

Rapport du Projet Application Music

Réalisé par : EL KASSY ANASS
CHRIKI MARIAM
SOUHAIL WAFDI
LOKMAN TAHDI

DEV MOBILE
Année Académique : 2025–2026

Tableau De Matière

Introduction Générale.....	4
Chapitre 1 : Contexte Général du Projet.....	5
1.1 Introduction	5
1.2 Description du Projet.....	5
Vision	5
Objectifs	5
Périmètre fonctionnel.....	6
Utilisateurs cibles	6
1.3 État de l'existant	6
1.4 Problématique	6
1.5 Solution proposée	7
1.6 Objectifs	7
Fonctionnels	7
Techniques.....	7
1.7 Conclusion du chapitre	7
Chapitre 2 : Étude, Analyse et Conception	8
2.1 Cahier des charges	8
2.1.1 Spécifications fonctionnelles	8
2.1.2 Spécifications techniques	8
2.1.3 Choix de la base de données	8
2.2 Analyse des flux	9
2.2.1 Flux d'authentification	9
2.2.2 Flux musical	9
2.2.3 Flux de données	9
2.3 Conception	10
2.3.1 Diagramme de cas d'utilisation	10
2.3.2 Diagramme de classes	11
2.3.3 Diagramme de séquence.....	11
Chapitre 3 : Étude Technique.....	12
3.1 Architecture	12
3.1.1 Arborescence du projet.....	12
3.2 Choix des outils	13
3.3 Sécurité	13
Chapitre 4 : Réalisation	14
4.1 Écrans et fonctionnalités	14
4.1.1 Écran d'authentification – Connexion.....	14
Figure 4.1 – Interface de connexion de l'application React Native Music App....	15
4.1.2 Écran d'inscription	15
Figure 4.2 – Interface d'inscription de l'application React Native Music App	16
4.1.3 Écran principal – Liste des morceaux	16
Figure 4.3 – Interface de la bibliothèque musicale	17
4.1.4 Écran de lecture audio.....	17

Figure 4.4 – Interface du lecteur audio.....	18
4.1.5 Écran de gestion des playlists	18
Figure 4.5 – Interface de gestion des playlists	19
4.2 Analyse globale de la réalisation	19
4.3 Conclusion du chapitre	19
Chapitre 6 : Difficultés rencontrées et solutions	20
Conclusion Générale	21

Introduction Générale

Avec l'évolution rapide des technologies mobiles et la démocratisation des smartphones, les applications de streaming musical occupent aujourd'hui une place essentielle dans le quotidien des utilisateurs. Ces derniers recherchent des solutions capables d'offrir une **expérience fluide, rapide et accessible à tout moment**, tout en garantissant une navigation simple et une interface ergonomique. La musique étant devenue un contenu largement consommé sur mobile, le besoin d'applications performantes et intuitives ne cesse de croître.

Dans ce contexte, le projet **React Native Music App** a pour objectif de concevoir et de développer une **application mobile de streaming musical multiplateforme**, compatible avec les systèmes **Android et iOS**. L'application permet aux utilisateurs d'accéder à une bibliothèque musicale, d'écouter des morceaux audio et d'interagir avec les différentes fonctionnalités à travers une interface moderne, claire et facile à utiliser.

L'application est développée à l'aide du framework **React Native**, associé à l'environnement **Expo**, afin de faciliter le développement multiplateforme à partir d'un seul code source. Pour la gestion des utilisateurs, de l'authentification et des données, le projet s'appuie sur la plateforme **Firebase**, qui offre des services backend sécurisés et performants, adaptés aux applications mobiles modernes.

Ce rapport a pour but de présenter l'ensemble des étapes de réalisation du projet. Il décrit d'abord le **contexte général et les objectifs**, puis détaille les **besoins fonctionnels et techniques**, la **conception du système**, l'**architecture adoptée**, ainsi que la **réalisation de l'application**. Enfin, les difficultés rencontrées lors du développement et les solutions apportées seront exposées, avant de conclure par une synthèse du travail réalisé et les perspectives d'évolution futures du projet.

Chapitre 1 : Contexte Général du Projet

1.1 Introduction

Ce chapitre présente le cadre général du projet et les motivations ayant conduit à la réalisation d'une application mobile de streaming musical. L'objectif est de proposer une solution simple permettant aux utilisateurs d'accéder à des contenus musicaux, de gérer leurs playlists et de profiter d'une expérience fluide sur mobile.

