

1. Introduction

1.1. Research Background

1.1.1. Integration of Music Streaming and Karaoke Applications

The rapid growth of music streaming applications has reformed how users access and consume music. Applications like Spotify, Apple Music, and YouTube Music offer a wide variety of music libraries and personalized recommendations. But those music streaming applications focus only on the passive listening experience. On the other hand, karaoke applications such as Smule and StarMaker provide an interactive karaoke experience, but they lack a comprehensive music streaming functionality. This contradiction indicates a gap in the world for an application that combines music streaming capabilities with interactive karaoke features. By addressing this gap it offers a more comprehensive musical experience for the users.

1.1.2. Social Interaction in Music Applications

One of the most important aspects in music discovery and user engagement within digital platforms is social interaction. Studies have shown that users' musical preferences and exploration behaviors are mostly influenced by their social environments. One of the studies, Di Bona et al, demonstrated that social interactions affect the discovery processes, with users showing a different kind of taste in music, which is influenced by their social group's taste in music. By implementing social features into music applications can enhance user engagement and enable users to discover different tastes of music through their friends' influence on the taste of music they hear. [1]

1.1.3. Technological Advancements in Mobile Music Applications

Technologies like Kotlin and Firebase have significantly enhanced the development of mobile applications. Kotlin is a modern programming language that is really suitable for creating an Android Application because of its simple syntax and compatibility with the Java programming language. Real-time databases, authentication services, and cloud storage are just

some of the capabilities that Firebase offers to help developers develop responsive and scalable applications. In contrast to the traditional relational databases, Khawas and Shah emphasized Firebase's speed and efficiency in managing large amounts of unstructured data in Android applications. By utilizing these technologies can improve the functionality of the music applications and accelerate the development process. [2]

1.1.4. Enhancing User Engagement through Interactive Features

Social sharing options and karaoke capabilities are the examples of interactive features that can significantly enhance user engagement in music applications. In Hwang and Oh's study, they examined how interactive music affected consumer perception and behavioral intention. They discovered that interactive music elements improved user engagement and positively influenced users' opinions of the platform. By enabling users to actively contribute in the music creating and sharing, applications can create a more pleasing and personalized user experience. [3]

1.1.5. Addressing the Gap in Current Music Applications

Although the technologies in karaoke and music streaming applications nowadays have advanced, there remains a lack of platforms that successfully combine these features with robust social sharing capabilities. By implementing social interaction, karaoke features, and music playback onto a single platform, the proposed applications seek to address this gap. The application will offer a smooth and engaging user experience by utilizing Kotlin and Firebase, allowing users to listen to music, perform karaoke, and share their performances with others.

1.2. Problem Identification

Music plays an essential role in everyday life, serving as a medium for relaxation, emotional expression, and social interaction. However, current digital music platforms often segment the user experience between passive listening and active participation. While music streaming services like Spotify or Apple Music offer extensive music libraries for listening, they generally lack interactive features such as karaoke or performance sharing capabilities. This separation limits the user experience for those who wish to engage more actively with music beyond just passive consumption. [4]

Karaoke applications, on the other hand, provide users with interactive musical experiences but frequently lack the integration of high-quality music streaming and personalized playlist creation. This results in fragmented user journeys where individuals must rely on multiple applications to fulfill their listening and performance needs. Additionally, most karaoke platforms are limited in social connectivity features beyond simple performance uploads, missing opportunities for deeper engagement and community building among users. [5]

The importance of social interaction in music discovery and enjoyment has been highlighted in recent studies. Research by Park et al. (2020) showed that social engagement through music applications enhances user satisfaction and increases music discovery through peer influence. Despite these findings, few applications have succeeded in integrating social sharing, personalized playlists, and karaoke performance into a single cohesive platform. [6]

From a technological perspective, advancements in mobile development tools like Kotlin and Firebase have enabled faster, more efficient development of feature-rich applications. However, few current solutions leverage these technologies to create an all-in-one music experience that merges streaming, karaoke, and social networking. This indicates a missed opportunity in the current digital music ecosystem. [7]

Therefore, the lack of a unified platform that combines robust music playback, karaoke interaction, and social sharing features highlights a clear gap in the market. This thesis addresses this gap by proposing the development of a mobile application that integrates these three essential elements. By doing so, it aims to provide users with a seamless and enriched music engagement experience.

1.3. Research Status

1.3.1. Research Scope

This thesis aims to develop and implement an end-to-end mobile app that integrates music streaming, karaoke capabilities, and social sharing in a single application. The client on Android is developed using Kotlin, which provides the benefits of expressive syntax, null safety, and coroutine-based asynchronous programming. Firebase provides the application's backend side, including user authentication management, real-time data synchronization using Cloud Firestore, and secure media asset storage in Cloud Storage, to enable rapid development, robust security, and scalability as the application's user base increases.

