Tapify

"An innovative design to effortlessly organize, discover, and enjoy your music with a tap!"

Introduction

OUR ELEVATOR PITCH - "**For** music lovers **seeking** simplicity, **Tapify** is the effortless way to manage and enjoy your favorite tunes. **Tired of** cluttered interfaces and endless menus? **Our product** offers a minimalist design that simplifies the process, making it easy to find and enjoy music without frustration — because music should be easy to enjoy, not hard to find."

Whether trying to listen to a specific artist, album, or an entire discography, the multiple buttons and unnecessary options are cumbersome and a major buzzkill. Despite having hundreds of saved songs, the clutter and mismatched recommendations often leave users overwhelmed. This sparked our desire for a more streamlined and intuitive music selection experience.

Our project aims to develop an improved music player application to provide a faster, more efficient and concise music selection experience. Our team found that the current popular music players (such as Spotify, Poweramp and Foobar2000) have some shortcomings in user experience: for example, many functions are hidden under multiple clicks, the interface feels overly complicated, and managing the song list has become unnecessarily difficult. The goal is to create a music player with a simple interface and intuitive operation, which will reduce the operation steps for users to find and play music, thus improving the user experience.

This project will be mainly aimed at digital music lovers of all ages, especially those who want to find music quickly when commuting, studying or working. Our team's application will provide a variety of music browsing methods, including classification by artist, album, song list and recently played songs, to ensure that users can easily find the music they want to play. In addition, the optimized interface layout will make these common functions more prominent, reduce unnecessary clicks, and make users feel more convenient and comfortable when choosing music. At the same time, our team will focus on designing a flexible and efficient way of interaction, so that users can freely switch between different music selection methods, avoid information redundancy and ensure smooth operation.

In the design process, our team will strictly follow the principles of user experience and information design to ensure that the interface is intuitive, easy to understand and use. In order to ensure that the project meets the actual needs of users, our team will pass the test in the UIC campus. The team will also collect user feedback via an online music forum to collect opinions, in order to continuously optimize the design and function. Our team believes that this music player will provide users with a more efficient and convenient music listening experience by improving the interaction process and interface design. We hope that the application becomes an ideal choice for commuters and music lovers in the future.

Sketches and Diagrams

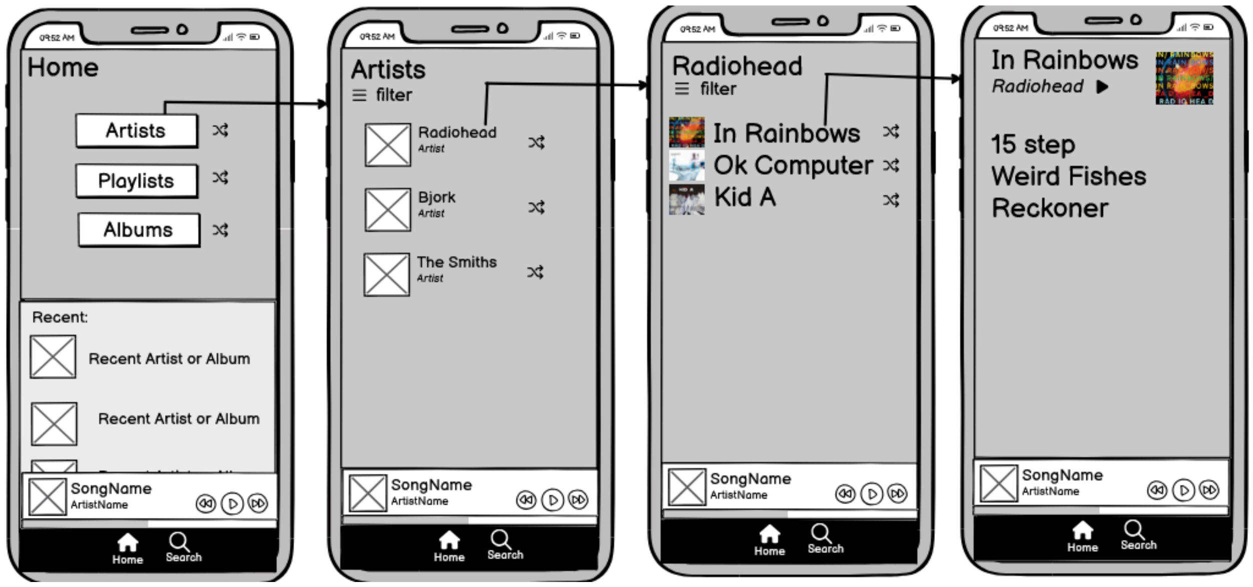
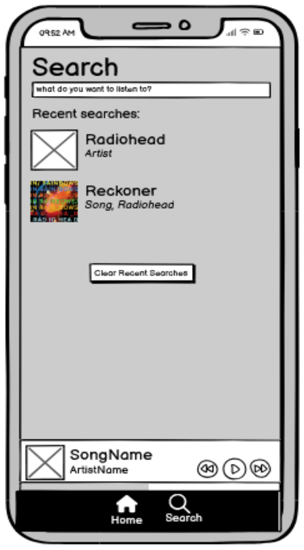


