****

**PROJECT DOCUMENT**

**ONLINE MUSIC PLAYER**

**ANDROID APPLICATION**

**Submitted by**

**Duong, Nguyen Thuy**

**Approved by**

**Professor Duc, Nguyen Manh**

Proposal Review Panel Representative:

Name Signature Date

Capstone Project 1- Mentor:

Name Signature Date

**Table of Contents**

**[I. Introduction 4](#_Toc29665)**

[1. Purpose 4](#_Toc25618)

[2. Application requirements 4](#_Toc11544)

**[II. GUI Design 5](#_Toc2536)**

**[1. Main Activity 5](#_Toc32190)**

[a. Header Bar 5](#_Toc11676)

[b. Search View 5](#_Toc15695)

[c. Categories 6](#_Toc11267)

[d. Trending Section 6](#_Toc21358)

[e. Other Sections 6](#_Toc23503)

[f. Player View 6](#_Toc31541)

**[2. Login Activity 7](#_Toc5655)**

[a. TextView and ImageView 7](#_Toc12833)

[b. EditTexts 7](#_Toc20513)

[c. Login Button 7](#_Toc14132)

[d. Forgot Password Button 8](#_Toc14451)

[e. Register Button 8](#_Toc25662)

**[3. Forgot Password Activity 8](#_Toc6519)**

[a. TextView and Image View 8](#_Toc14963)

[b. EditText 8](#_Toc5685)

[c. Send Email Button 9](#_Toc1212)

[d. Back to Login Page Button 9](#_Toc14835)

[e. Register Button 9](#_Toc17752)

**[4. Signup Activity 9](#_Toc22970)**

[a. TextView and ImageView 10](#_Toc4559)

[b. EditTexts 10](#_Toc32525)

[c. Create Account Button 10](#_Toc22348)

[d. Go to Login Page Button 10](#_Toc3417)

**[5. Songs List Activity 10](#_Toc4808)**

[a. ImageView and TextView 10](#_Toc8166)

[b. RecyclerView 10](#_Toc9150)

**[6. Player Activity 12](#_Toc19537)**

[a. TextViews 12](#_Toc21814)

[b. ImageView and Gif 12](#_Toc5592)

[c. PlayerView 12](#_Toc22634)

**[III. Backend Design 12](#_Toc32767)**

[1. Query data from Firebase 12](#_Toc1362)

[2. Set up RecyclerView 13](#_Toc25426)

[3. Update Views Count 13](#_Toc6363)

[4. Set up Trending Section 14](#_Toc32085)

[5. Filter content in SearchView 14](#_Toc9626)

[6. Show Now Playing View 15](#_Toc15587)

[7. Login 15](#_Toc21294)

[8. Signup 16](#_Toc5767)

[9. Send password reset email 16](#_Toc10683)

**[IV. Other Techniques 17](#_Toc18580)**

[1. Firebase 17](#_Toc21018)

[2. MyExoPlayer 17](#_Toc29959)

**[V. Conclusion and Development Direction 17](#_Toc14516)**

1. **Introduction**

Music has the power to uplift our spirits, evoke emotions, and create lasting memories. In today's fast-paced digital world, the demand for seamless access to our favorite tunes has never been higher.

To meet this demand and as a submission for educational purposes, we present an innovative solution - an online music player designed for the Android platform.

1. **Purpose**

The primary aim of this project is educational, focusing on the development of skills and understanding in Android application development. By creating a robust and user-friendly music player application, learners will gain hands-on experience in implementing fundamental concepts such as user interface design, data management, and integration with external services.

Additionally, this project serves to foster creativity and problem-solving abilities by tackling real-world challenges in software development.

1. **Application requirements**

**Online Music Streaming Integration:**

* The application must establish a secure connection with Firebase to stream music online.
* Ensure efficient handling of data transfer and minimize latency for smooth playback.

**Playback Control and Scrubbing Functionality:**

* Develop user-friendly controls for adjusting playback speed and scrubbing through songs.
* Ensure seamless synchronization between user actions and audio playback.

**Music Categorization:**

* Implement a robust categorization system to organize songs into different categories.
* Allow for easy navigation and exploration of music based on categories.

