ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

RAPPORT DE PROJET

Conception et réalisation d'une application mobile React Native de football « KooraGoal ».

4ème année Ingénierie Informatique et Réseaux

Réalisé par :
Hamza Taoufik
Aymane Namous
Ghali Ourhzal

Encadré par :
Pr. Zineb Naji

Année universitaire 2025/2026

Remerciements

Au terme de ce travail, nous tenons à exprimer notre profonde gratitude et nos sincères remerciements à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce projet.

Nous adressons nos remerciements les plus vifs à notre encadrant, **Mme Zineb Naji** pour sa disponibilité, ses conseils avisés et ses orientations précieuses qui nous ont guidés tout au long de la phase de développement.

Nous tenons également à remercier l'ensemble du corps professoral de l'**École Marocaine des Sciences de l'Ingénieur** pour la qualité de la formation dispensée et pour nous avoir fourni les outils théoriques et pratiques nécessaires à la réussite de ce projet.

Enfin, nous remercions nos familles et nos amis pour leur soutien moral et leurs encouragements constants.

Résumé

L'essor des technologies mobiles a transformé la manière dont les amateurs de sport consomment l'information. Ce projet, intitulé **"KooraGoal"**, consiste en la conception et le développement d'une application mobile multiplateforme dédiée aux fans de football.

L'objectif principal est de fournir une interface fluide et intuitive permettant de suivre les scores des matchs en direct via une API REST, de consulter les actualités sportives et de gérer des favoris (équipes et joueurs). L'application intègre des fonctionnalités modernes telles que l'authentification sécurisée, la personnalisation du profil (avatar), le support multilingue (Français/Anglais) et la gestion de thèmes (Mode Sombre/Clair).

D'un point de vue technique, l'application est développée avec le framework **React Native** et l'écosystème **Expo**. Elle s'appuie sur une architecture robuste utilisant **Firebase** pour l'authentification et **SQLite** pour la persistance locale des données, garantissant ainsi une expérience utilisateur optimale même en cas de connectivité limitée.

Mots-clés : *Développement Mobile, React Native, Expo, Firebase, SQLite, API REST, Football, Live Score.*

Abstract

The rise of mobile technologies has transformed how sports enthusiasts consume information. This project, titled **"KooraGoal"**, involves the design and development of a cross-platform mobile application dedicated to football fans.

The main objective is to provide a smooth and intuitive interface for following live match scores via a REST API, consulting sports news, and managing favorites (teams and players). The application integrates modern features such as secure authentication, profile customization (avatar), multilingual support (French/English), and theme management (Dark/Light Mode).

From a technical perspective, the application is developed using the **React Native** framework and the **Expo** ecosystem. It relies on a robust architecture using **Firebase** for the authentication and **SQLite** for local data persistence, ensuring an optimal user experience even with limited connectivity.

Keywords: *Mobile Development, React Native, Expo, Firebase, SQLite, REST API, Football, Live Score.*

Table des matières

Liste des figures

Liste des acronymes

API	Application Programming Interface
AVD	Android Virtual Device
BaaS	Backend as a Service
i18n	Internationalization
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
REST	Representational State Transfer
UI	User Interface
UID	Unique Identifier
UX	User Experience

Introduction Générale

La transformation numérique a profondément impacté l'industrie du sport. Aujourd'hui, l'instantanéité est devenue une exigence fondamentale pour les supporters qui souhaitent accéder aux résultats, aux statistiques et aux actualités de leurs équipes favorites, où qu'ils soient et à n'importe quel moment. Le smartphone est ainsi devenu le premier écran pour le suivi des événements sportifs en direct.

C'est dans ce contexte dynamique que s'inscrit notre projet : la réalisation de l'application **"KooraGoal"**. Il s'agit d'une solution mobile native conçue pour centraliser l'expérience du fan de football. Au-delà du simple affichage des scores, l'application vise à offrir une expérience personnalisée grâce à la gestion de comptes utilisateurs, la sauvegarde de favoris en local et une interface moderne et adaptable.

Ce projet pédagogique a pour ambition de nous permettre de maîtriser le cycle complet de développement d'une application mobile moderne. Il met en œuvre des technologies telles que **React Native** pour le développement cross-platform, **Firebase** pour l'authentification et **SQLite** pour le stockage local.