figure 1: proposed song selection transitions from the main page's artist button



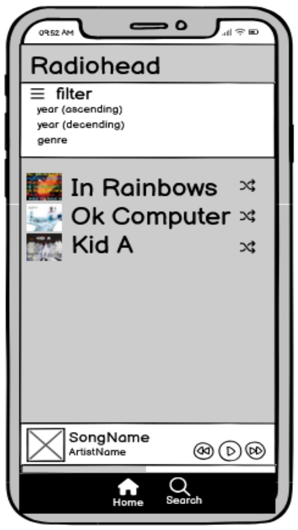


figure 2: proposed search page

figure 3: proposed filtering options

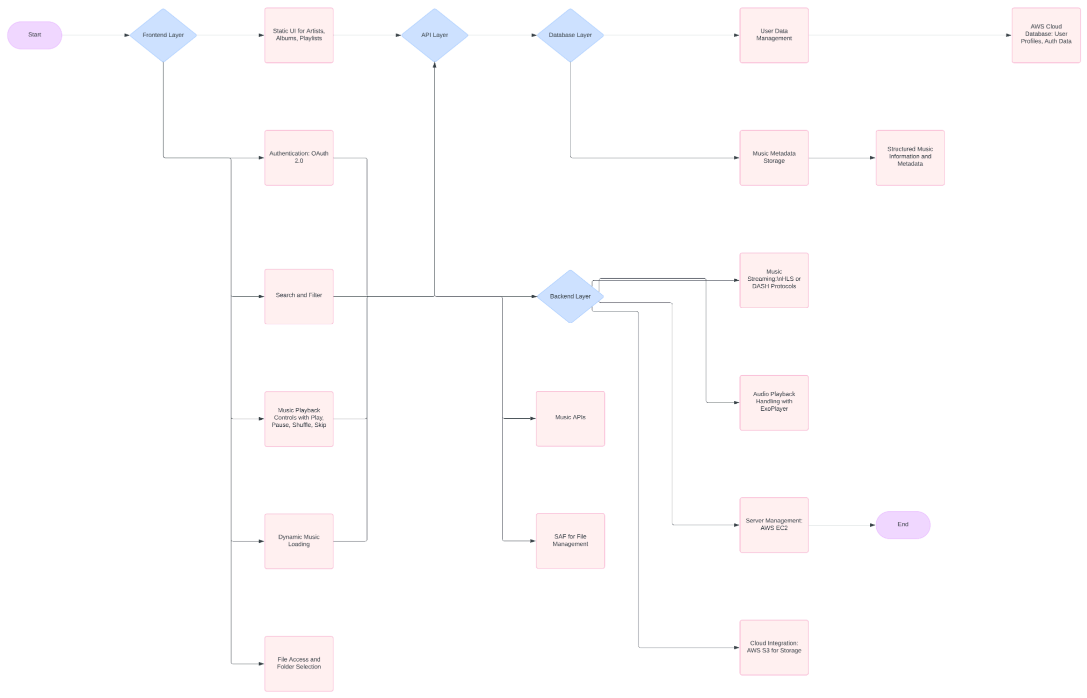

Start
Frontend Layer
Static UI for Artists, Albums, Playlists
API Layer
Database Layer
User Data Management
AWS Cloud Database: User Profiles, Auth Data
Authentication: OAuth 2.0
Music Metadata Storage
Structured Music Information and Metadata
Search and Filter
Backend Layer
Music Streaming:\nHLS or DASH Protocols
Music Playback Controls with Play, Pause, Shuffle, Skip
Music APIs
Audio Playback Handling with ExoPlayer
Dynamic Music Loading
SAF for File Management
Server Management: AWS EC2
End
File Access and Folder Selection
Cloud Integration: AWS S3 for Storage

Figure 4 : Software architecture diagram