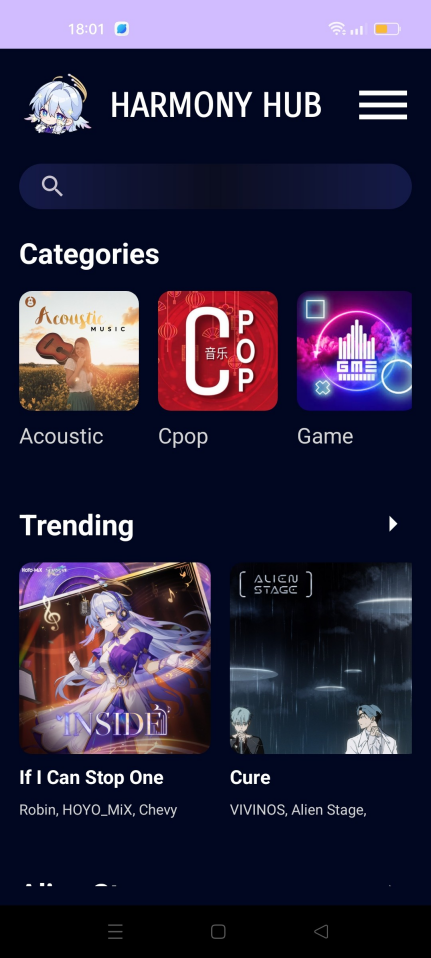
**User Authentication System:**

* Design and implement a user authentication mechanism for registration, login, and logout functionalities.
* Ensure data security and user privacy through encryption and secure authentication protocols.

**Song Search Capability:**

* Develop a search feature that allows users to find songs efficiently by title, artist, or keywords.
* Ensure fast and accurate search results through optimized algorithms and indexing techniques.

1. **GUI Design**
2. **Main Activity**



1. **Header Bar**

Header Bar is a Relative Layout which includes the application logo, application name and a menu icon.

Menu icon: Is an Image View which will show a Popup Menu whenever user clicks on it.

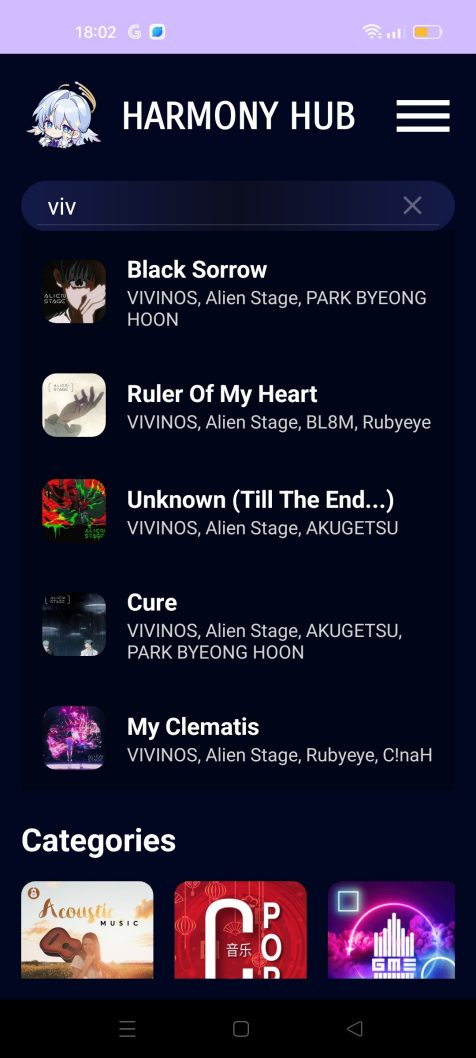
Popup Menu: If the user is logged in, the Popup Menu will show 2 items: Logout and My Account buttons. If the user is not logged in, the Popup Menu will show Login button.

Below the Header Bar there is a Scroll View, inside which is a Linear Layout.

1. **Search View**

Search View is the first item inside the Linear Layout. It includes a Search View and a Recycler View which is set invisible and only visible when showing search results. The Recycler View uses a layout (song\_list\_item\_recycler\_row.xml) and an Adapter class (SongsListAdapter.java) to show the search results.

Song\_list\_item\_recycler\_row.xml: Includes the song’s cover image, song’s title and subtitle. If user clicks on it, the app will navigate to the Player Activity.