Ce rapport détaille les étapes de conception et de réalisation de ce projet. Il s'articule autour de quatre chapitres principaux :

- Le **premier chapitre** est consacré au cadrage du projet. Il définit la problématique, les objectifs ainsi que les spécifications fonctionnelles et techniques (cahier des charges), tout en présentant la méthodologie de gestion de projet adoptée.
- Le **deuxième chapitre** présente la conception et l'architecture de la solution. Nous y détaillerons la structure technique, les diagrammes de navigation ainsi que la modélisation des données.
- Le **troisième chapitre** porte sur la réalisation technique. Il décrit l'environnement de développement, l'implémentation des fonctionnalités clés (Authentification, API, Favoris, Paramètres) et présente les interfaces graphiques finales.
- Enfin, le **quatrième chapitre** dresse un bilan du projet, en analysant les difficultés rencontrées, les solutions apportées, et en ouvrant sur les perspectives d'évolution de l'application.

CHAPITRE 1

CADRAGE ET ANALYSE DU PROJET

1. Contexte et Problématique

Le football est aujourd'hui le sport le plus populaire au monde, générant un flux d'informations continu (scores, transferts, statistiques). Cependant, les passionnés se retrouvent souvent contraints de naviguer entre plusieurs plateformes pour obtenir des scores en direct, lire des actualités fiables et suivre leurs équipes favorites.

Le projet **"KooraGoal"** naît de ce constat. Il s'agit de concevoir une application mobile centralisée, fluide et ergonomique, capable de regrouper l'ensemble de ces besoins en une interface unique. L'objectif est de proposer une expérience utilisateur optimale, allant de la consultation des scores en temps réel à la gestion personnalisée d'un profil utilisateur.

2. Objectifs du projet

Les objectifs principaux de l'application KooraGoal sont les suivants :

- **Centralisation :** Fournir les scores et les actualités au même endroit.
- **Temps réel :** Assurer la fraîcheur des données via une consommation d'API performante.
- **Personnalisation :** Permettre à l'utilisateur d'adapter l'application à ses préférences (langue, thème, favoris).
- **Communauté :** Offrir un espace membre sécurisé via authentification.

3. Spécification des besoins fonctionnels

Pour répondre à ces objectifs, l'application s'articule autour de plusieurs modules fonctionnels clés :

1. **Module d'Information (Cœur de l'application) :**
    - Affichage des scores de matchs en direct (Live Score).
    - Consultation du flux d'actualités (News Feed) avec détails des articles.
2. **Module Utilisateur (Authentification et Profil) :**
    - Inscription et connexion sécurisées (gérées via **Firebase Auth**).
    - Gestion du profil utilisateur : modification des informations personnelles et mise à jour de l'avatar (via la bibliothèque d'images).

3. **Module de Personnalisation :**
    - **Gestion des Favoris :** Possibilité d'ajouter des joueurs et des équipes en favoris pour un accès rapide.
    - **Internationalisation (i18n) :** Support multilingue complet (Français / Anglais).
    - **Thématisation :** Support d'un mode Sombre (Dark Mode) et Clair (Light Mode) pour le confort visuel.

4. Méthodologie de gestion de projet

Pour garantir le bon déroulement du développement de **KooraGoal** et respecter les délais impartis, nous avons adopté une démarche structurée combinant agilité et planification temporelle rigoureuse.

4.1. Méthode Agile et Kanban

Afin de gérer efficacement le flux de travail et de s'adapter aux imprévus techniques, nous avons opté pour la méthode **Kanban** via la plateforme **Taiga**. Cette approche visuelle a permis de décomposer le projet en tâches unitaires (User Stories) et de suivre leur état d'avancement en temps réel. Le tableau de bord a été organisé en colonnes classiques (« À faire », « En cours », « Terminé »), offrant une vue d'ensemble claire sur les fonctionnalités restantes et celles déjà implémentées. Cette méthode a favorisé une cadence de développement régulière et une meilleure priorisation des modules critiques (comme l'authentification et l'API).

4.2. Planification prévisionnelle (Diagramme de Gantt)

En parallèle de la gestion quotidienne des tâches, une vision macroscopique du projet a été établie à l'aide d'un **diagramme de Gantt**.