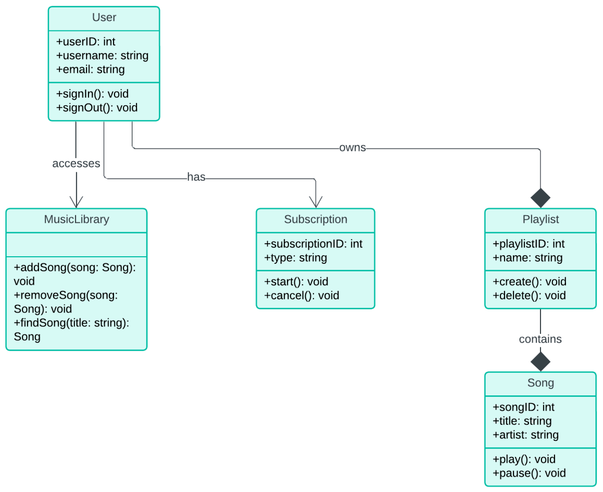


Figure 5: Initial database design

<u>User Needs</u>

The project aims to simplify the process of selecting and organizing music by designing an intuitive, decluttered user interface. Current music apps often present users with a complicated selection process and overwhelming song management options. This project seeks to resolve these issues, making ***music selection more efficient, visually streamlined, and easy to navigate,*** while providing users with multiple ways to play their music.

Target Users - The target users are individuals aged 13 and older who prefer listening to music digitally on mobile devices. This demographic includes casual listeners and dedicated music enthusiasts who value ease of use, quick access, and an organized music experience.

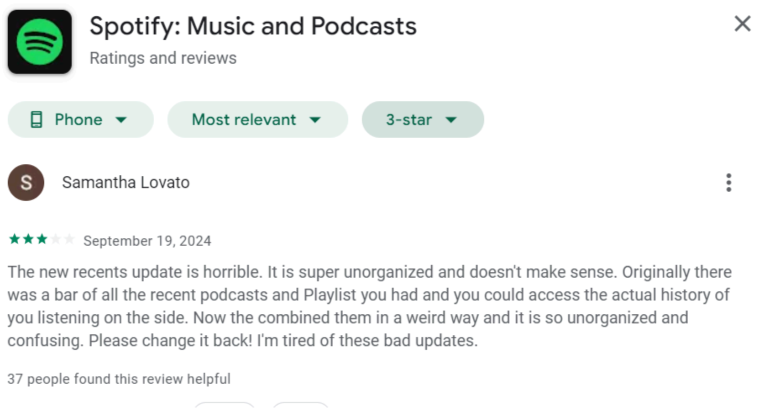
What Will Users Do?

