**VIETNAM GENERAL CONFEDERATION OF LABOR**

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**

Logo

Description automatically generated

**MIDTERM PROJECT**

**Mobile Application Development**

*Student: Lê Ngọc Bình*

*Phone Wai Yan Moe*

*Mai Thái Phong*

*Trần Huy Tiến*

*student ID: 522K0004*

522K0047

522C0005

522K0015

*Class: 22K50201*

**HO CHI MINH CITY, 2024**

**VIETNAM GENERAL CONFEDERATION OF LABOR**

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**

Logo

Description automatically generated

**MIDTERM PROJECT**

**Mobile Application Development**

*Student: Lê Ngọc Bình*

*Phone Wai Yan Moe*

*Mai Thái Phong*

*Trần Huy Tiến*

*Student ID: 522K0004*

522K0047

522C0005

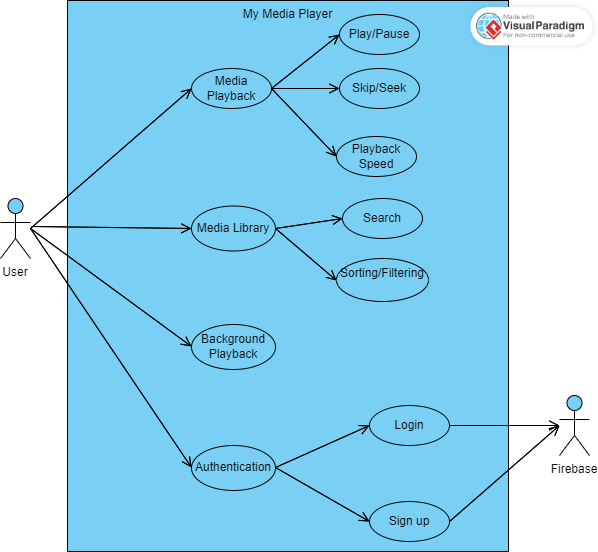
522K0015

*Class: 22K50201*

**HO CHI MINH CITY, 2024**

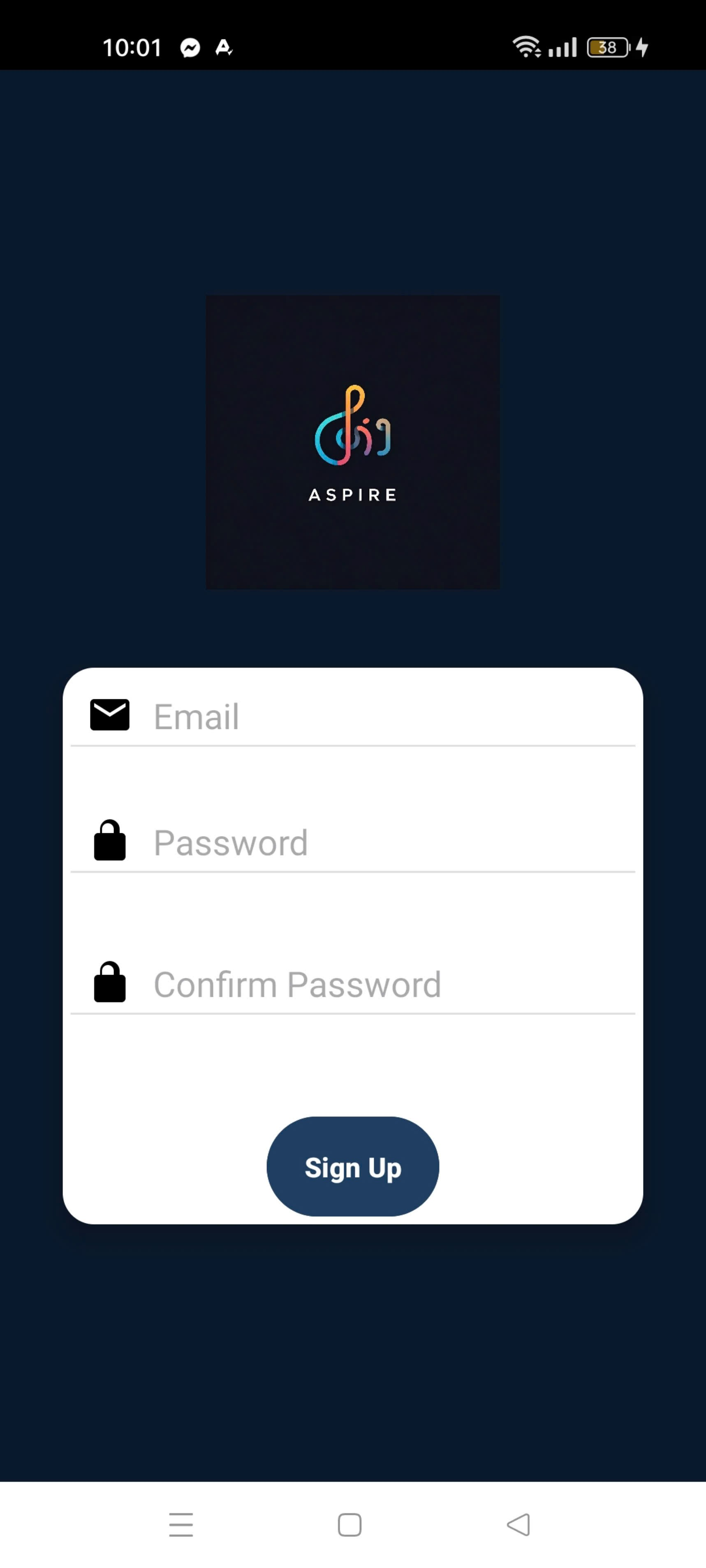
|  |  |  |
| --- | --- | --- |
| **Member name** | **Role** | **Contribution** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* **Use case diagram**