Tâche	Début	Fin	Durée	Planning (22 Déc 2025 – 5 Jan 2026)
PHASE INITIALE & ANALYSE				
Documentation et analyse des besoins	22 Déc	24 Déc	3 jours	3j
CONCEPTION				
Conception (diagrammes UML, navigation, architecture)	25 Déc	31 Déc	7 jours	7j
DÉVELOPPEMENT				
Setup environnement (React Native, Expo, Firebase, API)	01 Jan	02 Jan	2 jours	2j
Développement Authentification & Profil	01 Jan	03 Jan	3 jours	3j
Développement Matchs, Actualités et Favoris	27 Déc	04 Jan	9 jours	9j
Implémentation Thème & Multilingue	03 Jan	04 Jan	2 jours	2j
FINALISATION				
Tests & corrections bugs	04 Jan	05 Jan	2 jours	2j
Rédaction rapport final & dépôt livrables	02 Jan	05 Jan	4 jours	4j

Figure 1: Diagramme de Gantt

5. Conclusion

Ce premier chapitre a permis de définir le périmètre fonctionnel de l'application **KooraGoal**, d'identifier les besoins des utilisateurs et de structurer la démarche de gestion de projet. Avec des objectifs clairs et une méthodologie agile en place, nous disposons désormais d'une base solide. La prochaine étape consiste à traduire ces besoins en une solution technique viable. Le chapitre suivant détaillera l'architecture logicielle, la modélisation des données et la conception graphique nécessaires au développement de l'application.

CHAPITRE 2

CONCEPTION ET ARCHITECTURE

1. Introduction

La phase de conception constitue une étape charnière du cycle de vie du projet. Elle permet de définir l'ossature technique de la solution et de valider les choix structurels avant d'entamer le développement. Ce chapitre détaille l'architecture globale de l'application **KooraGoal**, justifie les technologies sélectionnées et présente la modélisation des données ainsi que la logique de navigation.

1.1. Architecture technique globale

Pour répondre aux exigences de performance et de portabilité, le développement de l'application repose sur l'écosystème **React Native** couplé à la plateforme **Expo**. Ce choix stratégique permet de produire une application native compatible à la fois avec Android et iOS à partir d'une base de code unique en JavaScript. L'architecture backend adopte une approche hybride. D'une part, nous utilisons une architecture **Serverless** via **Firebase** (BaaS) pour déléguer la complexité de la gestion de l'authentification. D'autre part, les données sportives (scores et classements) sont consommées en temps réel via une **API REST** externe spécialisée. Enfin, pour garantir une expérience fluide et la persistance des données utilisateur, le moteur de base de données **SQLite** est intégré directement dans l'application mobile.

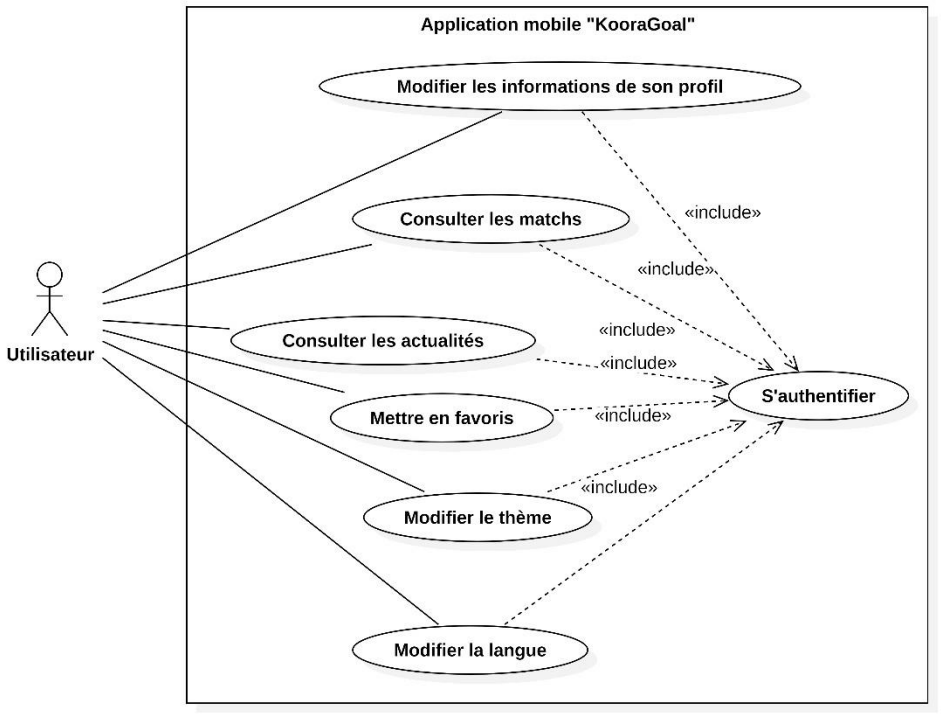

Figure 2:Diagramme des cas d'utilisation

1.2. Architecture logicielle

L'application est structurée selon une architecture basée sur les composants, favorisant la réutilisabilité et la maintenance du code. Cette structure se divise logiquement en trois couches distinctes. La couche de présentation regroupe les écrans principaux de l'application, tandis que la couche de composants rassemble les éléments d'interface réutilisables comme les cartes de match ou les boutons. La couche de services isole la logique métier et les appels réseaux afin de ne pas surcharger l'interface graphique. De plus, la gestion de l'état global de l'application, notamment pour le thème sombre et la langue, est assurée par le Context de React.