1. **Categories**

Categories is the next item in the Linear Layout. It includes a Text View (Categories) and a Recycler View which uses a layout (category\_item\_recycler\_row.xml) and an Adapter class (CategoryAdapter.java) to show the list of categories.

Category\_item\_recycler\_row.xml: Includes the category’s cover image and category’s name. If user clicks on a category, the app will navigate to the Songs List Activity.

1. **Trending Section**

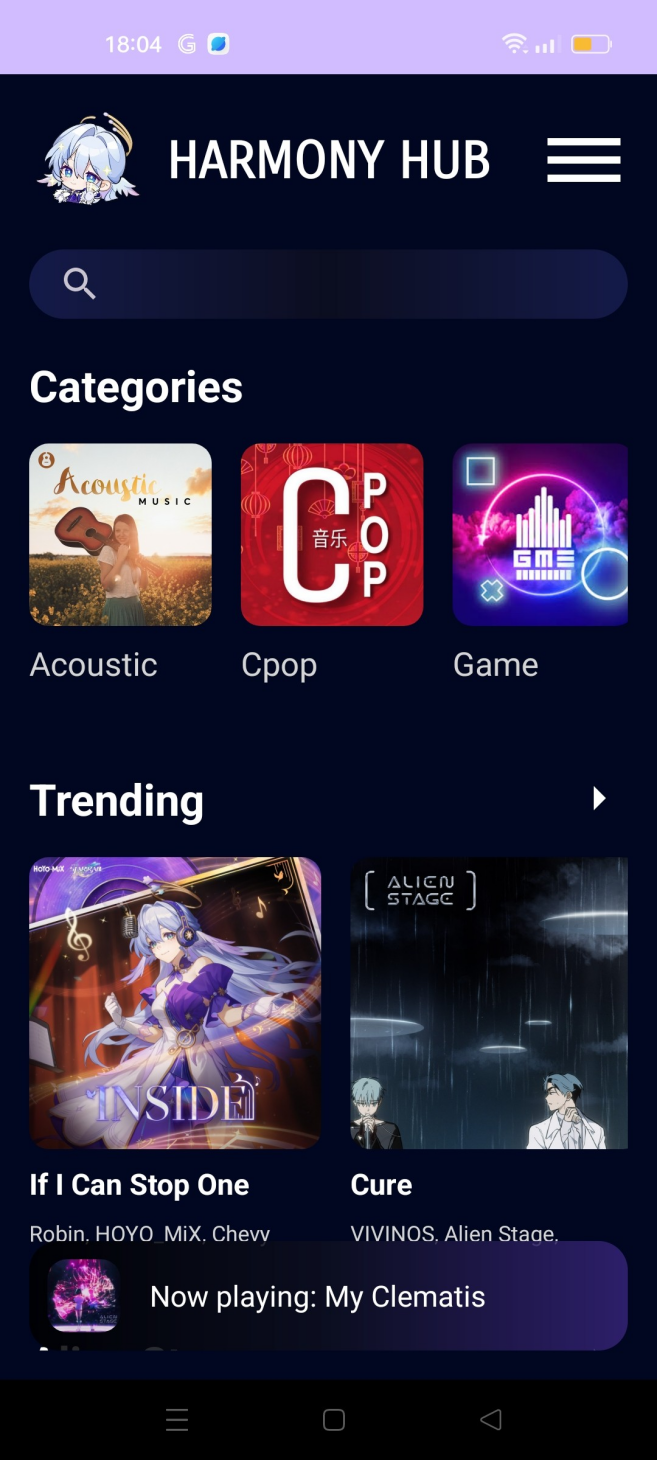
Trending section includes a Text View (Trending) and a Recycler View which uses a layout (section\_song\_list\_recycler\_row.xml) and an Adapter class (SectionSongListAdapter.java) to show the list of top 5 songs with the most view count.

Trending\_song\_recycler\_row.xml: Includes the song’s cover image, its title and subtitle. If user clicks on the Text View, the app will navigate to the Songs List Activity. If user clicks on the song, the app will navigate to the Player Activity.

1. **Other Sections**

The other sections is just like the Trending section, except for the fact that they are static on the database.

1. **Player View**



Player View is outside the Scroll View, so it will always appear at the bottom of the screen in the Main Activity whenever a song is playing.

If user clicks on the Player View, the app will navigate to the Player Activity where the song is playing.