At the core of the application, the app incorporates a comprehensive music playback system. Users can browse and stream from the library of tracks, create personalized playlists, and control playback with familiar media controls (play/pause, fast forward, and rewind). The karaoke part of the application overlays time-coded lyrics on instrumental or vocal-stripped tracks, captures live microphone input in sync with background music, and automatically mixes the resulting vocal recording with the original track. A combined preview interface allows listeners to hear or re-record their performances before sharing.

To encourage community activity, social sharing functionality includes the upload of mixed audio files, commenting, and liking friends' entries. All of the user data and metadata which includes recordings, images, playlist structures, and engagement metrics, are stored in Firebase Cloud Storage and Firestore. This real-time synchronization makes sure that the changes are replicated instantly throughout devices.

1.3.2. Research Benefit

By unifying music streaming, karaoke performance, and social sharing within a single mobile application, this research delivers a significantly enhanced user experience. Instead of toggling between separate apps for passive listening, live singing, and social interaction, users will enjoy a seamless transition from browsing and playing tracks to recording their own performances and sharing them with friends. This unified process not only minimizes the effort

but also enables more engaging interaction with music content, with consumers able to flip from consumption to creation of music and vice versa in an instant.

The incorporation of social features also renders the platform more attractive by enabling real-time music-based connections. With in-app commenting and liking other people's posts, users get actively involved with each other's creative process, making karaoke sessions that would otherwise be solitary activities into a shared environment. The social aspect not only maintains long-term user engagement but also creates a healthy community where musical discovery and peer review become mutual activities.

From a technical standpoint, the project demonstrates the effective synergy of modern development tools (Kotlin) for expressive, safe client-side code and Firebase for a scalable, secure backend. By showcasing how coroutines, real-time databases, and cloud storage can be combined together to support complex media workflows, this thesis contributes a practical blueprint for next-generation multimedia applications. In addition, by using Cloudinary media delivery via CDN and ExoPlayer for streaming performance, the research recognizes architecture patterns to guide the future development of socially interactive entertainment apps.

Lastly, by providing karaoke and music discovery in one application, the app extends to users with different technical expertise and musical talent. The intuitive interface and integrated help tutorials lower the learning barrier for newcomers, while capabilities like playlist management can satisfy users in hearing the music that they prefer. Thus, this project offers a landing stone to further explore how creative expression, community building, and technology can mix to reshape the mobile music experience.

1.4. Literature Review

This section surveys existing music, karaoke, and hybrid applications that focus on their core features and social capabilities to create the proposed karaoke-integrated music app within the broader landscape.

1.4.1. Music Streaming Applications

Music streaming services such as Spotify, YouTube Music, and Apple Music have revolutionized how users discover and listen to music, offering vast varieties of music, personalized recommendations, and social sharing features like collaborative playlists and activity feeds. However, these platforms are optimized for passive listening and do not provide any built-in tools for user performance recording or karaoke functionality.

1.4.2. Karaoke-focused Applications

Karaoke applications like Smule and StarMaker provide feature-rich interactive singing experiences, with real-time pitch correction, duet modes, and voice effects. While they prioritize performance, they are based on licensed karaoke songs and don't typically feature a rich streaming catalog or advanced playlist management. Social sharing is supported—users can share recordings, like, and comment—but integration with third-party streaming libraries is weak.

1.4.3. Software Engineering Perspective

From the software engineering perspective, current applications either adhere to a monolithic mobile architecture (Smule/StarMaker) or a microservices-supported streaming paradigm (Spotify). They largely utilize native development stacks (Android/Java or Kotlin, iOS/Swift) and RESTful APIs, with cloud providers used for storage and user management. None integrate real-time media mixing (karaoke recording) and large-scale streaming within a single unified platform, which needs a definite opportunity for the proposed combined solution.

1.4.4. Feature Comparison of Representative Applications

Table 1.1

Application	Streaming Catalog	Karaoke Recording	Social Feed (Likes/Comments)	Personalized Recommendations
Spotify	✓	✗	Collaborative playlists	✓
Smule	✗	✓	Likes, comments, duets	✗
StarMaker	✗	✓	Likes, comments, challenges	✗
Proposed App	✓	✓	✓	✗

1.5. Development Schedule

No	Phase	Duration in weeks												
		1	2	3	4	5	6	7	8	9	10	11	12	13
1	Requirement Analysis	■	■											
2	Design		■	■	■	■								
3	Implementation				■	■	■	■	■	■	■	■		
4	Evaluation & Testing											■	■	
5	Finalization													■