1.3. Conception des données

La stratégie de gestion des données combine le stockage distant pour l'authentification et le stockage local pour la personnalisation.

1.3.1. Gestion des utilisateurs (Firebase)

Le modèle utilisateur est géré par le service Firebase Authentication. Cette solution sécurisée attribue à chaque utilisateur un identifiant unique (UID) qui sert de clé primaire.

1.3.2. Modèle de données local (SQLite - Favoris)

Afin d'optimiser les performances et de réduire les appels API redondants, la gestion des favoris est assurée localement via une base de données SQLite embarquée.

1.4. Conception graphique et navigation

La navigation au sein de l'application est orchestrée par la bibliothèque React Navigation, qui gère la pile d'historique et les transitions entre les écrans.

Le flux utilisateur est conçu pour être intuitif. Il débute par une pile d'authentification (Stack Navigator) qui redirige l'utilisateur vers l'écran principal une fois connecté. Ensuite, la navigation principale repose sur un menu inférieur (Tab Navigator) permettant de basculer instantanément entre l'Accueil, les Matchs, les Favoris et les Paramètres, sans perdre le contexte de navigation.

L'interface utilisateur a été pensée pour offrir un confort visuel optimal dans toutes les conditions d'éclairage. Techniquement, cela se traduit par l'implémentation d'un système de thèmes dynamiques. L'application utilise des variables globales pour les couleurs de fond et

de typographie, qui commutent automatiquement entre le mode **Clair** et le mode **Sombre** selon la préférence définie par l'utilisateur dans les paramètres de l'application.

1.5.Conclusion

Ce chapitre a permis de poser les fondations techniques de l'application **KooraGoal**. Nous avons défini une architecture hybride et modulaire, tirant parti de la flexibilité de React Native pour l'interface, ainsi que de la robustesse de Firebase combinée à SQLite pour une gestion optimisée des données. Les choix de conception, tant au niveau du modèle de données que de la navigation, visent à garantir une expérience utilisateur fluide, personnalisable et performante. Cette structure validée nous permet désormais d'aborder la phase de réalisation technique présentée dans le chapitre suivant.

CHAPITRE 3

RÉALISATION DE L'APPLICATION

1. Introduction

Après avoir validé l'architecture logicielle et la conception des données lors de la phase précédente, ce chapitre est consacré à la mise en œuvre concrète de l'application **KooraGoal**. Il a pour objectif de détailler le passage de la théorie à la pratique, en présentant l'environnement technique déployé et les méthodes de développement adoptées. Nous y décrirons la configuration des outils, l'implémentation des fonctionnalités critiques (authentification, consommation d'API, persistance locale) et nous conclurons par une présentation visuelle des interfaces utilisateur réalisées.

2. Environnement de développement

Après avoir établi l'architecture et la conception de l'application, nous entamons la phase de réalisation. Ce chapitre a pour objectif de détailler l'environnement technique mis en place, les outils exploités ainsi que l'implémentation concrète des fonctionnalités. Il présente la traduction des besoins fonctionnels en code source et illustre le résultat final à travers l'interface utilisateur.

2.1.1. Outils et IDE

Pour garantir un cycle de développement efficace, le choix de l'Environnement de Développement Intégré (IDE) s'est porté sur **Visual Studio Code**. Cet éditeur, léger et extensible, a été configuré avec des plugins spécifiques à l'écosystème JavaScript et React, facilitant ainsi l'écriture du code et le formatage automatique. En complément, **Android Studio** a été indispensable pour la configuration des émulateurs (Android Virtual Devices). Il a permis de simuler le comportement de l'application sur différents terminaux mobiles et de surveiller les journaux d'erreurs (logs) en temps réel. La gestion de versions a été assurée par Git, permettant de conserver un historique des modifications tout au long du projet, le codebase étant par la suite hébergé sur GitHub.

2.1.2. Technologies

React Native constitue le socle fondamental de notre développement. Ce framework open-source, développé par Meta, a été choisi pour sa capacité à produire des applications mobiles natives à partir d'une base de code unique en JavaScript. Il permet de composer une interface utilisateur riche via des composants déclaratifs tout en offrant des performances proches du natif.