Users will interact with the app by opening it, browsing their music library, and selecting what they want to play. *To gather feedback*, users will be recruited from the UIC campus, as well as through online recruitment platforms. After consenting to participate, they will engage in usability testing at various development stages.

During testing, users will interact with the app with minimal guidance, providing natural feedback on their experience. They will also be asked to verbalize their thought process and describe any challenges or positive impressions they have while navigating the app. This ***"think-aloud" method*** will be used to identify pain points, areas of improvement, and intuitive design elements. Data will be collected at key points during the development process to ensure the app meets the intended goals.

<u>Precedents</u>

The paper **"*MP3-music player application development*"** by Jia Tan published in Google Scholars does an analysis of popular music players and puts forth what features need to be discarded. The project proposed in the paper aimed to develop an application with access to a vast song library, download options, touch gestures, and overcoming the limitation of the order in which players can choose their songs.

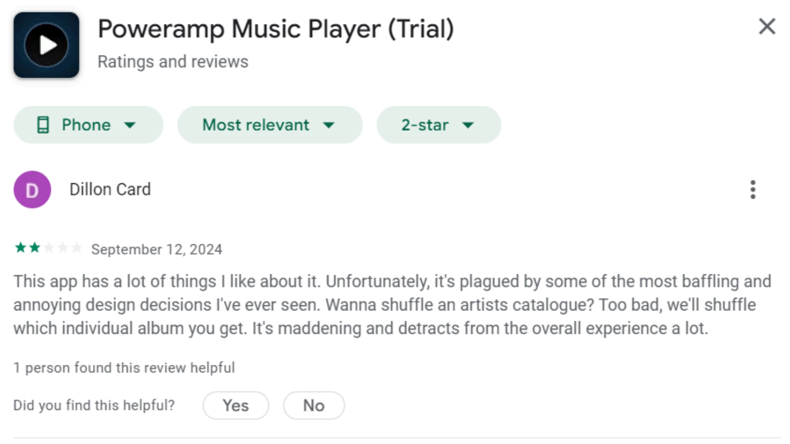


The new recents update is horrible. It is super unorganized and doesn't make sense. Originally there was a bar of all the recent podcasts and Playlist you had and you could access the actual history of you listening on the side. Now the combined them in a weird way and it is so unorganized and confusing. Please change it back! I'm tired of these bad updates.

37 people found this review helpful

Figure 5: Recent public review of Spotify on Playstore

Looking at Spotify, one of the most popular music streaming platforms, there are several reviews *(figure 5: example of such reviews)* that point out the tediousness of its user interface. To access a preferred artist's music, the user must first click on the 'Library' tab, then scroll through a small list of categories at the top of the screen to find the desired song. This extra navigation makes the experience less intuitive. This current design is not ideal since the main important aspects of the page are hidden, and the unimportant bits take up most of the space. Tapify intends to streamline access to frequently played content.


Poweramp Music Player (Trial)

Ratings and reviews

Dillon Card

September 12, 2024

This app has a lot of things I like about it. Unfortunately, it's plagued by some of the most baffling and annoying design decisions I've ever seen. Wanna shuffle an artists catalogue? Too bad, we'll shuffle which individual album you get. It's maddening and detracts from the overall experience a lot.

1 person found this review helpful

Figure 6: Recent public review of Poweramp on Play Store

Lastly, we looked into a music app named Poweramp, which was also found through searching for music players in the Android app store. The user can open the app to see different categories such as ‘Folders’, ‘artists’, ‘albums’, ‘genres’, but we felt that this was too much information on the screen that may confuse the user, and lacks the ability to shuffle at any stage of the selection, so we reduced the amount of options to select, and added a shuffle button on each screen.

Ingredients

Development will target various Android devices to ensure compatibility across different screen sizes and hardware configurations. Testing will be conducted primarily on devices running Android 9 (Pie) and higher to take advantage of the latest Android features and APIs.

