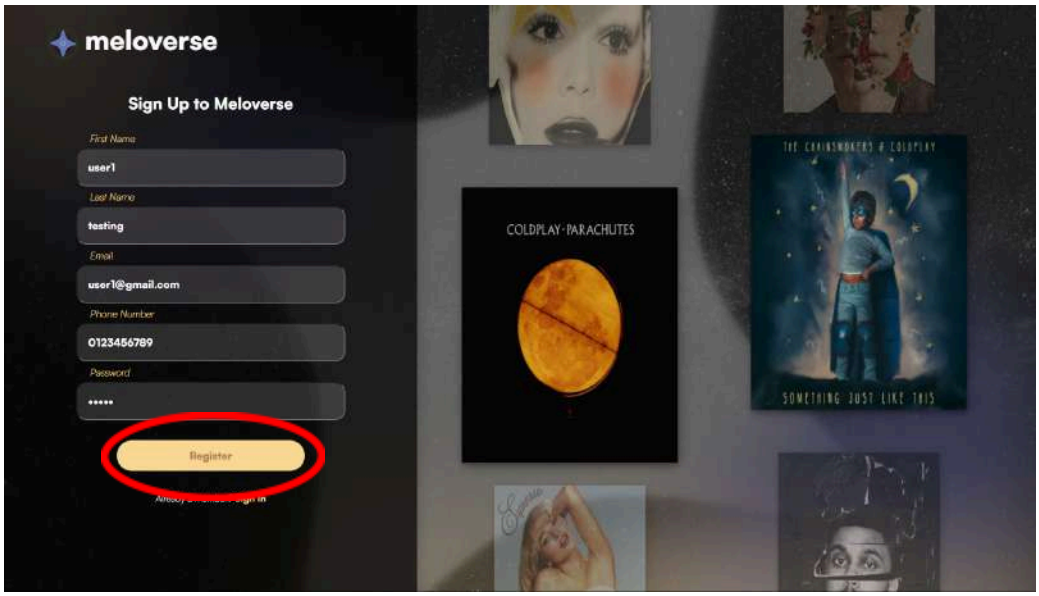
When you first visit the website, you will land on our Landing Page with two buttons: **Sign Up** and **Login**.

If You Don't Have an Account: Click the **Sign Up** button to create a new account.

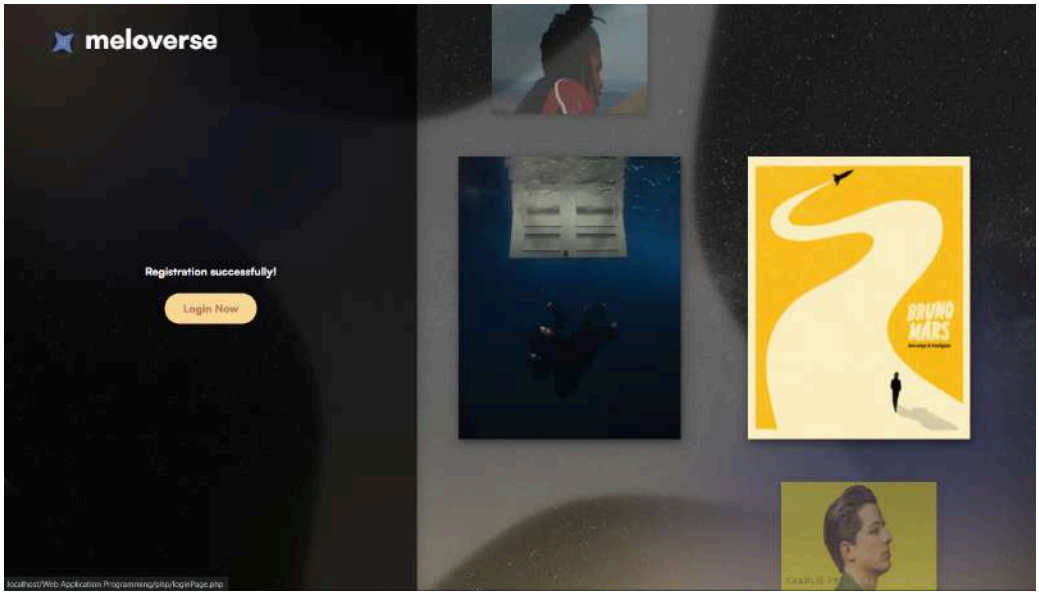
If You Already Have an Account: Click the **Login** button to access your account.

For New Users:

If you don't have an account, click the Sign Up button. You will then be redirected to the Sign Up page. Then, you have to fill in the required information to create an account and access our website. Once you have filled in the information, click the Register button.

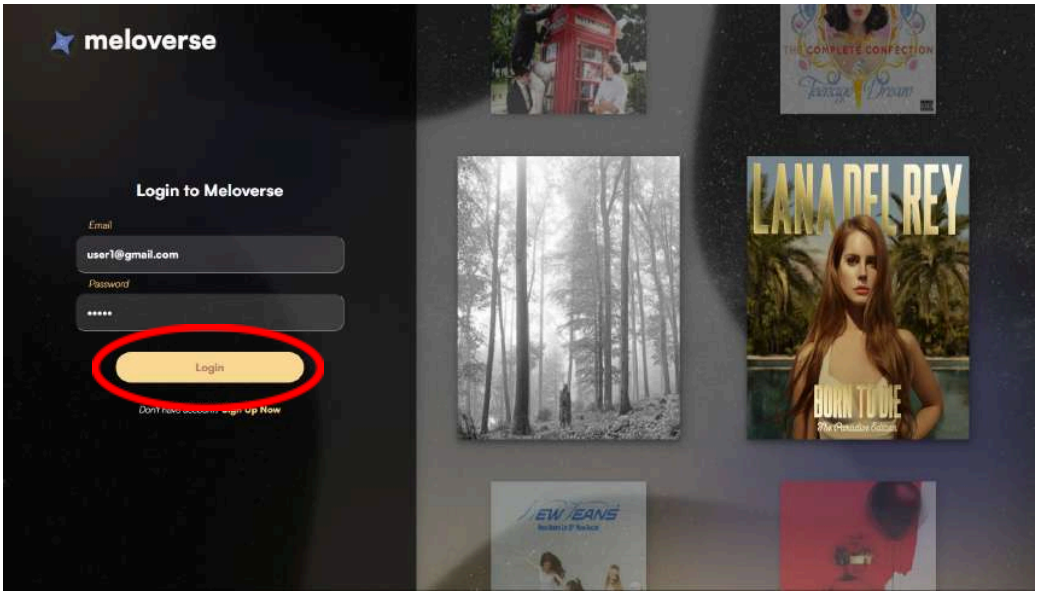


After clicking the register button, a message (Registration successfully!) will pop up, and you are ready to go.

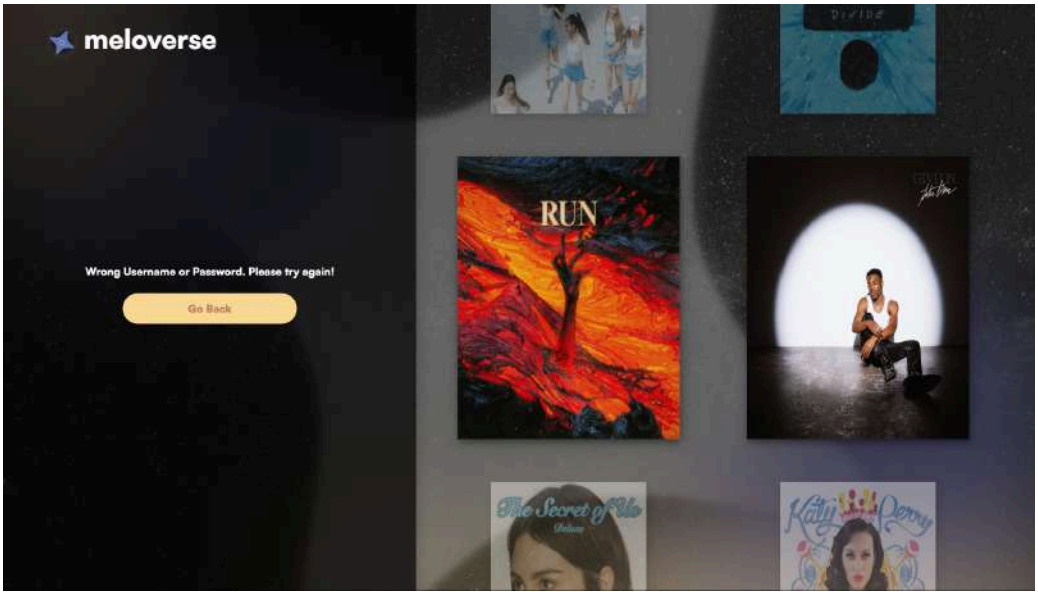


For Existing Users:

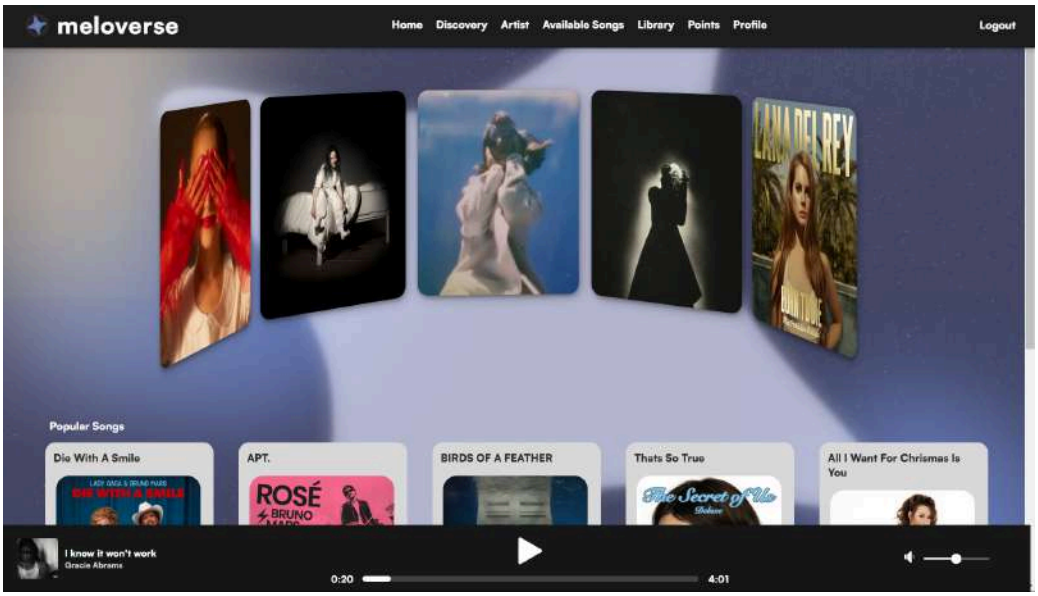
If you click 'Login,' you will be redirected to our login page. On this page, you will need to fill in your account credentials and, once done, click **'Login'** and you will be successfully logged in. However, if you do not already have an account, click **'Sign Up Now'** to be redirected to the Sign Up page.



If you enter the wrong information, you will see an error message: **'Wrong Username or Password. Please try again!'** and will need to re-enter your credentials.



After logging in, you will arrive at the Meloverse home page. On the home page, you can view two sections: our selection of **popular songs** and **popular artists**. At the top, you will see a header section, and at the bottom, a footer section. Our header and footer will be present at every page.



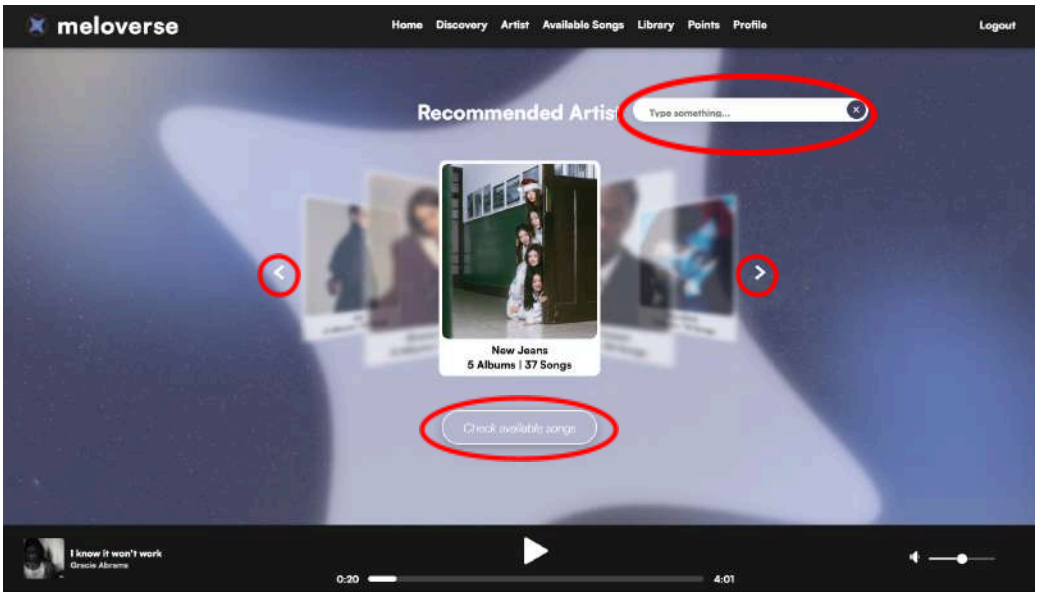
Header Section:

From the header section, you can access different pages such as **Home**, **Discovery**, **Artists**, **Library**, **Available Songs**, **Points**, and **Profile** by clicking on the titles.

Footer Section:

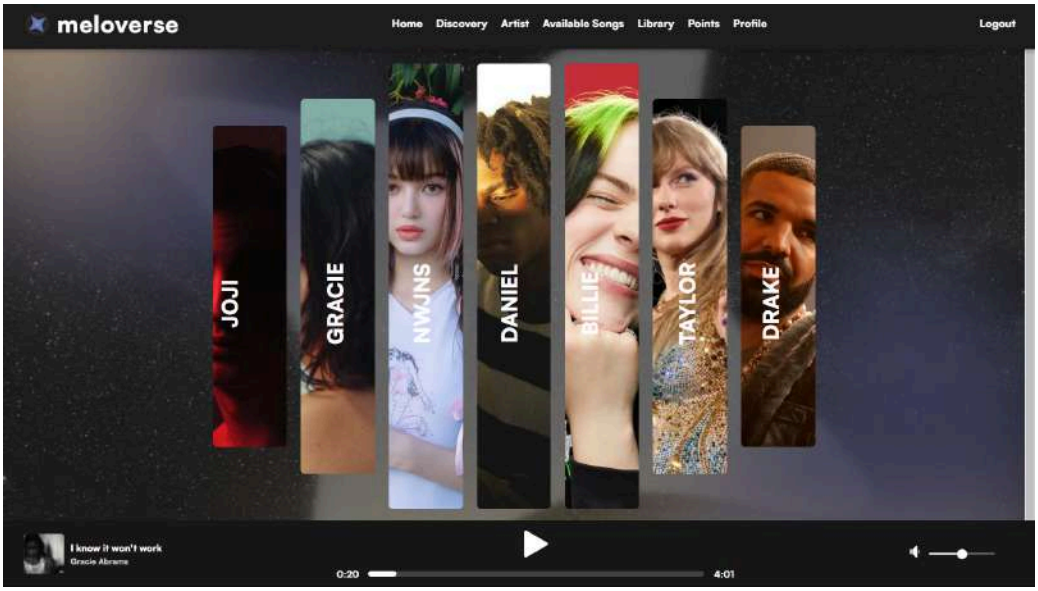
From the footer section, you can access the song controller, which includes features such as the pause/play button, volume slider, and progress bar (allowing you to click and adjust the song's duration).

By clicking **'Discovery'** in the header, you will be redirected to the Discovery Page. There, you can browse our recommended artists and search for your favorite artist using the **search bar**. To scroll through our Recommended Artists, we have **arrow buttons** on either side.

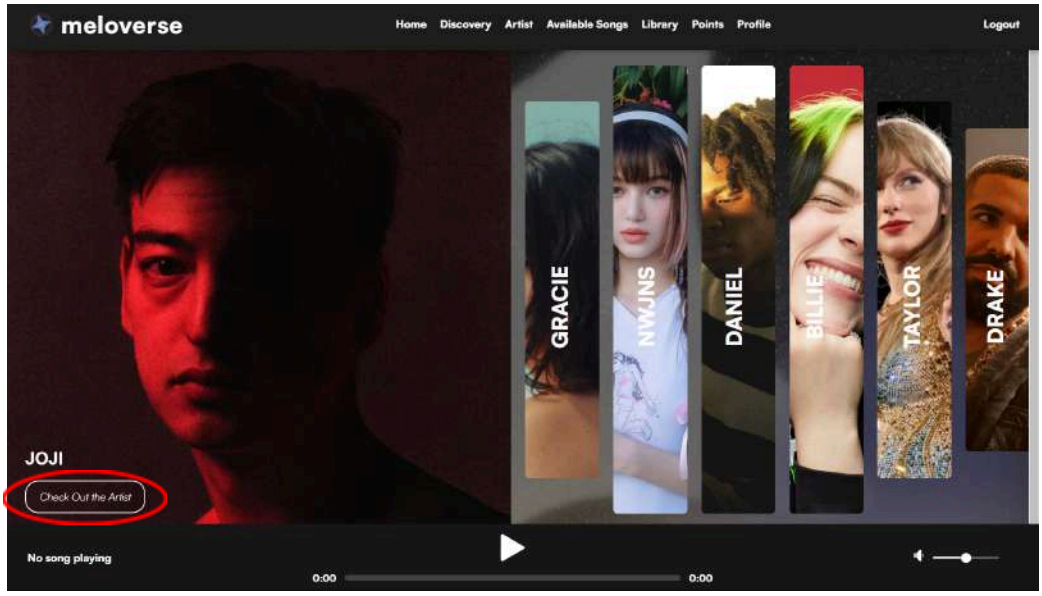


To check out the available songs for the current artist, click the **'Check Available Songs'** button. You will be directed to our Available Songs List page.