Expo a été utilisé comme surcouche à React Native pour accélérer le processus de création et de déploiement. Cet ensemble d'outils et de services facilite considérablement l'accès aux API natives du téléphone, telles que la caméra pour la gestion de l'avatar ou le système de fichiers, sans nécessiter de configuration complexe des fichiers natifs Android.

Concernant les données sportives, l'application consomme une l'**API REST** : API-Football offerte par API-Sports Cette interface de programmation fournit les flux de données au format JSON, incluant les scores en direct, les détails des matchs et les articles d'actualité. Elle agit comme la source de vérité dynamique de l'application, interrogée via des requêtes HTTP asynchrones.

SQLite est la technologie retenue pour la persistance locale des données. Contrairement aux solutions de stockage simples (comme AsyncStorage), SQLite offre une véritable base de données relationnelle embarquée dans le mobile. Elle est utilisée ici pour structurer et sauvegarder de manière pérenne la liste des favoris de l'utilisateur, garantissant un accès rapide aux informations même en mode hors ligne.

3. Implémentation des fonctionnalités clés

3.1.1. Module d'authentification et Profil

La sécurisation de l'application repose sur le service **Firebase Authentication**. L'implémentation gère le cycle de vie complet de l'utilisateur : de l'inscription à la connexion, jusqu'à la déconnexion. Une fois authentifié, l'application maintient la session active grâce à un observateur d'état. Pour la gestion du profil, nous avons intégré un sélecteur d'images (Image Picker). Ce module demande les permissions nécessaires à l'utilisateur pour accéder à la galerie, récupère l'image sélectionnée et permet de définir un avatar personnalisé qui s'affiche ensuite dans l'en-tête du profil et le menu de navigation.

3.1.2. Gestion des Matchs et Actualités

L'affichage des matchs en direct et des actualités constitue le cœur fonctionnel de l'application. Techniquement, cette partie repose sur l'utilisation du hook useEffect de React pour déclencher la récupération des données via l'API dès le chargement de l'écran. Pour optimiser les performances d'affichage face à des listes potentiellement longues, nous avons utilisé le composant FlatList. Celui-ci implémente un mécanisme de "lazy loading" (chargement paresseux), ne rendant graphiquement que les éléments visibles à l'écran, ce qui garantit une fluidité de défilement (scroll) optimale.

3.1.3. Système de Favoris (Intégration SQLite)

Le système de favoris permet aux utilisateurs de suivre leurs équipes et joueurs préférés. Cette fonctionnalité a nécessité la création d'une couche d'abstraction pour communiquer avec la base de données **SQLite** locale. Les opérations d'ajout et de suppression sont exécutées via des transactions SQL asynchrones pour ne pas bloquer l'interface utilisateur. Lors du lancement de l'application, une requête est effectuée pour charger les favoris en mémoire et mettre à jour l'interface visuelle.

3.1.4. Gestion des Paramètres (Internationalisation & Thèmes)

La personnalisation de l'expérience utilisateur est gérée de manière centralisée grâce à l'API **Context** de React. Pour l'internationalisation (i18n), un contexte dédié distribue les chaînes de caractères dans la langue choisie (Français ou Anglais) à travers toute l'application. De la même manière, la gestion du thème (Sombre ou Clair) utilise un contexte de style qui contient les variables de couleurs. Lorsqu'un utilisateur bascule un interrupteur dans les paramètres, l'état global est mis à jour, provoquant le re-rendu immédiat de tous les composants avec la nouvelle palette graphique.

4. Présentation de l'interface utilisateur

Afin de faciliter la compréhension et de refléter une expérience utilisateur fluide, les captures d'écran sont organisées selon un flux logique du parcours utilisateur : Nous commençons par l'onboarding (accueil des nouveaux utilisateurs, inscription/connexion), puis nous explorons les écrans principaux, les interactions clés et les principales fonctionnalités de l'application. Enfin, nous concluons par les mécanismes de gestion du profil et de déconnexion.