UI Development

- Programming Language: Java for Android development.
- Frameworks/Libraries:
    - **Android Jetpack**: Components like Navigation, LiveData, and ViewModel for better app architecture. Research from Android.com indicated that “Jetpack Compose is Android’s recommended modern toolkit for building native UI. It simplifies and accelerates UI development on Android” [4], especially in a music player, where playback controls and user data change frequently.
    - **ExoPlayer**: For handling audio playback, providing better control and features compared to the default MediaPlayer.
- Design Tools: **Figma** or **Adobe XD** for creating mockups and prototypes.
- User Testing: Tools like **Maze** for gathering feedback on the user experience.

Backend Development

- Server-Side Language: Node.js, Python (Django or Flask), or Java (Spring Boot)
- Database: MySQL for structured data/AWS cloud database *(to be decided)*
- Cloud Services: AWS (S3 for storage, EC2 for server hosting)

- Streaming Protocols: HLS (HTTP Live Streaming) or DASH (Dynamic Adaptive Streaming over HTTP) for delivering audio streams.

APIs

- Music APIs: Use services like Spotify API, Apple Music APIs
- Authentication: OAuth 2.0 for user authentication
- **Android Storage Access Framework (SAF):** This API will be used to manage and access music files from the user's local storage. The SAF enables applications to read, play, and manage audio files securely while preserving user privacy by limiting access to necessary directories only.

Development Tools

- IDE: Android Studio for Android app development.
- Version Control: Git for codebase management
- Build Tools: Maven for building your Android app.

Testing and Deployment

- Analytics: Firebase Analytics or Google Analytics for user behavior tracking.
- Deployment: AWS Deployment

Innovation

1. Streamlined Navigation - Tapify's UI reduces operational steps, enabling users to browse music through artists, albums, playlists, and recently played songs with fewer taps. No more digging through convoluted menus.
2. Decluttered Interface - By eliminating unnecessary options, Tapify creates a clean interface where the focus is on enjoying music, not wrestling with app mechanics.
3. Efficient Interaction Design - Tapify introduces a flexible music selection method, allowing users to switch effortlessly between different ways of accessing their library—whether they want to listen to a favorite artist or explore an entire discography.
4. User-Centered Development - With real feedback from UIC campus testers and music forums, Tapify will continuously evolve to meet the needs of digital music lovers.

Some other features to enhance Tapify if time permits -

1. Dynamic Suggestions Based on Time/Activity: Use machine learning to recommend music based on the user’s routine. For instance, upbeat tracks in the morning, relaxing tunes in the evening, or study music during working hours.
2. Smart Widgets: Allow users to create custom quick-access panels on the app’s home screen for their favorite artists, albums, or playlists.
3. Lyrics Search Integration: Users can find songs by typing a lyric they remember, even if they don’t know the title or artist.

Project Plan / Scope:

Milestone	**Task/Goal**	**Deadline**
Week 1-2	Define requirements and finalize project plan	2nd week
Week 2-4	Develop low-fidelity prototype	4th week
Week 4-5	Conduct user testing on low-fidelity prototype	5th week
Week 5-7	Build first functional prototype	7th week
Week 7-8	User testing on functional prototype	8th week
Week 9	Final functional prototype + presentation prep	9th week
Week 10	Final presentation and project submission	Final week

1. User Interface (UI) and Navigation

- Components:
    - Static UI elements for Artists, Albums, Playlists, and Recently Played sections.
    - Button navigation between pages (e.g., from Artists to Albums).
    - Search bar for easy access to music.
    - Filter menu to organize music by categories (artist, album, etc.).
    - Basic settings for selecting music folder location and managing permissions.
- Responsible Team Member - Design and develop the UI, ensuring smooth transitions between sections. Integrate the search bar and filter menu for efficient navigation and music discovery, focusing on a clean and intuitive user experience.