1.2 Description du Projet

Vision

L'objectif principal est de développer une application mobile de musique permettant :

- la consultation d'une bibliothèque musicale,
- la lecture de morceaux audio,
- la gestion de playlists personnalisées,
- l'authentification des utilisateurs.

L'application vise à offrir une expérience utilisateur fluide et ergonomique, tout en garantissant la sécurité et la persistance des données.

Objectifs

Les objectifs du projet sont :

- Permettre l'écoute de musique via une application mobile multiplateforme.
- Offrir une interface simple et intuitive.
- Gérer les utilisateurs via un système d'authentification sécurisé.
- Stocker et récupérer les données musicales via le cloud.
- Assurer une navigation fluide entre les écrans.

Périmètre fonctionnel

Le périmètre fonctionnel de l'application comprend :

- Authentification des utilisateurs (inscription, connexion, déconnexion).
- Affichage de la liste des morceaux musicaux.
- Lecture audio (play, pause, changement de piste).
- Gestion des playlists.
- Navigation entre les écrans principaux.
- Gestion des paramètres de l'application.

Utilisateurs cibles

Cette application s'adresse à toute personne souhaitant écouter de la musique depuis son smartphone, sans nécessiter de connaissances techniques particulières.

1.3 État de l'existant

Plusieurs applications de streaming musical existent déjà (Spotify, Deezer, SoundCloud). Cependant :

- elles sont souvent complexes pour un usage basique,
- certaines fonctionnalités sont payantes,
- leur architecture est opaque pour un projet pédagogique.

Le projet **React Native Music App** se positionne comme une solution simple et éducative, adaptée à un contexte académique.

1.4 Problématique

Comment concevoir une application mobile de musique :

- simple à utiliser,
- performante,
- sécurisée,

- compatible Android et iOS,
- intégrant une gestion utilisateur et une lecture audio efficace ?

1.5 Solution proposée

- **Frontend mobile** : React Native avec Expo
- **Backend** : Firebase
- **Authentification** : Firebase Authentication
- **Base de données** : Firebase Realtime Database / Firestore
- **Navigation** : React Navigation
- **Lecture audio** : Expo AV (Audio)

1.6 Objectifs

Fonctionnels

- Lecture et gestion de musique ;
- Gestion des utilisateurs ;
- Navigation fluide.

Techniques

- Architecture basée sur les composants ;
- Intégration Firebase ;
- Code maintenable et évolutif.

1.7 Conclusion du chapitre

Ce chapitre a permis de présenter le contexte, les objectifs et la solution globale retenue pour le projet. Les chapitres suivants détaillent l'analyse et la conception du système.

Chapitre 2 : Étude, Analyse et Conception

2.1 Cahier des charges

2.1.1 Spécifications fonctionnelles

- Authentification utilisateur ;
- Lecture audio ;
- Gestion des playlists ;
- Navigation entre écrans ;
- Affichage dynamique des données.

2.1.2 Spécifications techniques

- React Native + Expo ;
- Firebase SDK ;
- JavaScript ;
- React Navigation ;
- Expo Audio.

2.1.3 Choix de la base de données

Firebase est retenu pour :

- sa facilité d'intégration ;
- sa synchronisation en temps réel ;
- sa sécurité ;
- son adaptation aux applications mobiles.

2.2 Analyse des flux

2.2.1 Flux d'authentification

- Inscription / Connexion via Firebase Authentication ;
- Gestion de session ;
- Redirection selon l'état utilisateur.

2.2.2 Flux musical

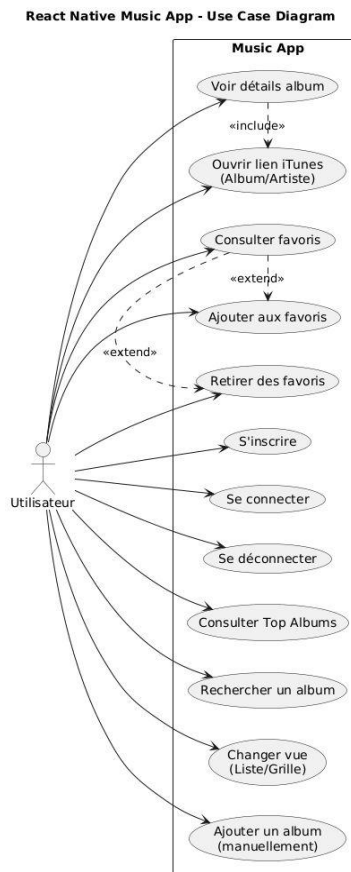
- Récupération des morceaux depuis Firebase ;
- Lecture via Expo Audio ;
- Contrôle de lecture.