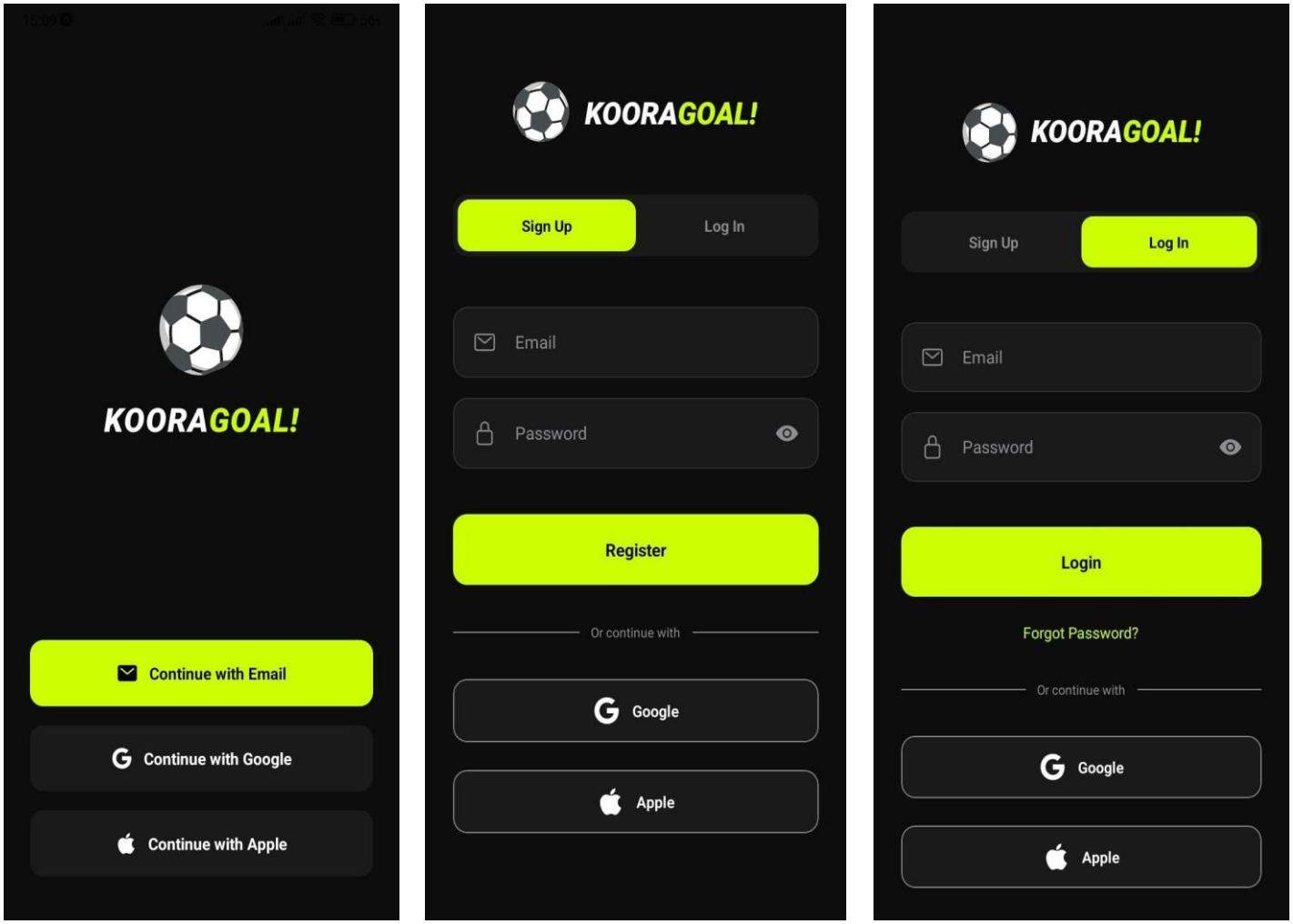

Figure 3:(De gauche à droite) Interfaces d'onboarding, d'inscription et de connexion