2. Music Playback and Controls

- Components:
    - Play, Pause, Shuffle, and Skip buttons, located at the bottom of each page for the last played, or currently playing song.
    - Progress bar displaying the current song's position.
    - Contextual playback based on user selection (e.g., playing from an album, playlist, or artist).
    - File access permissions for music folder selection.
- Responsible Team Member - Implement music playback controls using ExoPlayer, ensuring that users can play, pause, shuffle, and skip tracks seamlessly. Manage the progress bar display during playback and handle file access permissions to allow users to select their music folder.

3. Music Loading and Organization

- Components:

    - Load and display music data in Artists, Albums, and Playlists sections, including album art and artist details.
    - Ensure selected music plays correctly through the playback controls.
- Responsible - Manage the loading of music data into the app's sections, ensuring that album art, song titles, and artist details are correctly displayed.Link the music selections with playback controls, ensuring the shuffle function works correctly across sections.

4. System Integration and Testing

- Components:
    - Integration of UI elements with playback functionality and dynamic music loading.
    - Testing of navigation, playback, shuffle, and search functionality.
    - User feedback collection for refinement.
- Responsible Team Member - Oversee the integration of UI, playback, and music loading features. Conduct testing to ensure the system works smoothly and gather user feedback for improvements.

System Architecture

The Tapify system will focus on client-side functionality. Key components include:

1. User Interface (UI): Built using Jetpack Compose or XML for handling static pages like Artists, Albums, Playlists, and Recently Played, with smooth navigation between them.
2. Local Music Access: Managed using Android's Storage Access Framework (SAF) to access and organize music files from the user's device.
3. Playback Engine: ExoPlayer will manage all music playback actions, such as play, pause, shuffle, and skip, and update the progress bar.
4. Data Persistence: SharedPreferences will store user preferences (e.g., folder location, shuffle settings) and recently played songs.

This architecture ensures Tapify functions entirely offline, with no need for server-side components.

Scope of Work

1. **Static UI and Navigation**: Implement static UI for Artists, Albums, Playlists, and Recently Played sections. Users will be able to navigate between these sections via button interactions.
2. **Music Playback Controls**: Play, Pause, Shuffle, and Skip buttons integrated with ExoPlayer, providing smooth playback. The progress bar will display song progress.
3. **Dynamic Music Loading**: Album art, artist information, and song details will be dynamically loaded into the relevant sections.
4. **Search and Filter**: Users can search for specific songs or artists and filter content by category.
5. **File Access and Folder Selection**: The app will request file access permissions and allow users to select or change their music folder location.

Task Assignment Overview

UI/UX Development

- Designs and implements static UI elements and navigation across Artists, Albums, Playlists, and Recently Played sections. Integrates search and filter functionality.

Music Playback and File Access

- Implements core playback functionality using ExoPlayer, managing play, pause, shuffle, and skip actions. Ensures the progress bar works and handles file access permissions for music folder selection.

Music Loading and Organization

- Manages dynamic loading of music data into the Artists, Albums, and Playlists sections. Ensures album art and song details are correctly displayed, with music playing according to the user's selection.

System Integration and Testing

- Oversees system integration to ensure smooth functionality across all components. Conducts testing and gathers user feedback for app optimization.

References

[1] Tan, Jia Yi. 2022. MP3-music player application development. Diss. UTAR. (Google Scholar)

[2] Spotify: Music and Podcasts, Retrieved September 21st 2024 from https://play.google.com/store/apps/details?id=com.spotify.music&hl=en_US&pli=1

[3] Poweramp Music Player, Retrieved September 21st 2024 from https://play.google.com/store/apps/details?id=com.maxmpz.audioplayer&hl=en_US

[4] Jetpack Compose UI App Development Toolkit, Retrieved September 22nd 2024 from https://developer.android.com/compose

[5] Wang, L. (2014, July 30). I Love You, Spotify … Now Change. UX Magazine. https://uxmag.com/articles/i-love-you-spotify-now-change