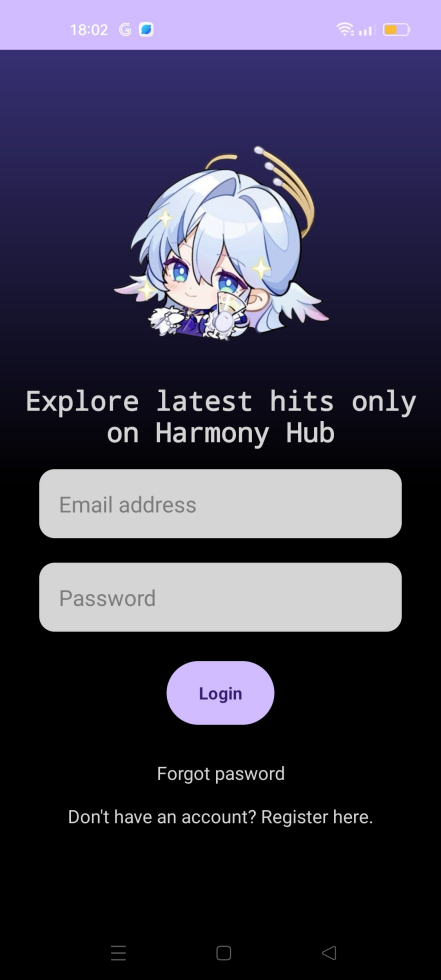
1. **Login Activity**
2. **TextView and ImageView**

Welcome Message: Positioned at the center of the screen, these TextView and ImageView display the application logo and a welcoming message to the users.

1. **EditTexts**

Email EditText: An input field for the user to enter their email address.

Password EditText: An input field for the user to enter their password.



1. **Login Button**

Functionality: When clicked, this button initiates the login process. It interacts with the back-end to authenticate the user's credentials.

Design: A prominent, visually appealing button that stands out on the screen.

1. **Forgot Password Button**

Functionality: Redirects users to the Forgot Password Activity where they can initiate the process of resetting their password.

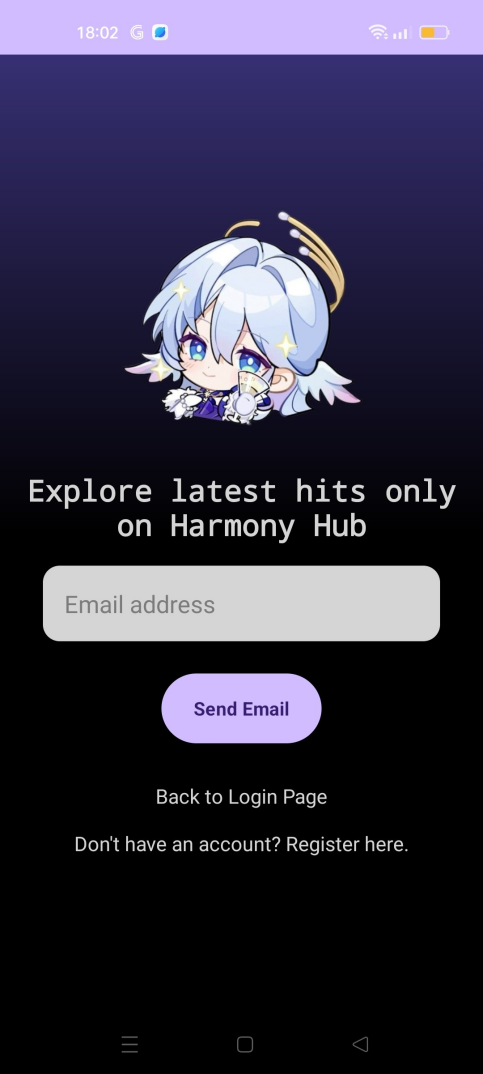
Design: A simple TextView styled to look like a button, encouraging users to recover their password if they have forgotten it.

1. **Register Button**

Functionality: Navigates users to the Signup Activity where they can create a new account.

Design: A TextView that looks like a link, prompting users who do not have an account to register.

1. **Forgot Password Activity**
2. **TextView and Image View**

Welcome Message: Positioned at the center of the screen, these TextView and ImageView display the application logo and a welcoming message to the users.  


1. **EditText**

Email EditText: An input field where users can enter their email address to receive a password reset link.