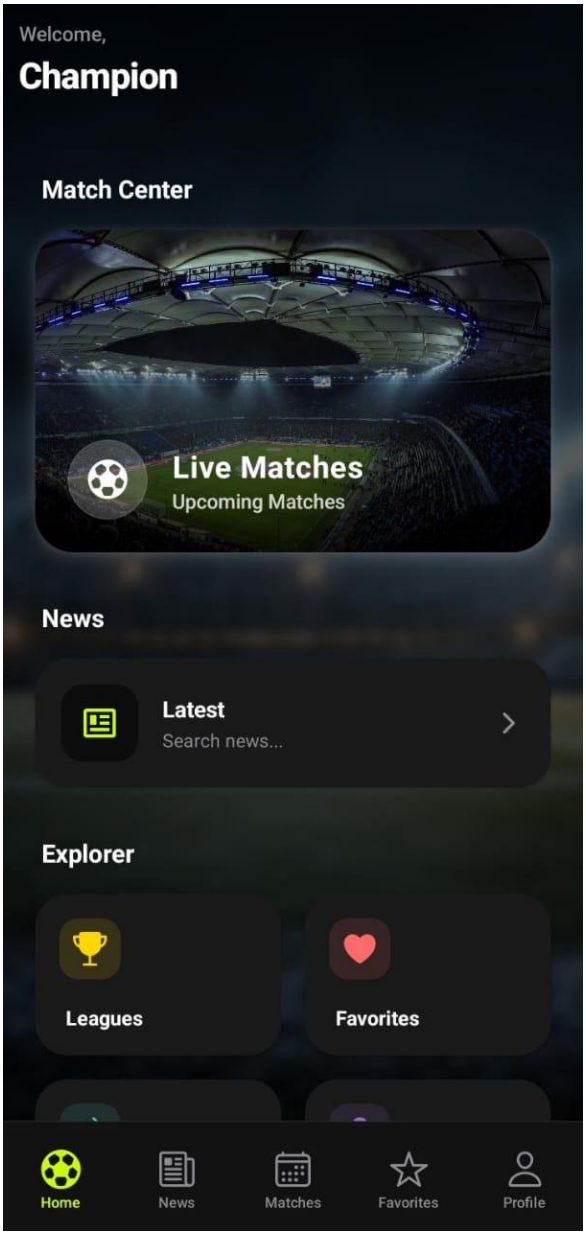

Figure 4: Interface d'accueil regroupant des accès rapides

Figure 5:Interface "Actualités" permet à l'utilisateur d'avoir les dernières nouveautés du football

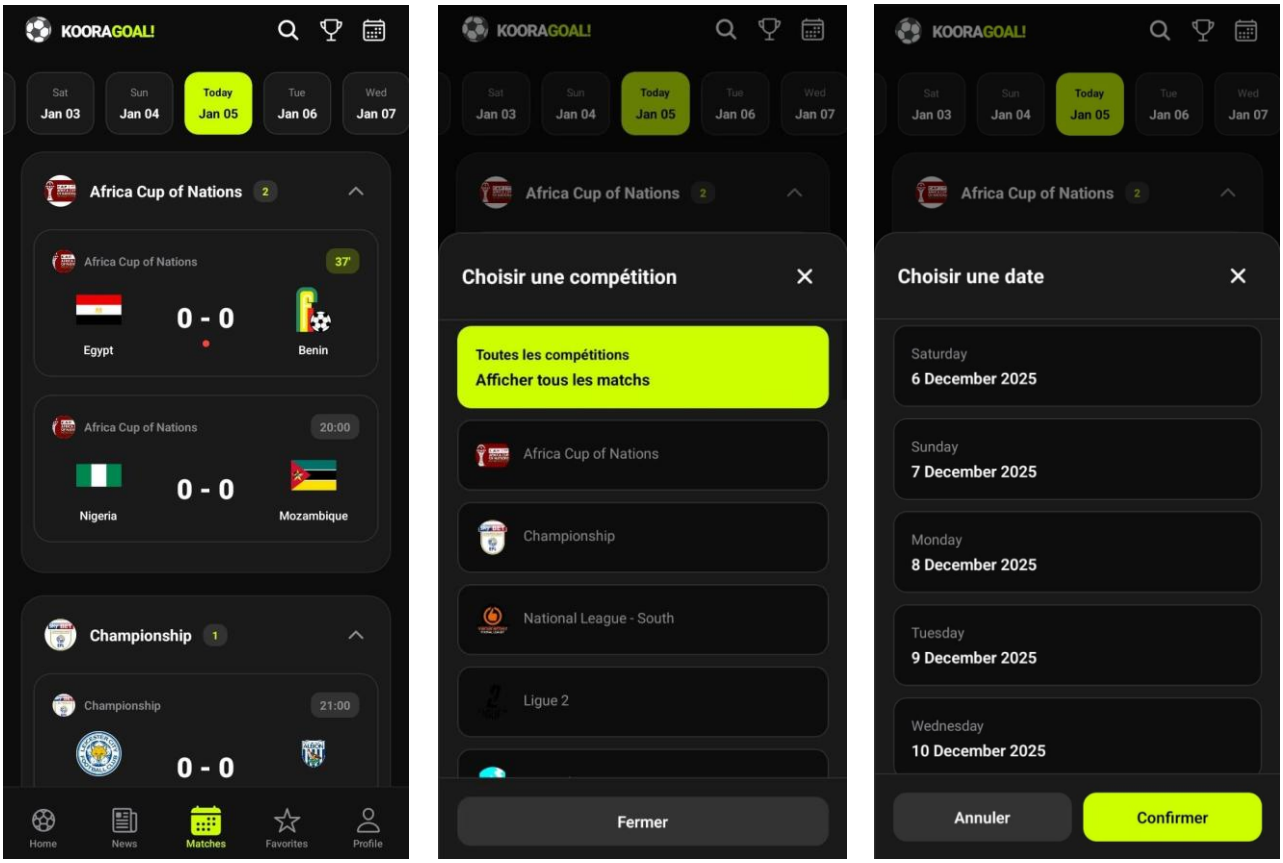

Figure 6: Interfaces "Matchs" affiche les scores des matchs en cours, le planning des matchs avec filtrage