2.2.3 Flux de données

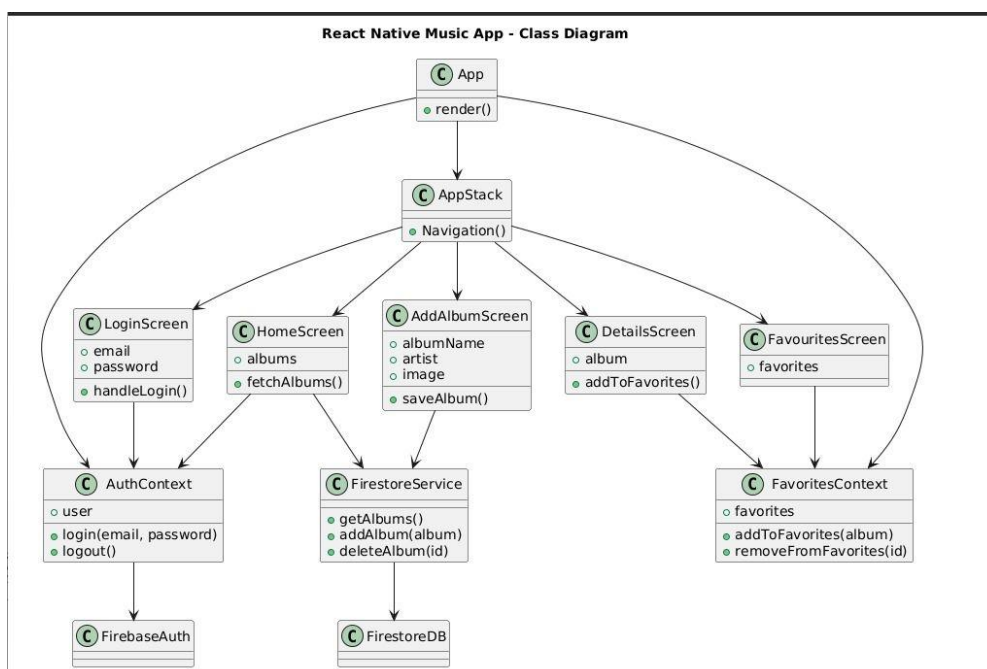
- Chargement des données au démarrage ;
- Mise à jour dynamique de l'interface.