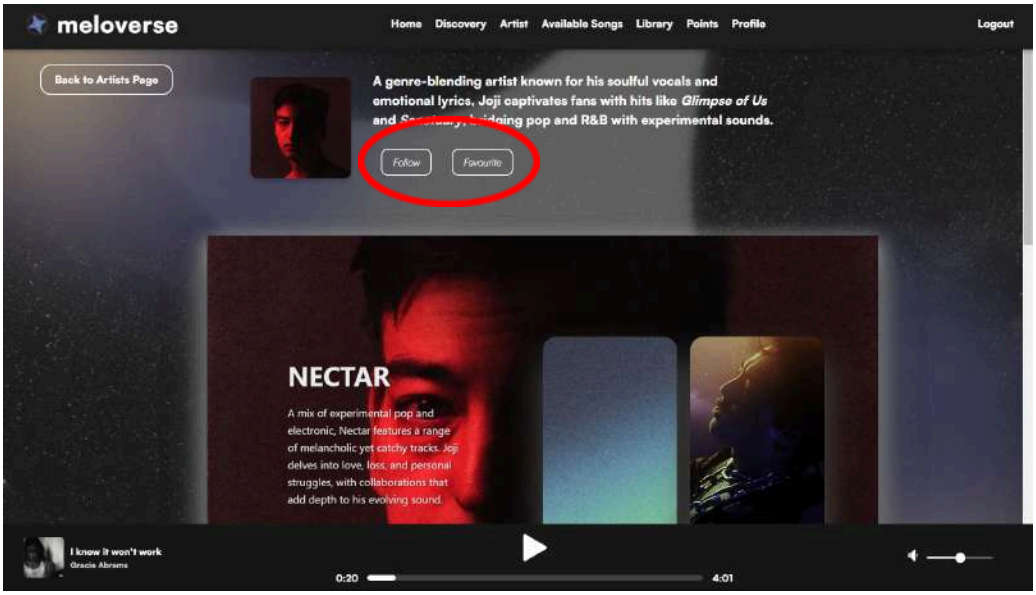
By clicking **'Artist'** in the header, you will be redirected to the Artist Page. Here you can view information of your favourite artist by clicking on their icons.



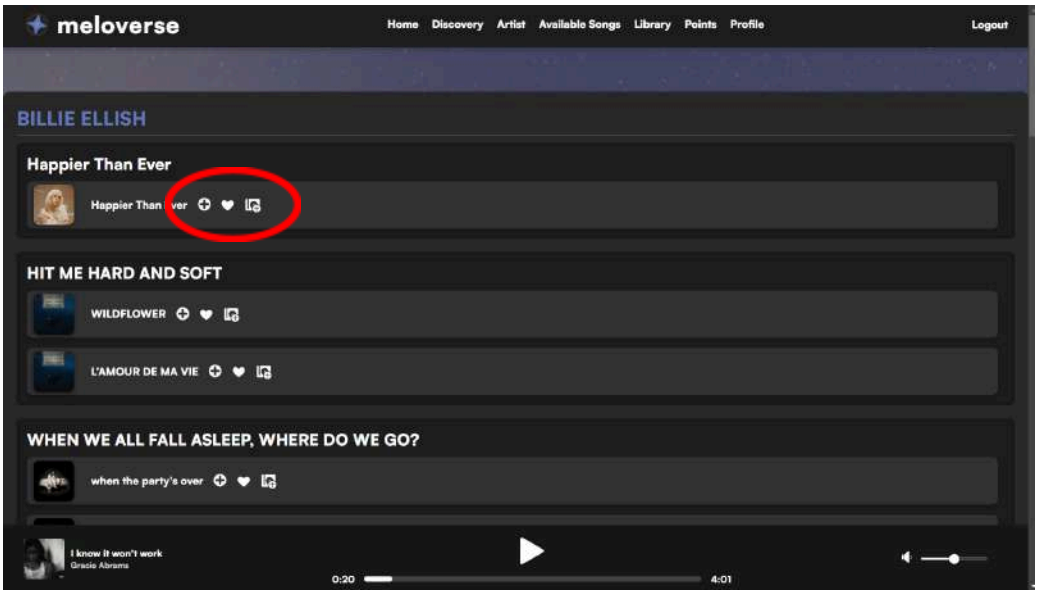
For example, by clicking Joji.



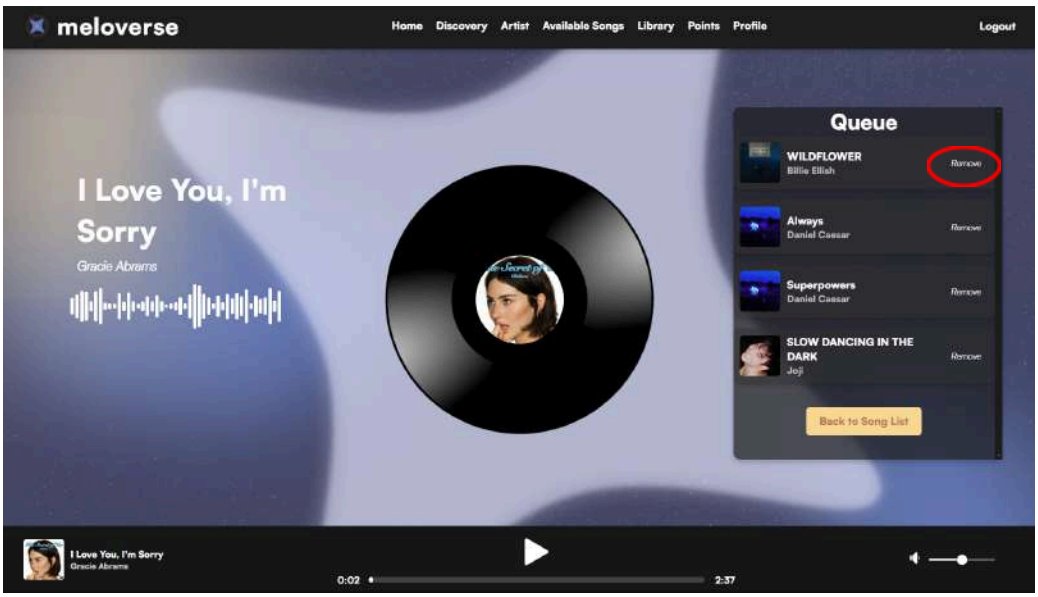
To view more information of **'Joji'**, click **'Check Out the Artist'** button to be redirected to the artist information page, here you can **'Follow'** or **'Favourite'** the artists, check out album information, view soundtracks and merchandise.



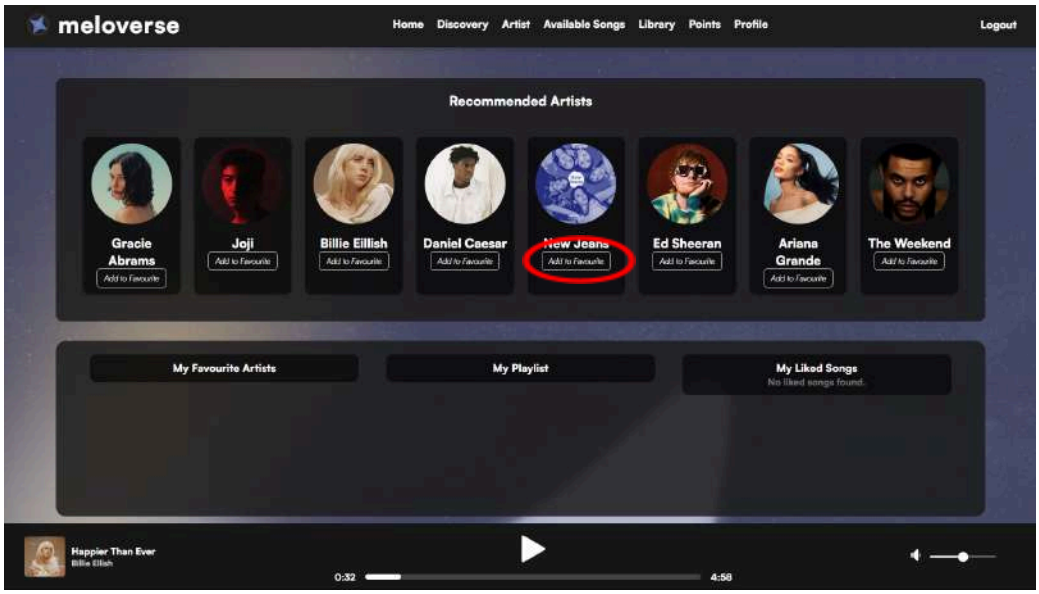
Clicking **‘Available Songs’** in the header redirects you to the **Available Song List** page, where you can browse songs. Each song has three buttons: **Add to Queue**, **Add to Liked Songs**, and **Add to Playlist** to manage your music easily.



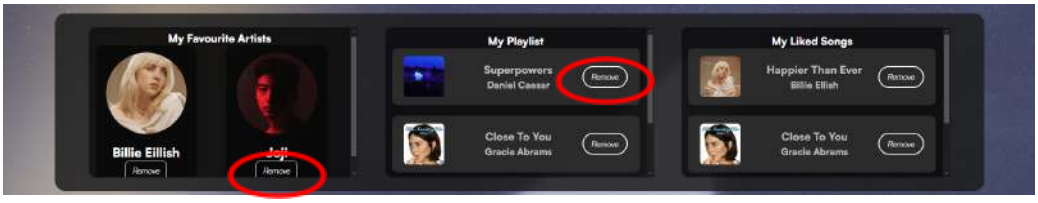
After clicking on a song title, you will be directed to the **Song Page**, where the selected music will start playing, and the **queue list** will be displayed. You can manage your queue by using the **‘Remove’** button to remove songs from the list. The currently playing song will be displayed in the footer at the bottom of the page, where you can also control your music playback, volume and play/pause.