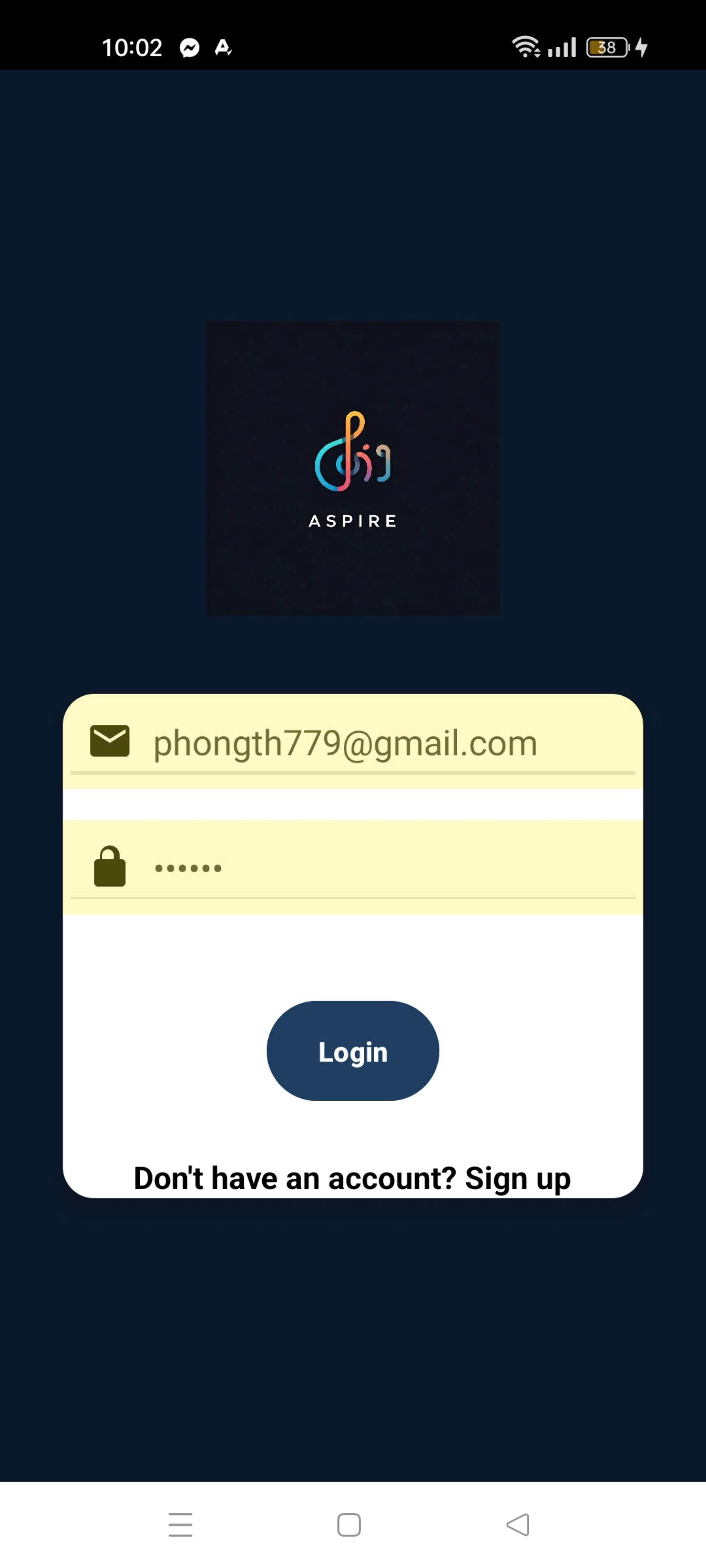
* **Screenshot**

1. **Sign-Up Screen**



- A form that allow users to register by entering their