2.3 Conception

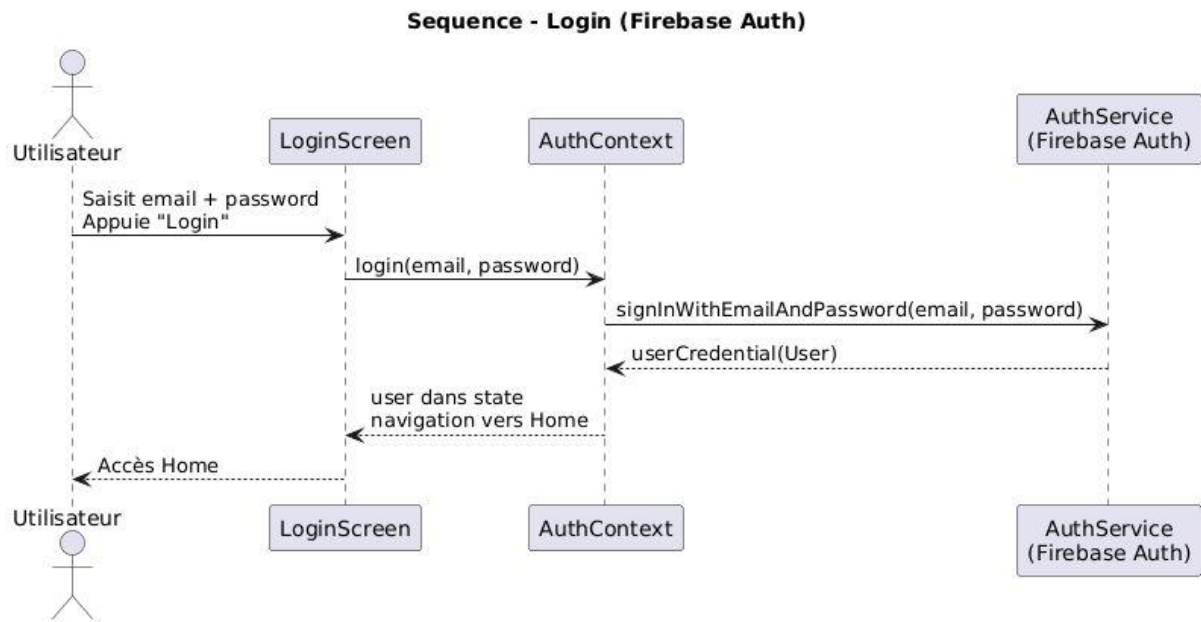
2.3.1 Diagramme de cas d'utilisation



2.3.2 Diagramme de classes



2.3.3 Diagramme de séquence



Chapitre 3 : Étude Technique

3.1 Architecture

L'application **React Native Music App** repose sur une architecture modulaire et bien organisée, basée sur les bonnes pratiques du développement mobile avec React Native. Cette architecture facilite la maintenance, l'évolutivité et la lisibilité du code.

L'architecture adoptée s'appuie principalement sur :

- **Les composants React :**
L'interface utilisateur est découpée en composants réutilisables, ce qui permet d'éviter la duplication du code et de simplifier les mises à jour futures.
- **La séparation entre l'interface utilisateur et la logique métier :**
La logique applicative (gestion des données, appels Firebase, lecture audio) est isolée dans des services dédiés, tandis que les écrans se concentrent sur l'affichage et l'interaction avec l'utilisateur.
- **Les services Firebase :**
Firebase est utilisé comme backend pour gérer l'authentification des utilisateurs et la récupération des données, assurant ainsi une communication fiable entre l'application et le cloud.

3.1.1 Arborescence du projet

L'organisation du projet est structurée de manière claire afin de faciliter la navigation dans le code source :

```

react-native-music-app-main/
├── App.js
├── src/
│   ├── components/
│   ├── screens/
│   ├── navigation/
│   ├── services/
│   ├── context/
│   ├── firebase.js
│   └── assets/

```

3.2 Choix des outils

Le choix des technologies utilisées a été guidé par des critères de simplicité, de performance et de compatibilité multiplateforme :

- **React Native :** permet de développer une application mobile Android et iOS à partir d'un seul code source, réduisant ainsi le temps et le coût de développement.
- **Expo :** facilite le développement avec React Native en fournissant un environnement prêt à l'emploi et des bibliothèques intégrées, notamment pour la gestion audio.

- **Firestore** : utilisé comme backend sécurisé pour l'authentification des utilisateurs et la gestion des données, sans nécessiter la mise en place d'un serveur dédié.
- **React Navigation** : assure une navigation fluide et intuitive entre les différents écrans de l'application.

3.3 Sécurité

La sécurité des données utilisateur constitue un aspect essentiel de l'application. Plusieurs mécanismes ont été mis en place :

- **Authentification Firebase** :
Chaque utilisateur doit être authentifié afin d'accéder aux fonctionnalités de l'application.
- **Règles de sécurité Firebase** :
Des règles spécifiques limitent l'accès aux données uniquement aux utilisateurs autorisés.
- **Accès restreint aux données utilisateur** :
Les données sont associées à l'utilisateur connecté, garantissant la confidentialité et l'intégrité des informations.

Chapitre 4 : Réalisation

4.1 Écrans et fonctionnalités

Ce chapitre présente les interfaces effectivement implémentées dans l'application **React Native Music App**, telles qu'illustrées par les **cinq captures d'écran** fournies dans le fichier de documentation. Chaque interface correspond à une fonctionnalité précise de l'application.

4.1.1 Écran d'authentification – Connexion

La première interface correspond à l'écran de **connexion** de l'application.

Fonctionnalités :

- Saisie de l'adresse email et du mot de passe.
- Authentification sécurisée via **Firestore Authentication**.
- Vérification des identifiants utilisateur.
- Affichage de messages d'erreur en cas d'échec de connexion.

Cet écran constitue le point d'entrée principal de l'application et garantit un accès sécurisé aux fonctionnalités musicales.