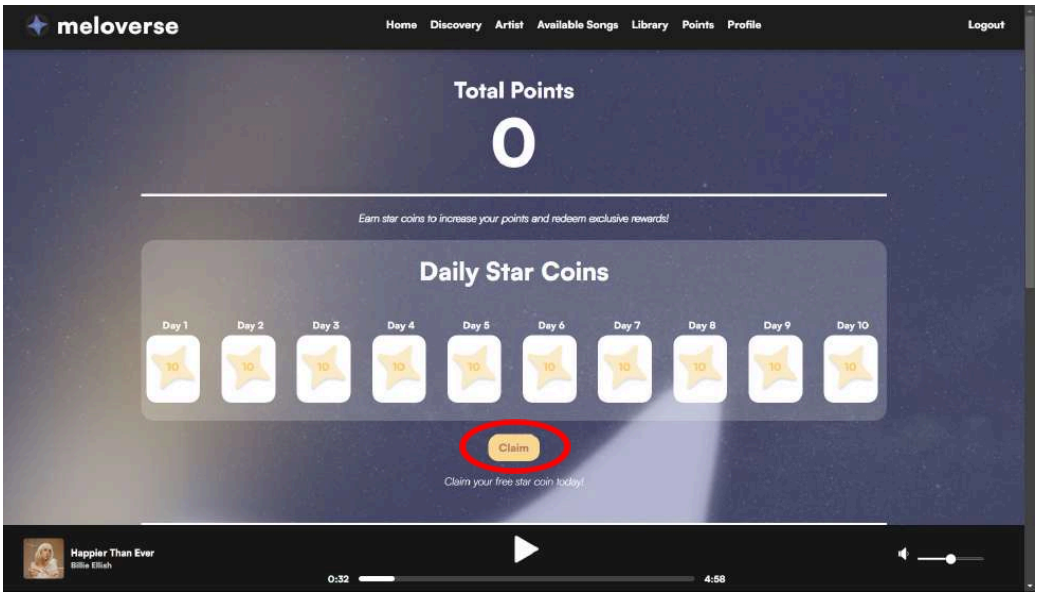
To access the Library Page, click on the **'Library'** option in the header, which will redirect you to the page which displays **your favourite artists, liked songs, and playlists**. Below each artist profile, there is an **'Add Favourite'** button; by clicking this button, the artist's profile will be added to the 'My Favourite Artist' section.



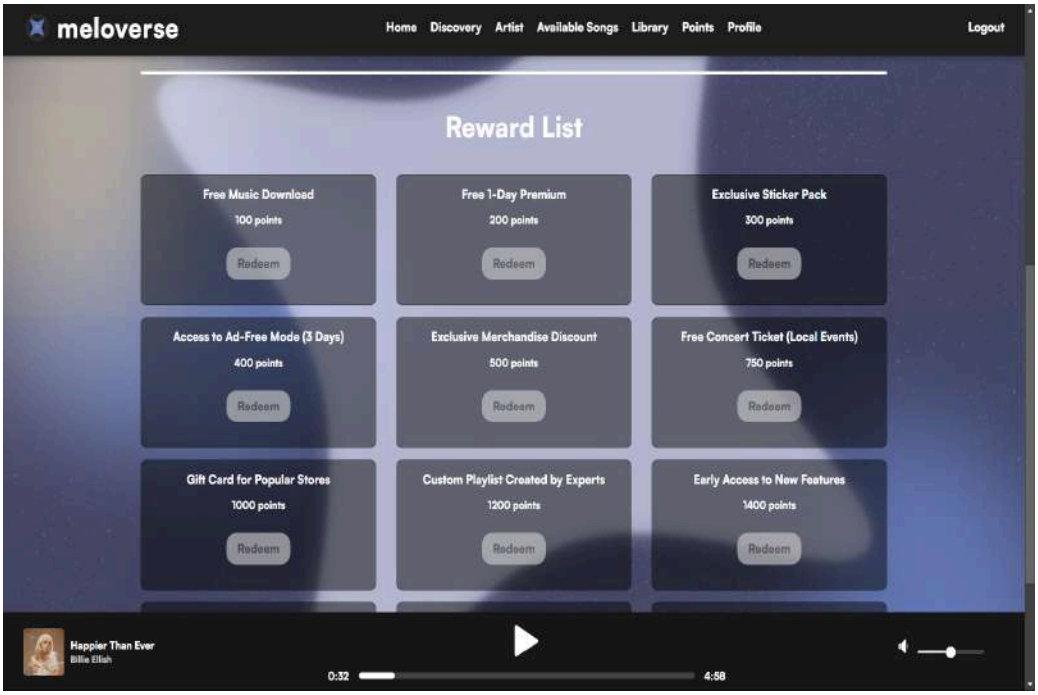
As you add songs to your **playlist** and **liked songs**, a **'Remove'** button will appear, allowing you to remove these songs if desired. Similarly, favourite artists can also be removed using the **'Remove'** button.



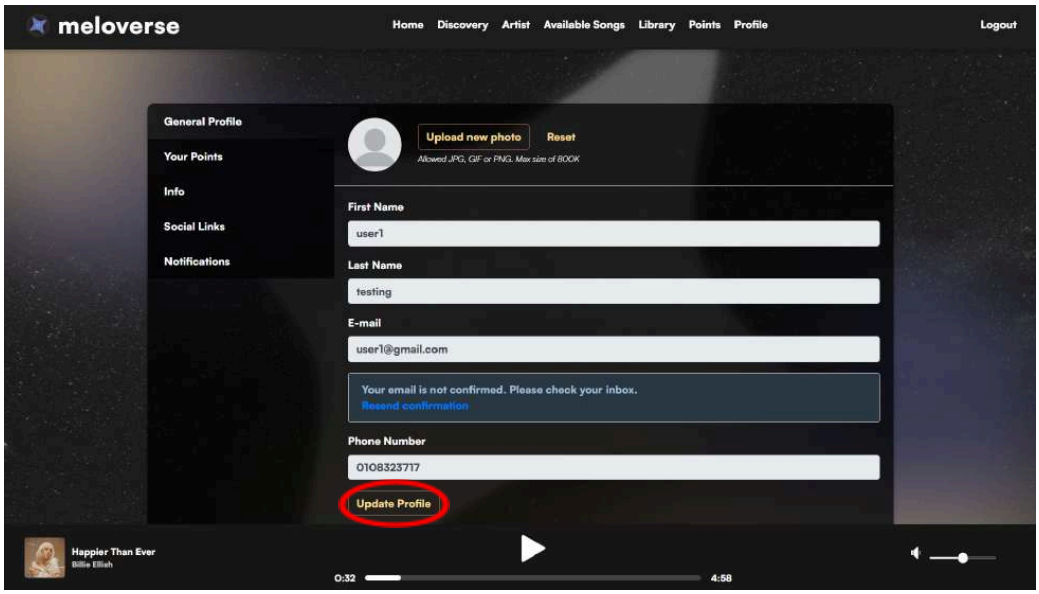
By clicking **'Points'** in the header, you will be redirected to the Points Page. This is where your points will be displayed, along with the points system and rewards system. By clicking the **'Claim'** button, you will be awarded 10 points. You can only claim more points by logging in everyday. At the same time, the daily star coins for Day 1 will be shown as claimed.



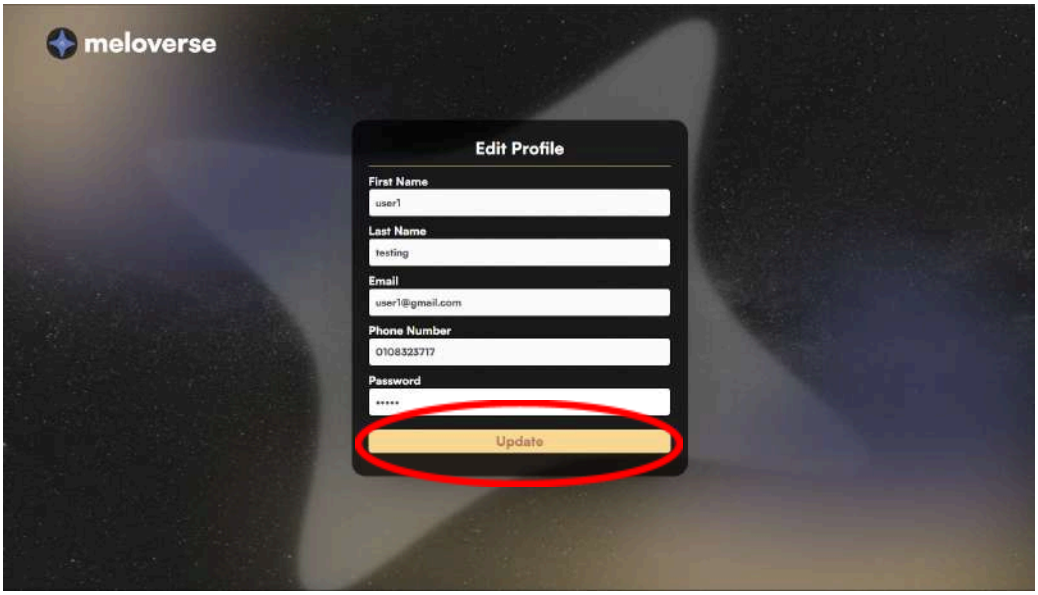
By accumulating points day by day, you can redeem amazing rewards from the rewards list below.