1. **Send Email Button**

Functionality: When clicked, this button triggers the process of sending a password reset email to the user.

Design: A visually distinct button that is easy to find and interact with.

1. **Back to Login Page Button**

Functionality: Redirects users back to the Login Activity.

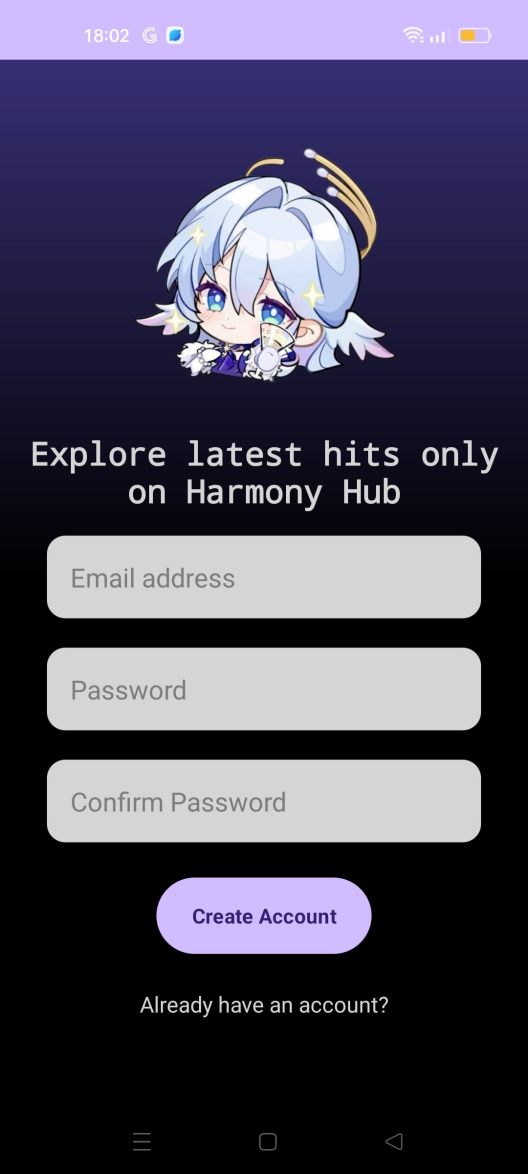
Design: Styled as a TextView link, making it intuitive for users to navigate back.

1. **Register Button**

Functionality: Directs users to the Signup Activity for creating a new account.

Design: Styled as a TextView link, prompting users who don't have an account to sign up.

1. **Signup Activity**



1. **TextView and ImageView**

Welcome Message: Positioned at the center of the screen, these TextView and ImageView display the application logo and a welcoming message to the users.

1. **EditTexts**

Email EditText: An input field where users can enter their email address.

Password EditText: An input field where users can enter their desired password.

Confirm Password EditText: An input field where users can re-enter their password to confirm it.

1. **Create Account Button**

Functionality: When clicked, this button triggers the account creation process.

Design: A visually distinct button that is easy to find and interact with.

1. **Go to Login Page Button**

Functionality: Redirects users to the Login Activity if they already have an account.

Design: Styled as a TextView link, making it intuitive for users to navigate to the login screen.

1. **Songs List Activity**
2. **ImageView and TextView**

ImageView: Displays an image related to the song category/section.

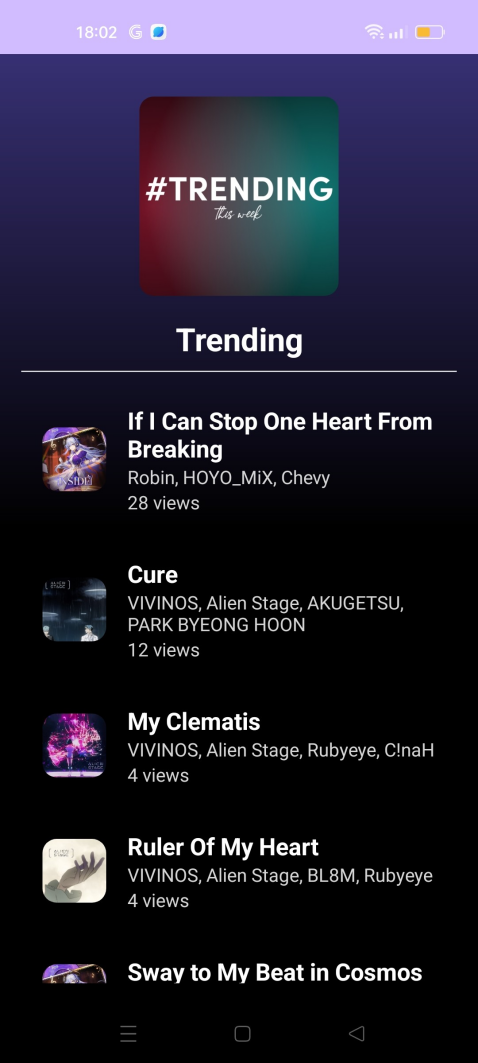
TextView: Displays the name of the song category/section.