email and password.

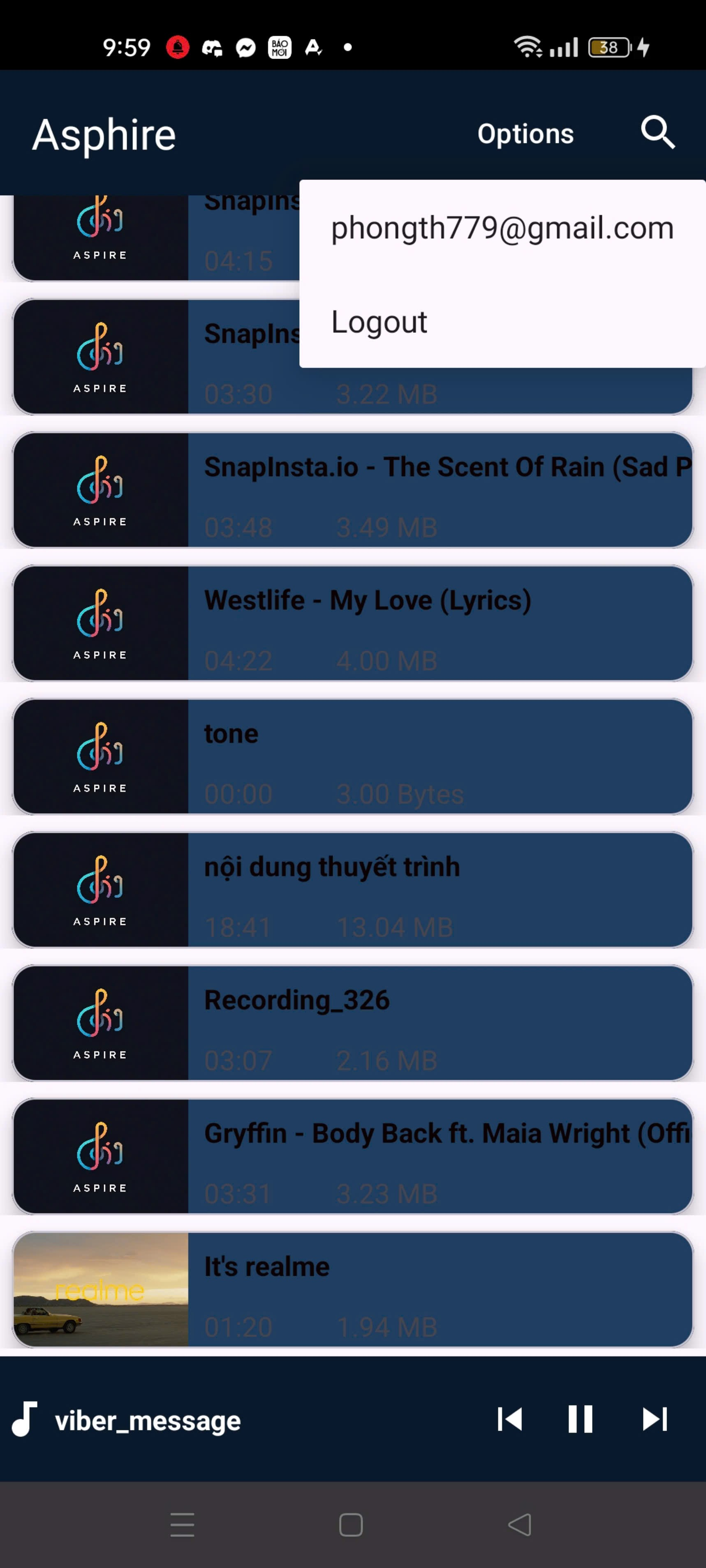
1. **Login Screen**



- Users input their credentials to login the app .