By clicking **'Profile'** in the header, you will be redirected to the Profile Page. This is where your user information will be displayed. To update your information, click the **'Update Profile'** button.

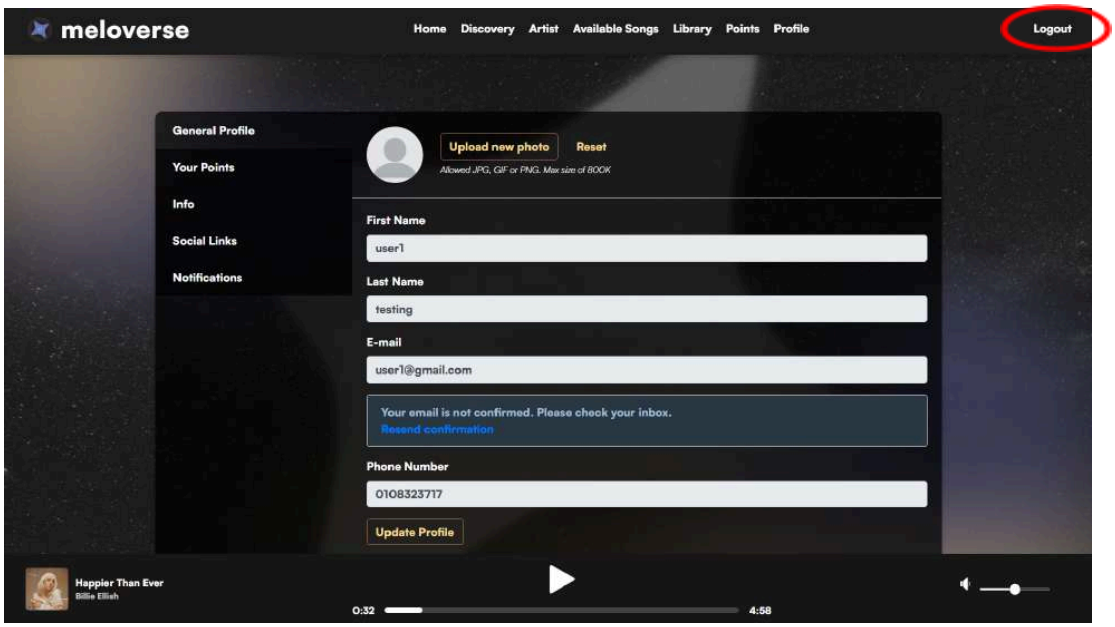


By clicking the **'Update Profile'** button, you will be redirected to the Update Profile Page.

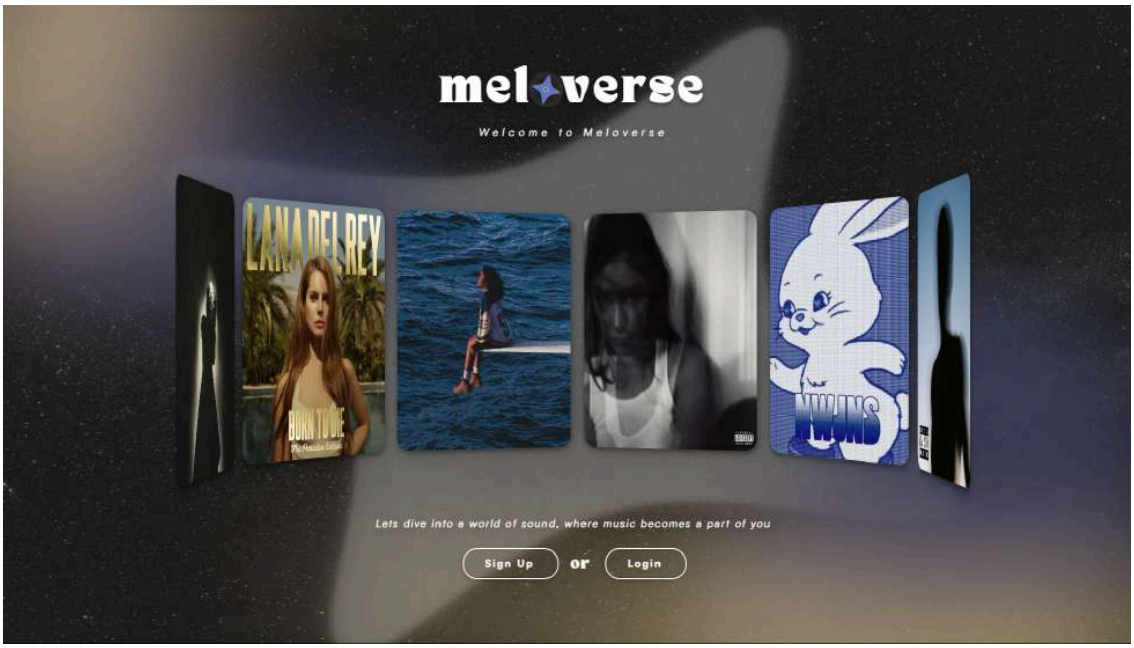


Once you have finished updating your profile information, click the **'Update'** button. You will then be redirected back to the Profile Page.

Lastly, when you're done enjoying or listening to music on the website, you can click **'Logout'** from the header section at the top right corner of the page in order to logout.



After clicking the Logout button, you will be redirected to the Landing Page, where you will have to re-login if you want to access the website again.

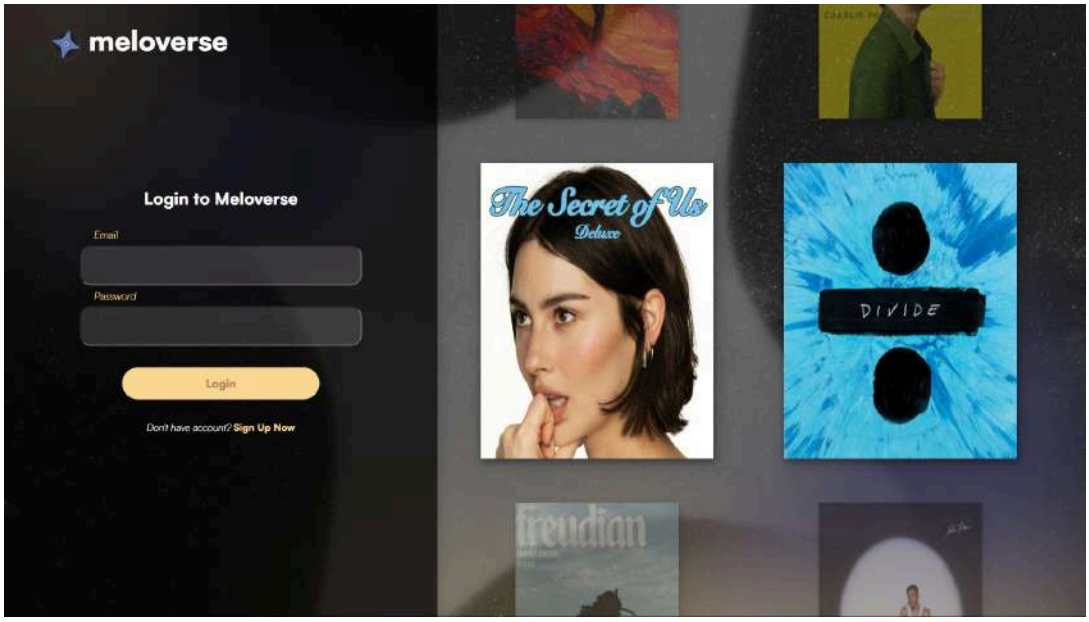


4.0 Test Case

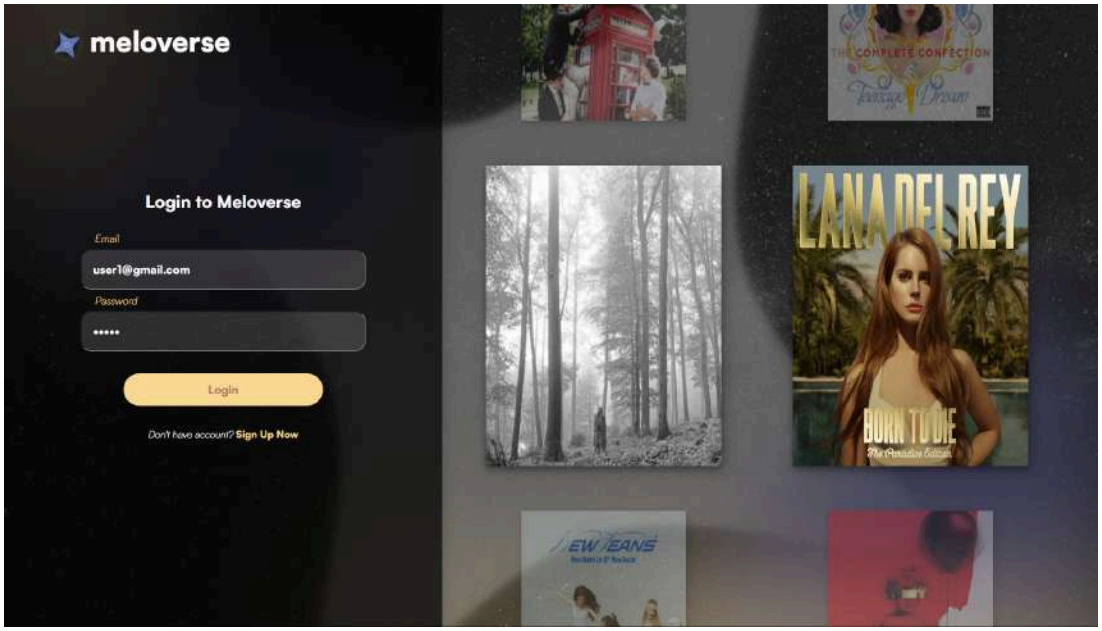
Test Case 1: User Login

Objective: Verify that users can log in successfully with valid credentials.
Steps:

1. Navigate to the login page.



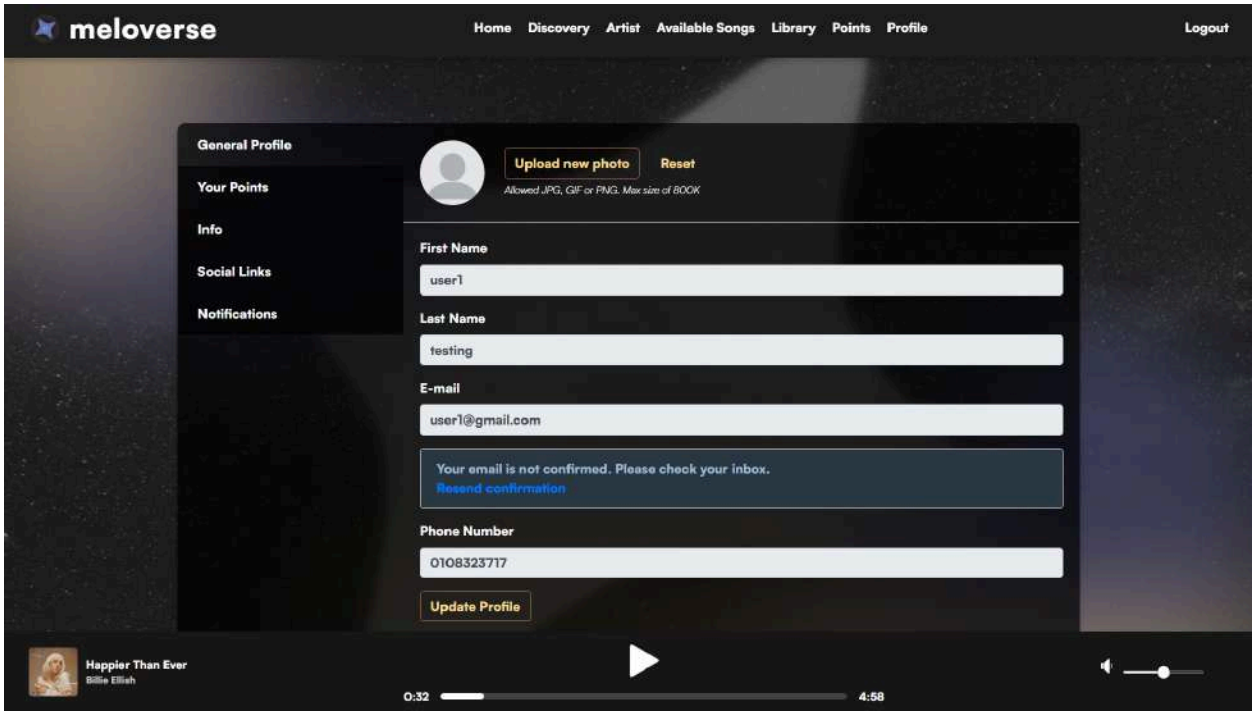
2. Enter a valid email and password.



3. Click the "Login" button.

Expected Result:

After login, the user will be redirected to the home page and when the user navigates to the profile page, the profile page will display the user's profile information such as first name, last name, email, phone number, etc.

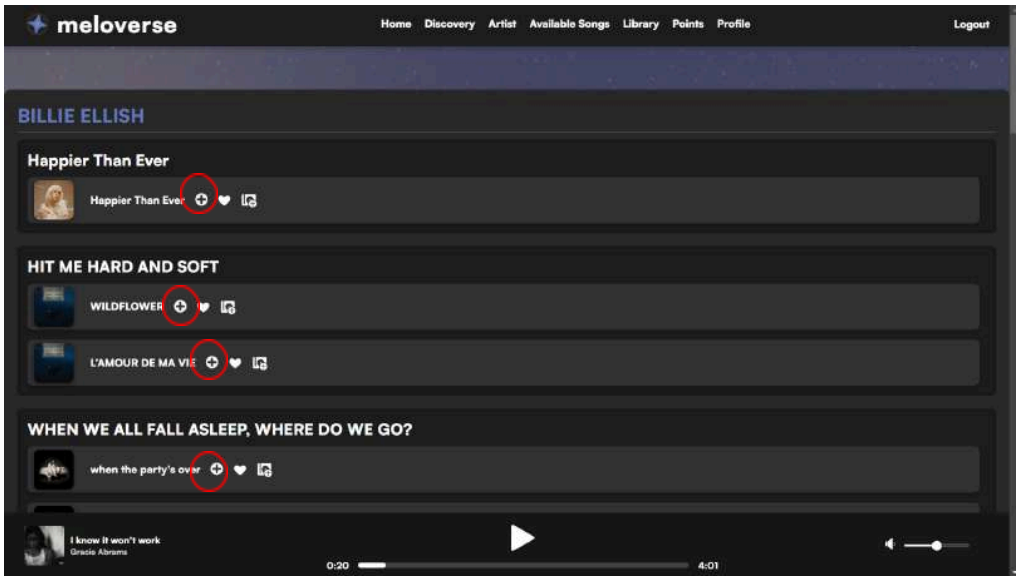


Test Case 2: Add Song to Queue

Objective: Confirm that the "Add to Queue" feature functions correctly.

Steps:

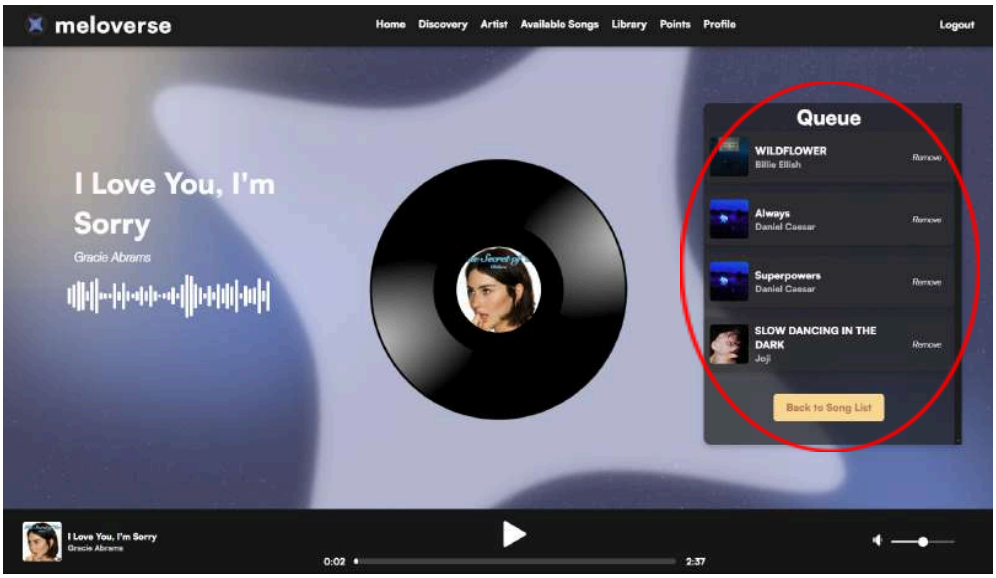
1) Navigate to the "Available Songs" page.



2) Select a few songs from the list and click the "Add to Queue" button next to it.
3) Open the song page and check whether the songs are displayed in the queue list.