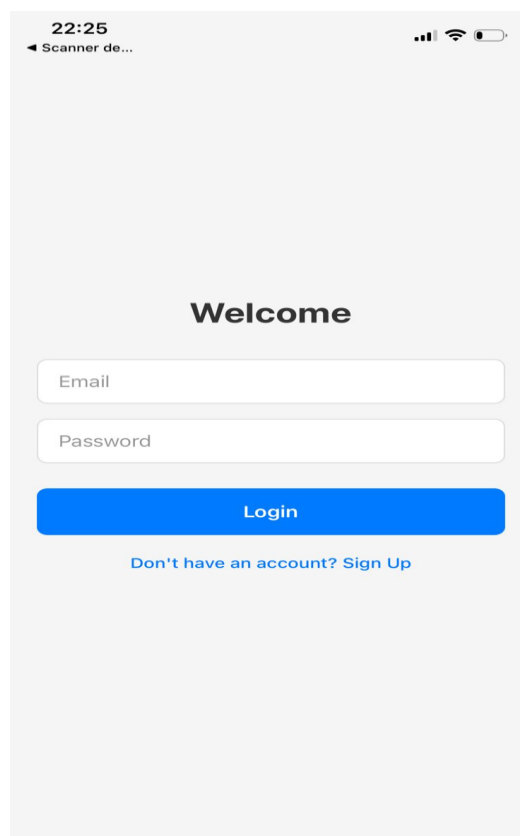


Figure 4.1 – Interface de connexion de l'application React Native Music App

4.1.2 Écran d'inscription

La deuxième capture illustre l'écran d'**inscription** permettant la création d'un nouveau compte utilisateur.

Fonctionnalités :

- Création d'un compte via email et mot de passe.
- Validation des champs saisis.
- Enregistrement du nouvel utilisateur dans Firebase Authentication.
- Redirection automatique vers l'application après une inscription réussie.

Cette interface permet d'élargir l'accès à l'application à de nouveaux utilisateurs.

Figure 4.2 – Interface d'inscription de l'application React Native Music App

4.1.3 Écran principal – Liste des morceaux

La troisième interface représente l'écran principal de l'application, affichant la **liste des morceaux musicaux disponibles**.

Fonctionnalités :

- Chargement dynamique des morceaux depuis Firebase.
- Affichage des informations principales (titre, artiste, image associée).

- Sélection d'un morceau pour lancer la lecture.
- Interface claire et ergonomique facilitant la navigation.

Cet écran constitue le cœur fonctionnel de l'application.

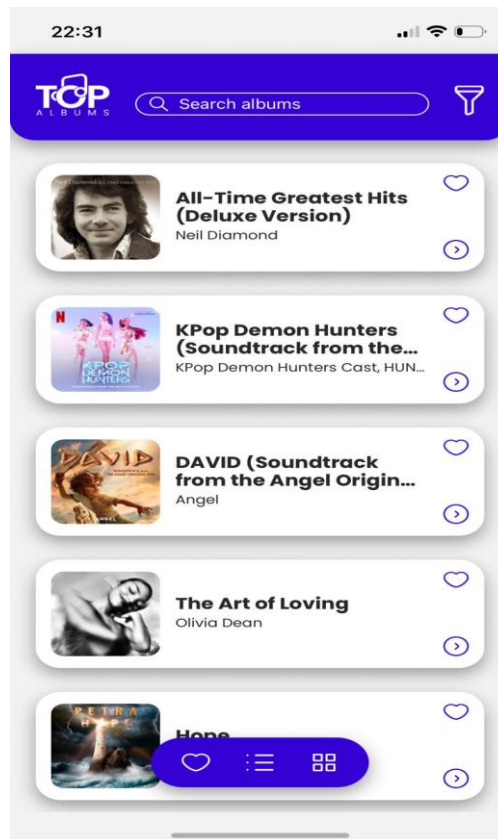


Figure 4.3 – Interface de la bibliothèque musicale

4.1.4 Écran de lecture audio

La quatrième capture correspond à l'écran du **lecteur audio**.

Fonctionnalités :

- Lecture des fichiers audio via **Expo AV (Audio)**.
- Commandes de lecture : lecture et pause.
- Affichage de l'état du morceau en cours.
- Interaction fluide avec le lecteur.

La lecture audio est optimisée afin d'assurer une expérience utilisateur fluide et continue.



Figure 4.4 – Interface du lecteur audio

4.1.5 Écran de gestion des playlists

La cinquième interface illustre la **gestion des playlists**.

Fonctionnalités :

- Affichage des playlists de l'utilisateur.
- Ajout de morceaux à une playlist.
- Organisation personnalisée du contenu musical.
- Sauvegarde des playlists associées à l'utilisateur authentifié.