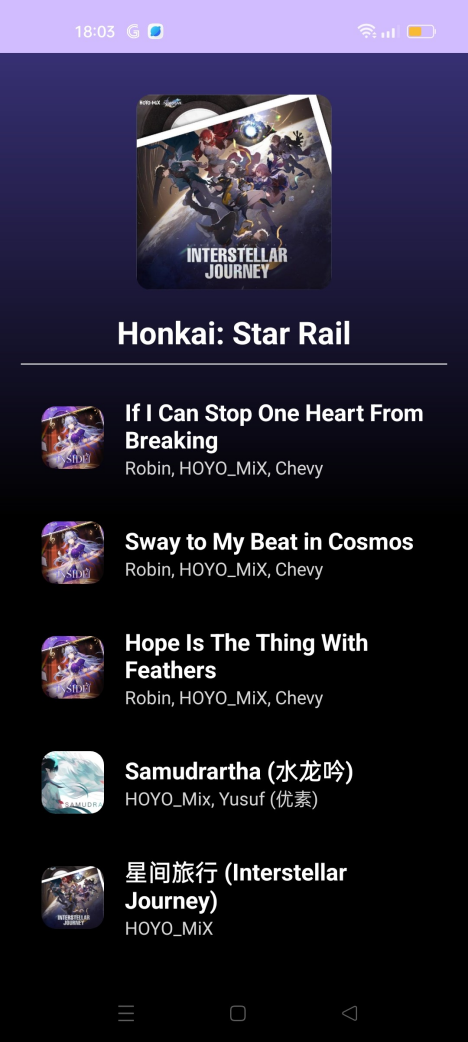
1. **RecyclerView**

RecyclerView displays the list of songs in the selected category/section.

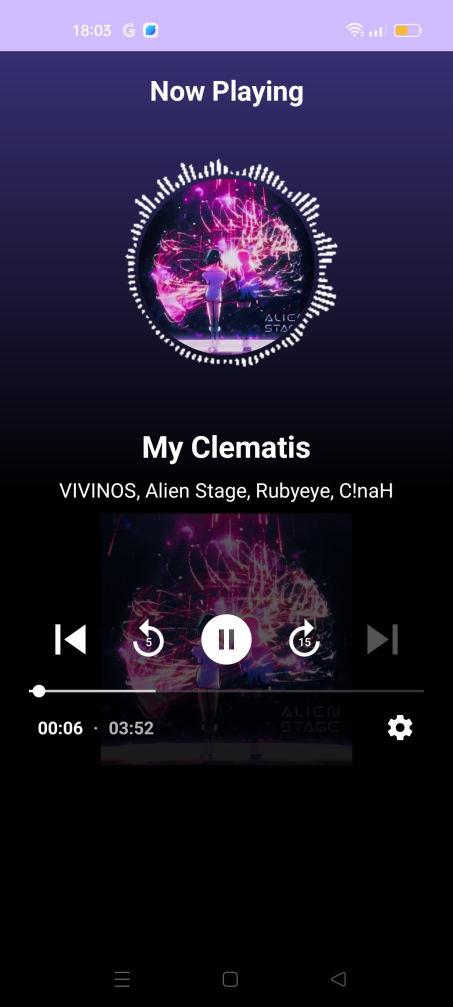
If the section is the Trending section: Using the trending\_song\_recycler \_row.xml layout and the TrendingSongsAdapter for the list. The trending\_song\_ recycler\_row.xml includes the song’s cover image, its title, subtitle and the view count.