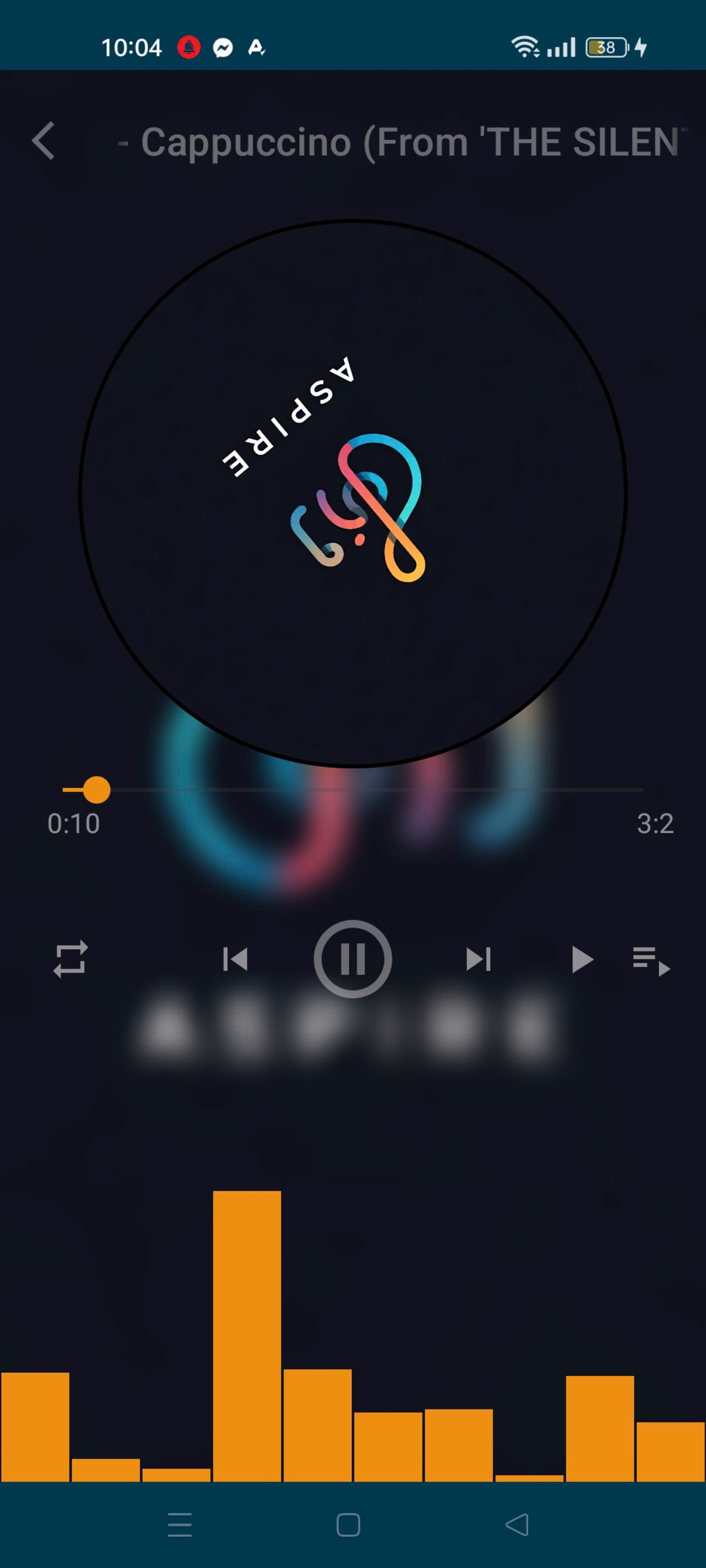
1. **Media Library(i will take this screenshot again if the app is fixed)**

- Displays a list of available songs using a RecyclerView with data inserted by the *SongAdapter*.