Cette fonctionnalité permet une personnalisation avancée de l'expérience musicale.

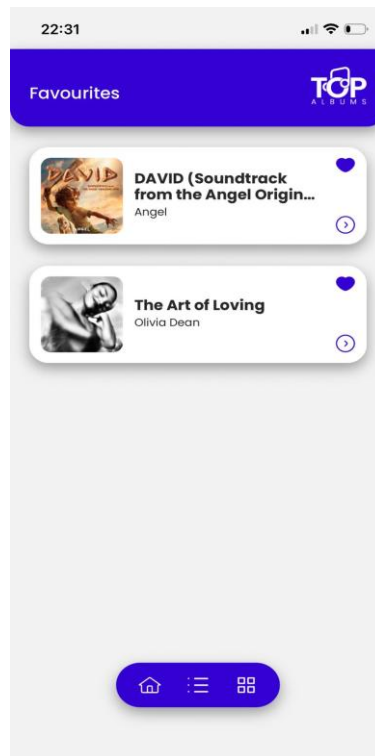


Figure 4.5 – Interface de gestion des playlists

4.2 Analyse globale de la réalisation

- L'application repose sur une **architecture modulaire**, séparant clairement les écrans, les composants et les services.
- La navigation entre les interfaces est fluide grâce à **React Navigation**.
- Firebase assure la gestion sécurisée des utilisateurs et des données musicales.
- L'utilisation de **Expo AV** permet une intégration efficace de la lecture audio.
- Les interfaces sont simples, cohérentes et adaptées à un usage mobile.

4.3 Conclusion du chapitre

Ce chapitre a permis de présenter la réalisation concrète de l'application React Native Music App à travers l'analyse détaillée des cinq interfaces développées. L'application répond aux objectifs fixés en offrant une solution de streaming musical simple, fonctionnelle et adaptée à un contexte académique, tout en exploitant les technologies modernes du développement mobile.

Chapitre 6 : Difficultés rencontrées et solutions

- **Gestion de la lecture audio** : instabilité de la lecture lors des changements d'écran et gestion des ressources audio.
Solution : utilisation de la bibliothèque Expo AV (Audio), permettant une gestion fiable de la lecture (play, pause, stop) et une libération automatique des ressources audio.
- **Compatibilité Expo et bibliothèques audio** : certaines fonctionnalités audio nécessitent des versions compatibles du SDK Expo.
Solution : alignement des versions des dépendances avec la version du SDK Expo utilisée et nettoyage du cache Expo afin d'éviter les conflits.
- **Configuration Firebase** : difficultés liées à l'initialisation du SDK Firebase et à la communication asynchrone avec l'application mobile.
Solution : configuration correcte du projet Firebase, centralisation de l'initialisation du SDK et utilisation de services dédiés pour l'authentification et l'accès aux données.
- **Authentification et persistance de session** : perte de session lors du redémarrage de l'application.
Solution : mise en place de Firebase Authentication avec persistance de session et écoute de l'état utilisateur afin de maintenir l'utilisateur connecté.
- **Navigation entre écrans** : complexité dans la gestion des transitions entre les écrans d'authentification et les écrans principaux.
Solution : structuration de la navigation à l'aide de React Navigation, avec une séparation claire entre les piles de navigation (authentification et application).
- **Performance de l'interface** : ralentissements lors du chargement des listes de morceaux.
Solution : optimisation du rendu des listes, chargement dynamique des données et limitation des re-rendus inutiles.
- **Sécurité des données** : protection des données utilisateur stockées dans Firebase.
Solution : mise en place de règles de sécurité Firebase restreignant l'accès aux données selon l'utilisateur authentifié.

Conclusion Générale

Le projet React Native Music App a permis de concevoir et de réaliser une application mobile moderne dédiée au streaming musical, intégrant des fonctionnalités essentielles telles que la lecture audio, la gestion des utilisateurs et une navigation fluide entre les différents écrans. La réalisation de ce projet a offert l'opportunité de mettre en pratique les concepts fondamentaux du développement mobile multiplateforme à l'aide de React Native et de l'environnement Expo.

Ce projet a également renforcé les compétences techniques en matière d'architecture applicative, de gestion de composants, ainsi que d'intégration de services backend à travers Firebase, notamment pour l'authentification des utilisateurs et la sécurisation des données. Par ailleurs, une attention particulière a été portée à l'expérience utilisateur afin de proposer une interface claire, intuitive et agréable à utiliser.