If the section is another section or a category: Using the song\_list\_item \_recycler\_row.xml layout and the TrendingSongsAdapter for the list. The song\_list\_item\_recycler\_row.xml includes the song’s cover image, its title and subtitle.





1. **Player Activity**



1. **TextViews**

Display song title, album, and status (e.g., "Now Playing")

1. **ImageView and Gif**

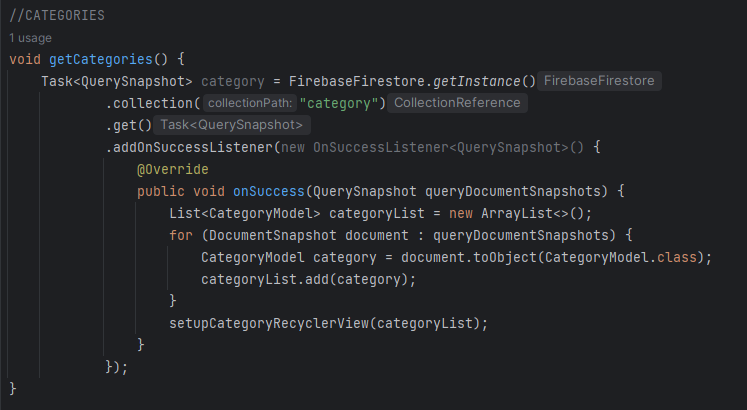
Show the song cover image and an optional gif animation.

1. **PlayerView**

PlayerView is a media playback control interface (MyExoPlayer).

1. **Backend Design**
2. **Query data from Firebase**

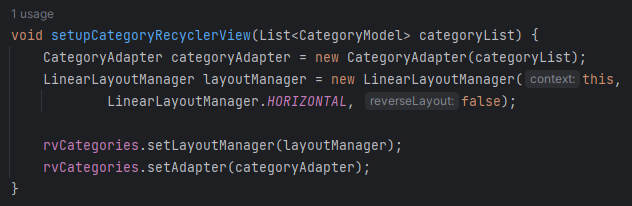
All the songs, categories, sections and user authentication is from Firebase database. Below is an example of query the categories (category name, cover url and list of songs) from the database:



After having a list of queried items, we need to make them appear in the Recycler View.

1. **Set up RecyclerView**

Below is an example of setting up the Category Recycler View:



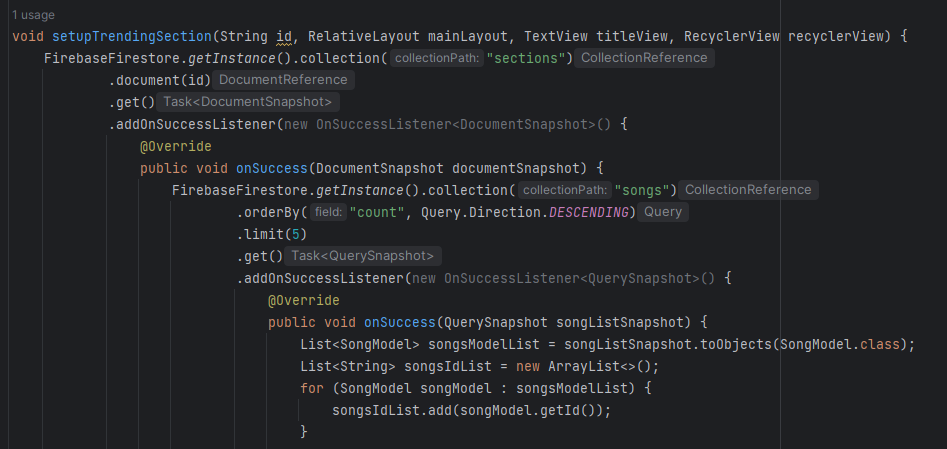
1. **Update Views Count**

Each time a song is played, this function will query the number of view count of this song from the database, add 1 more view to it and update the number of view count. If there is no view so far, the fuction will update it to 1.



1. **Set up Trending Section**

This function will first order the songs list in descending order, then take the top 5 songs and add into the query list. The remaining part is the same as III.2. Set up Recyler View.



1. **Filter content in SearchView**

This function will loop through the songs list and check whether the query text is in the song’s title or subtitle. If it is, that song will be added into the query list.