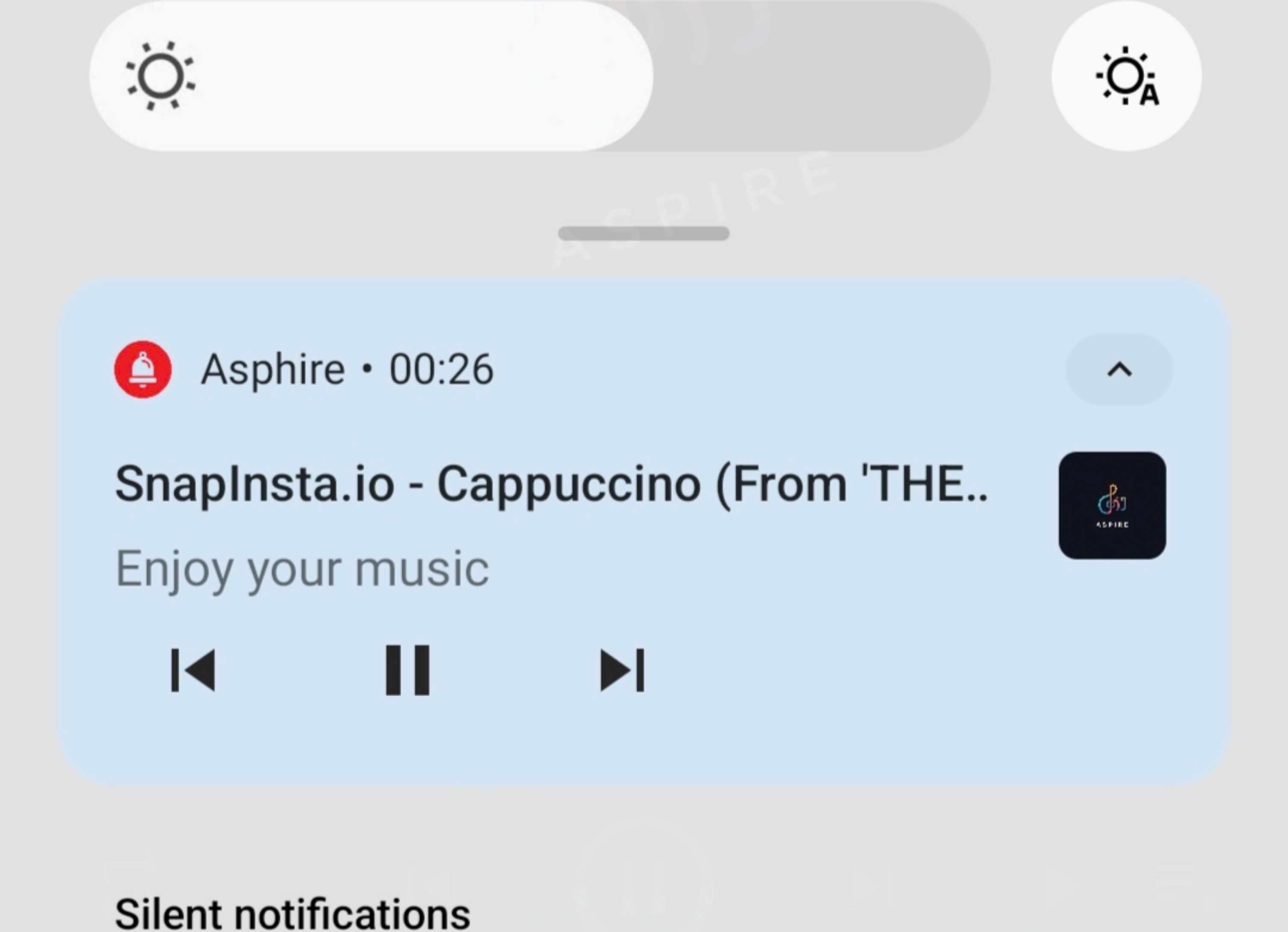


1. **Song PlayerView Screen**

- Users can control song playback (play, pause, next, etc.).



**5. Song Background**

****

* **Advantages and disadvantages**

1. **Advantages**

**1. Firebase Authentication Integration:**

- The integration with Firebase provided a ***secure and reliable solution*** for managing user registration and login.

**2. Modular and Scalable Code Design:**

- The use of the *adapter pattern* in the `SongAdapter` class ensured clear separation between the data layer and UI components, improving code modularity, **reusability, and scalability** for future enhancements.