Expected Result:

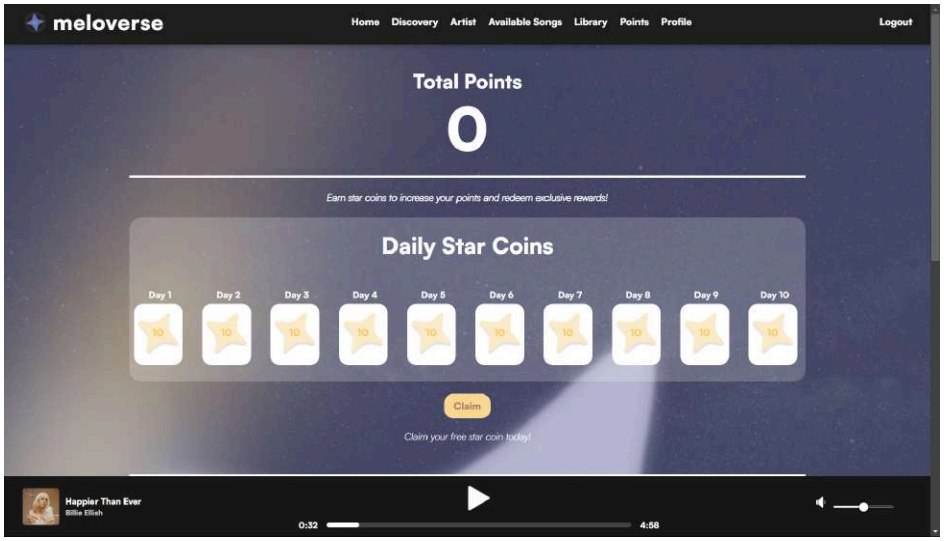
The selected song appears in the queue list.



Test Case 3: Points Claim Functionality

Objective: Ensure that users can claim daily points.
Steps:

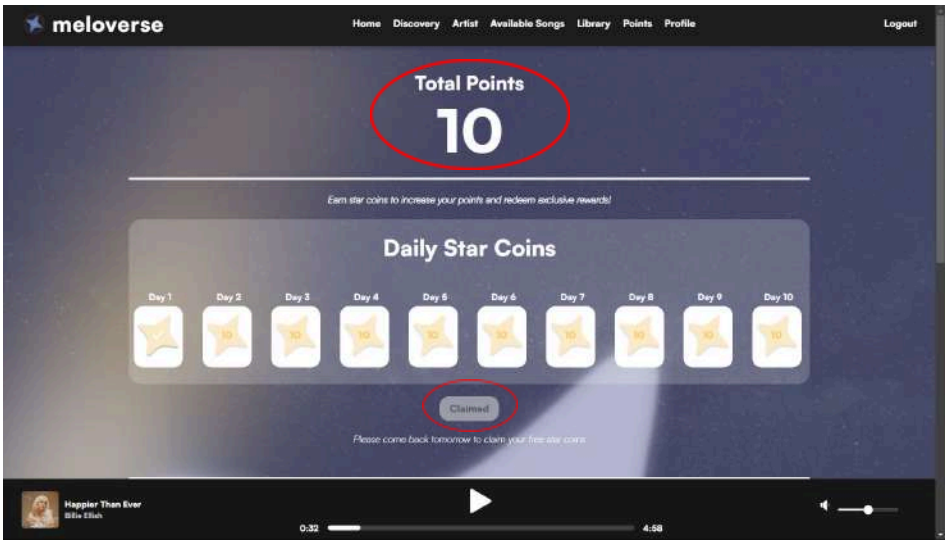
1. Login and navigate to the "Points" page.



2. Click the "Claim" button for the daily reward.

3. Observe the updated points total and the status of the daily reward (marked as "Claimed").

Expected Result:
The points increase by 10, and the daily reward for the day is marked as claimed.



5.0 Features and Functionalities

5.1 SQL Database

Our database, named login, contains seven tables: artists, favourites, likedsongs, playlist, queue, songs, and users, designed to manage music-related data such as user details, songs, playlists, and favorite tracks. It uses the InnoDB storage engine with a utf8mb4_general_ci collation for efficient data storage and retrieval.

I’m

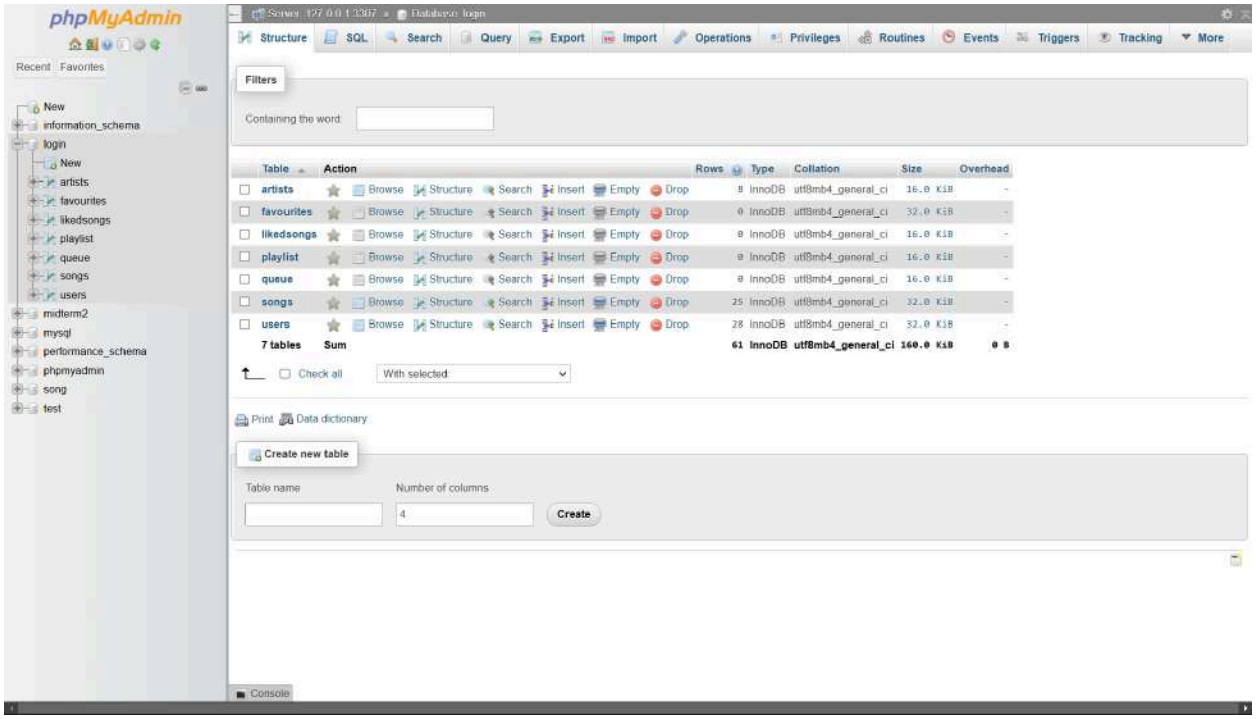


Figure 5.1. Database

5.2 HTML5 form

Figure 5.2 shows us a HTML5 form snippet that creates a login interface for Meloverse, featuring input fields for email and password, with client-side validation and autocomplete disabled for enhanced security. It also includes a call-to-action link for users to navigate to the signup page if they do not have an account.

```
<!-- HTML Form -->
<header>Login to Meloverse</header>
<form action="" method="post">
    <div class="field input">
        <label for="email">Email</label>
        <input type="text" name="email" id="email" autocomplete="off" required>
    </div>

    <div class="field input">
        <label for="password">Password</label>
        <input type="password" name="password" id="password" autocomplete="off" required>
    </div>

    <div class="field">
        <input type="submit" class="btn" name="submit" value="Login" required>
    </div>
    <div class="links">
        Don't have account? <a href="signinPage.php">Sign Up Now</a>
    </div>
</form>
```