1. **Show Now Playing View**

This function will check whether there is a song playing. If there is, the player\_view View will set visible. If not, it will be set gone. If user clicks on it, the app will navigate to the Player Activity.



1. **Login**

This function uses a built-in method of FirebaseAuth class: signInWithEmail AndPassword(String email, String password). If the user is successfully logged in, the app will navigate to the Main Activity. If not, a Toast will appear to send failed notification.



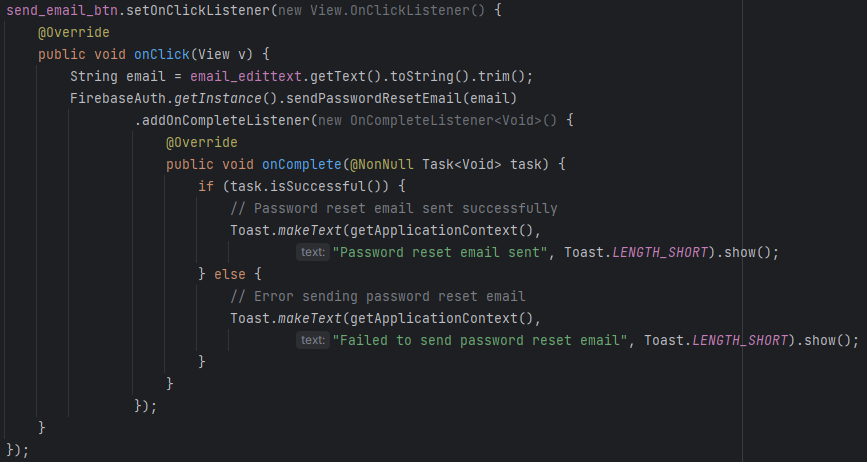
1. **Signup**

This function uses a built-in method of FirebaseAuth class: createUserWith EmailAndPassword(String email, String password). If the user is successfully signed in, the app will send a successful notification. If not, the app will send a failed notification.



1. **Send password reset email**

This function uses a built-in method of FirebaseAuth class: sendPassword ResetEmail(String email). If the mail is successfully sent, the app will send a successful notification. If not, the app will send a failed notification.



1. **Other Techniques**

When developing an application that relies on data from external sources such as databases or APIs, it's essential to integrate these data sources seamlessly.

1. **Firebase**

Firebase provides a suite of tools for app development, including authentication, real-time database, cloud storage, and more. To integrate Firebase Firestore and Firebase Authentication into your Android application, we need to add the appropriate dependencies to our build.gradle file:

implementation("com.google.firebase:firebase-firestore:25.0.0")

implementation("com.google.firebase:firebase-auth:23.0.0")

1. **MyExoPlayer**

MyExoPlayer is used to handle media playback in the application. We can use the URLs retrieved from Firebase Storage to stream media. To integrate the Media3 library, including ExoPlayer, into the Android project, we need to add the appropriate dependencies to our build.gradle file:

implementation("androidx.media3:media3-exoplayer:1.3.1")

implementation("androidx.media3:media3-exoplayer-dash:1.3.1")

implementation("androidx.media3:media3-ui:1.3.1")

1. **Conclusion and Development Direction**

In conclusion, the development of an online music player application for the Android platform presents an exciting opportunity to combine technical expertise with creative innovation. By focusing on educational goals and addressing the outlined requirements, we aim to create a robust and user-centric solution that redefines the music listening experience.

**Future Development Direction:**

**Enhanced User Experience:** Continuously gather user feedback and iterate on the application design to enhance usability and accessibility. Incorporate user-centric features such as personalized recommendations, social sharing options, and accessibility features to cater to diverse user preferences.

**Expanded Content Catalog:** Collaborate with content providers and licensors to expand the music catalog, offering users a wider selection of songs across various genres and languages. Implement features for user-generated content and community-driven playlists to foster engagement and diversity.

**Advanced Audio Features:** Explore advanced audio processing technologies to improve sound quality and enhance the listening experience. Integrate features such as equalizers, audio effects, and immersive audio formats to cater to audiophiles and music enthusiasts.

**Data Analytics and Personalization:** Leverage data analytics and machine learning algorithms to analyze user behavior and preferences. Implement personalized recommendations, curated playlists, and targeted content delivery to provide a tailored music experience for each user.