**3. Object-Oriented Programming (OOP) Principles:**

- By encapsulating song data within the **`Song` class**, the application adhered to OOP best practices, such as encapsulation and abstraction. This structured data management reduced errors and ensured better maintainability.

**4. User-Friendly Interface and Navigation:**

- The app provided intuitive navigation **(standardized UI taken from the reference)** between activities (e.g., from sign-up to the main library). Smooth transitions via Intents contributed to a seamless user experience.

**5. Efficient Data Binding and Performance:**

- **The RecyclerView paired with the `SongAdapter**` handled large datasets efficiently, ensuring smooth scrolling and data binding even with an extensive music library.

1. **Disadvantages**

**1. UI Responsiveness Across Devices:**

- Achieving consistent UI alignment across different screen sizes and resolutions was challenging. Have problem syncing the background playback with the in-app content.

**2. Media Playback Management Issues:**

- Implementing playback controls such as pause, resume, and handling interruptions (e.g., using another audio required app) resulting in some delays during development.

**3. Playlist Management Constraints:**

- Although the application supports displaying songs, playlist creation and management functionality is limited.

**4. Dependency on Network Connectivity:**

- As Firebase authentication requires an *active internet connection*, users cannot log in or register offline. This dependency on network availability may limit usability in areas with poor connectivity.

* **Techniques Applied**

1. **Firebase Authentication Integration(Login/Signup)**

- **`createUserWithEmailAndPassword()`:** Used for registering new users by creating accounts with email and password.

- `signInWithEmailAndPassword()`: Allows users to log in with existing credentials.

- **`FirebaseAuth.getInstance()`:** Provides an instance of Firebase Authentication for managing user sessions.

- Error Handling: Specific error messages are handled (e.g., weak passwords, duplicate accounts) to enhance user feedback through **Toast messages**.

1. **Adapter Design Pattern for Data Binding(Items on RecyclerView)**

- `SongAdapter` Class: Implements the adapter pattern to bind a list of songs to a **RecyclerView**. This ensures scalability for managing large datasets efficiently.

- **`onBindViewHolder()`:** Dynamically binds song details (like title and artwork) to individual UI components for each item in the list.

- RecyclerView Integration: Provides smooth scrolling and efficient rendering of media data.

1. **Object-Oriented Programming (OOP) Principles(Song class)**

- **`Song` Class:** Encapsulates song-related metadata (title, URI, artwork, size, and duration) following OOP principles.

- **Getter Methods:** Provides controlled access to song attributes, ensuring data consistency and security.

- **Modular Code Design:** Each class and activity has specific responsibilities, promoting code readability and maintainability.

1. **Intent-Based Navigation between Activities (Login->MainScreen carrying the username,...)**

- **Intents:** Used to switch between SignUpActivity, LoginActivity, and MainActivity, providing smooth user flow.

- **`startActivity()` and `finish()`:** Manage activity transitions to ensure the user navigates seamlessly and avoids unintended back navigation after logging in.

1. **User Feedback Mechanism(Toast message triggering with certain events)**

- **Toast Messages:** Display quick, non-intrusive notifications for events such as successful sign-up, login failures, and authentication errors.

- **Form Validation:** Ensures the email and password fields are not left empty, providing instant feedback to users before they proceed.

1. **UI and Data Management Techniques**

- **XML Layouts:** Used for designing user interfaces for activities like sign-up, login, and the media library.

- **RecyclerView and ViewHolder Pattern:** Ensures memory-efficient handling of song lists, reducing lag during navigation.

- **Responsive Design:** XML constraints are used to adapt layouts to different screen sizes, though some minor alignment challenges were noted.(SHOULD WE KEEP LIMITATIONS HERE?)

1. **Error Handling and Session Management**

- **Authentication Errors:** Custom error messages are shown for failed sign-up or login attempts, enhancing user experience.

- **Session Management:** `FirebaseAuth.getCurrentUser()` checks if a user is already logged in, allowing for appropriate redirection to the main activity.

* **References**

**Part 1. Music Player in android studio | Listing songs on screen**

<https://www.youtube.com/watch?v=__Yardq8bmI>

# Part 2. Playing songs, showing player view and control buttons for music player in android studio

<https://www.youtube.com/watch?v=GyatQmrX8D0&t=4407s>

# Part3: Playback service with notification in Music Player in android studio

<https://www.youtube.com/watch?v=x-MquHD14zk>