Figure 5.2. Code Snippet of loginPage.php

5.3 SQL Commands

5.3.1 CREATE (INSERT)

Figure 5.3. Code Snippet of signinPage.php

5.3.2 READ (SELECT)

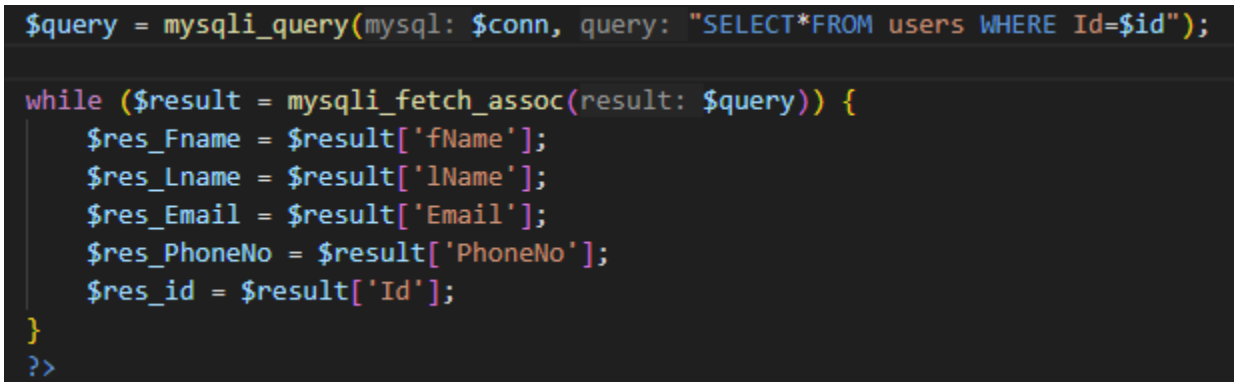


Figure 5.4. Code Snippet of profilePage.php

5.3.3 UPDATE (MODIFY)

Figure 5.5. Code Snippet of updateProfile.php

5.3.4 DELETE

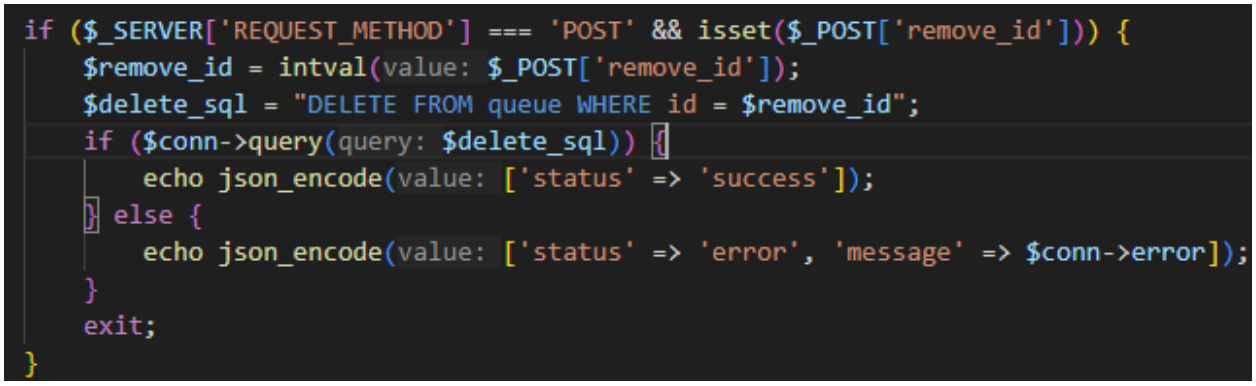


Figure 5.6. Code Snippet of queue_display.php

5.4 Usage of sessions

In this code, PHP sessions are used to store and manage temporary user-specific data across multiple pages during the user's session. The session_start() function initializes the session and allows data to persist throughout the user's interaction with the web application.

Specifically, the variable $_SESSION['current_song'] is used to keep track of the song that the user is currently interacting with or listening to. This ensures the state of the current song is preserved, enabling a seamless user experience. If the session data is unavailable, it defaults to null. On the other hand, the variable $_SESSION['is_playing'] is used to store whether the song is currently playing, ensuring the application remembers the playback state. If this data does not exist in the session, it defaults to false.

Using sessions avoids the need to pass such data between pages using URLs or form submissions, thereby enhancing security and usability. Sessions are essential for maintaining stateful data in a stateless protocol like HTTP, ensuring a more personalized and dynamic experience for the user.

```php
<?php
// Database connection
session_start();
// Assuming user authentication has been handled
$host = "localhost:3307";
$user = "root";
$pass = "";
$db = "login";
$conn = new mysqli(hostname: $host, username: $user, password: $pass, database: $db);

if ($conn->connect_error) {
    die("Failed to connect DB: " . $conn->connect_error);
}

$current_song = isset($_SESSION['current_song']) ? $_SESSION['current_song'] : null;
$is_playing = isset($_SESSION['is_playing']) ? $_SESSION['is_playing'] : false;
$conn->close();


?>
```

Figure 5.7. Code Snippet of homePage.php

5.5 Protection against SQL injection

```
$stmt = $conn->prepare(query: "SELECT * FROM users WHERE Email = ? AND Password = ?");
$stmt->bind_param(types: "ss", var: &$email, vars: &$password);
```

Figure 5.8. Code Snippet of loginPage.php

The code utilizes prepared statements and parameterized queries to protect against SQL injection attacks. By using the prepare() and bind_param() methods, the user inputs (e.g., email and password) are treated as parameters rather than executable SQL code. This ensures that any malicious input, such as "' OR 1=1 --", is not executed by the database engine but instead treated as a literal string.

Prepared statements safeguard the application by separating SQL logic from user data, thereby preventing attackers from injecting malicious SQL queries. This approach adheres to best practices in secure database interactions and significantly reduces the risk of unauthorized access or database manipulation.

5.6 Protection against HTML injection

```
$_SESSION['valid'] = htmlspecialchars(string: $row['Email']);
$_SESSION['fname'] = htmlspecialchars(string: $row['fName']);
$_SESSION['lname'] = htmlspecialchars(string: $row['lName']);
$_SESSION['phoneno'] = htmlspecialchars(string: $row['PhoneNo']);
$_SESSION['id'] = htmlspecialchars(string: $row['Id']);
```

Figure 5.9. Code Snippet of loginPage.php

The application uses the htmlspecialchars() function to sanitize output data fetched from the database before storing it in session variables. This encoding converts special characters, such as <, >, and ", into their HTML entity equivalents (e.g., <, >, ").

By sanitizing user-generated content, the application prevents malicious scripts from being executed when the data is displayed in the browser. This measure effectively mitigates Cross-Site Scripting (XSS) attacks, ensuring that any injected code is treated as plain text rather than executable HTML or JavaScript. This approach is critical for maintaining the integrity of the user interface and protecting end users from potential exploits.

5.7 CSS Usage

CSS (Cascading Style Sheets) is used in this project to enhance the visual presentation and user experience of the web application. By separating the content (HTML) from the design (CSS), the code achieves better maintainability and readability.

In the provided code down below:

```
<link rel="stylesheet" href="/Web Application Programming/CSS/loginPage.css">
```

Figure 5.10. Code Snippet of loginPage.php

For example, the external CSS file loginPage.css is linked using the <link> tag to define the styles for the login page elements such as the form, input fields, buttons, and overall layout. This ensures consistent styling across the page. Another CSS file, loginCarousel.css, is used to style the carousel feature that displays images dynamically. This adds interactivity and an aesthetically pleasing design for the users.

Custom styling improves the user interface (UI) by organizing content, defining colors, fonts, spacing, and responsiveness, thereby making the web application visually appealing and user-friendly. Using external CSS allows for reusability and modularity, where the same styles can be applied to multiple pages. It also promotes faster page loading since the CSS file is cached by the browser. Provided below Figure 5.11 displays all of the CSS we used.

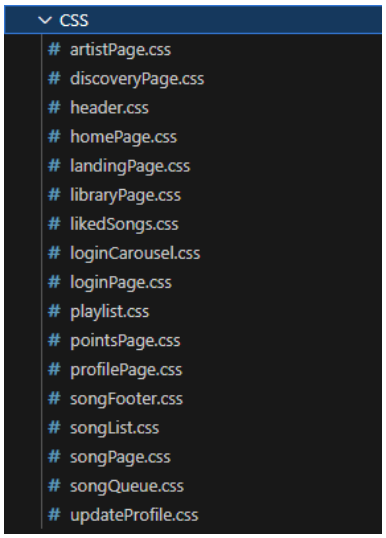


Figure 5.11. CSS list

5.8 AJAX

5.8.1 AJAX Call for Song Removal:

When the "Remove" button is clicked, an AJAX POST request is sent to the same script (queue_display.php) with the remove_id of the song to be removed from the queue table. The server-side script processes the request, deletes the song from the queue, and sends a JSON response indicating success or failure.

```
$.ajax({
    url: '/Web Application Programming/php/queue_display.php',
    type: 'POST',
    data: { remove_id: queueId },
    success: function(response) {
        console.log(response); // Debug server response
        const result = JSON.parse(response);
        if (result.status === 'success') {
            parentItem.remove();
            if ($('#queue-list .queue-item').length === 0) {
                $('#queue-list').html('<p>No songs in the queue.</p>');
            }
        } else {
            alert('Error removing song: ' + result.message);
        }
    },
    error: function() {
        alert('An error occurred. Please try again.');
    }
});
```

Figure 5.12. Code Snippet of queue_display.php

5.8.2 Server-side Handling:

The PHP script checks if the request method is POST and if remove_id is set. If so, it performs a DELETE query on the queue table to remove the specified entry.

```
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['remove_id'])) {
    $remove_id = intval(value: $_POST['remove_id']);
    $delete_sql = "DELETE FROM queue WHERE id = $remove_id";
    if ($conn->query(query: $delete_sql)) {
        echo json_encode(value: ['status' => 'success']);
    } else {
        echo json_encode(value: ['status' => 'error', 'message' => $conn->error]);
    }
    exit;
}
```

Figure 5.13. Code Snippet of queue_display.php

5.9 JSON

5.9.1 Server-to-Client Communication

When a song is removed from the queue, the PHP server script sends a response in **JSON format**. This happens in the following code snippet:

```php
if ($conn->query(query: $delete_sql)) {
    echo json_encode(value: ['status' => 'success']);
} else {
    echo json_encode(value: ['status' => 'error', 'message' => $conn->error]);
}
exit;
```

Figure 5.14. Code Snippet of queue_display.php

The PHP json_encode() function is used to create a JSON-encoded response. The response includes:

- status: Indicates whether the operation succeeded (success) or failed (error).
- message: If there's an error, the error message is included.

5.9.2 Client-side Handling:

The parsed JSON object (result) is used to determine the next actions:

- If result.status === 'success', the song is removed from the UI.
- If result.status === 'error', an alert displays the error message.

Example of handling JSON response in JavaScript:

```javascript
success: function(response) {
    console.log(response); // Debug server response
    const result = JSON.parse(response);
    if (result.status === 'success') {
        parentItem.remove();
        if ($('#queue-list .queue-item').length === 0) {
            $('#queue-list').html('<p>No songs in the queue.</p>');
        }
    } else {
        alert('Error removing song: ' + result.message);
    }
},
```

Figure 5.15. Code Snippet of queue_display.php