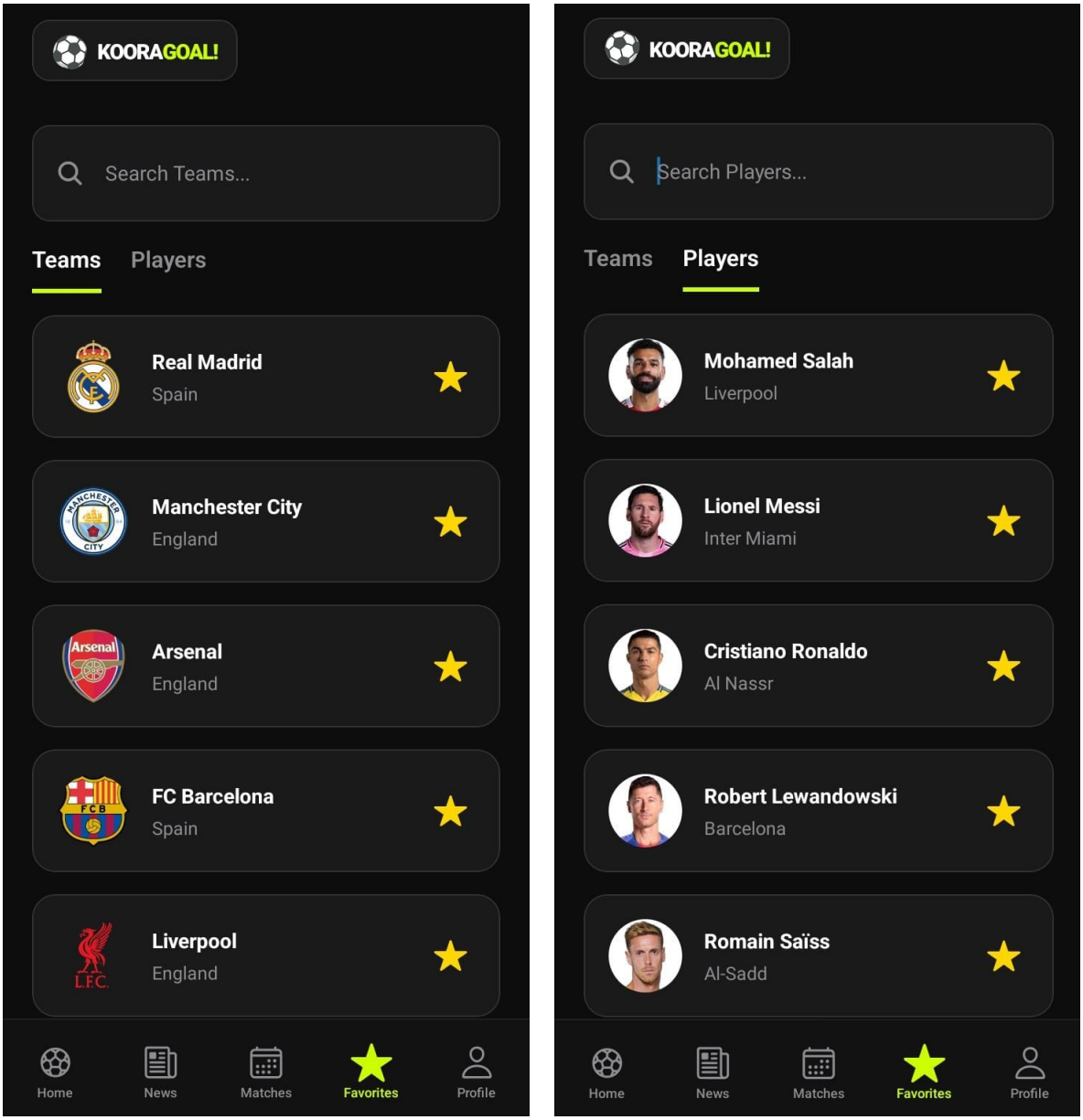

Figure 7: Interface "Favoris" permet à l'utilisateur de mettre en favoris des équipes et des joueurs

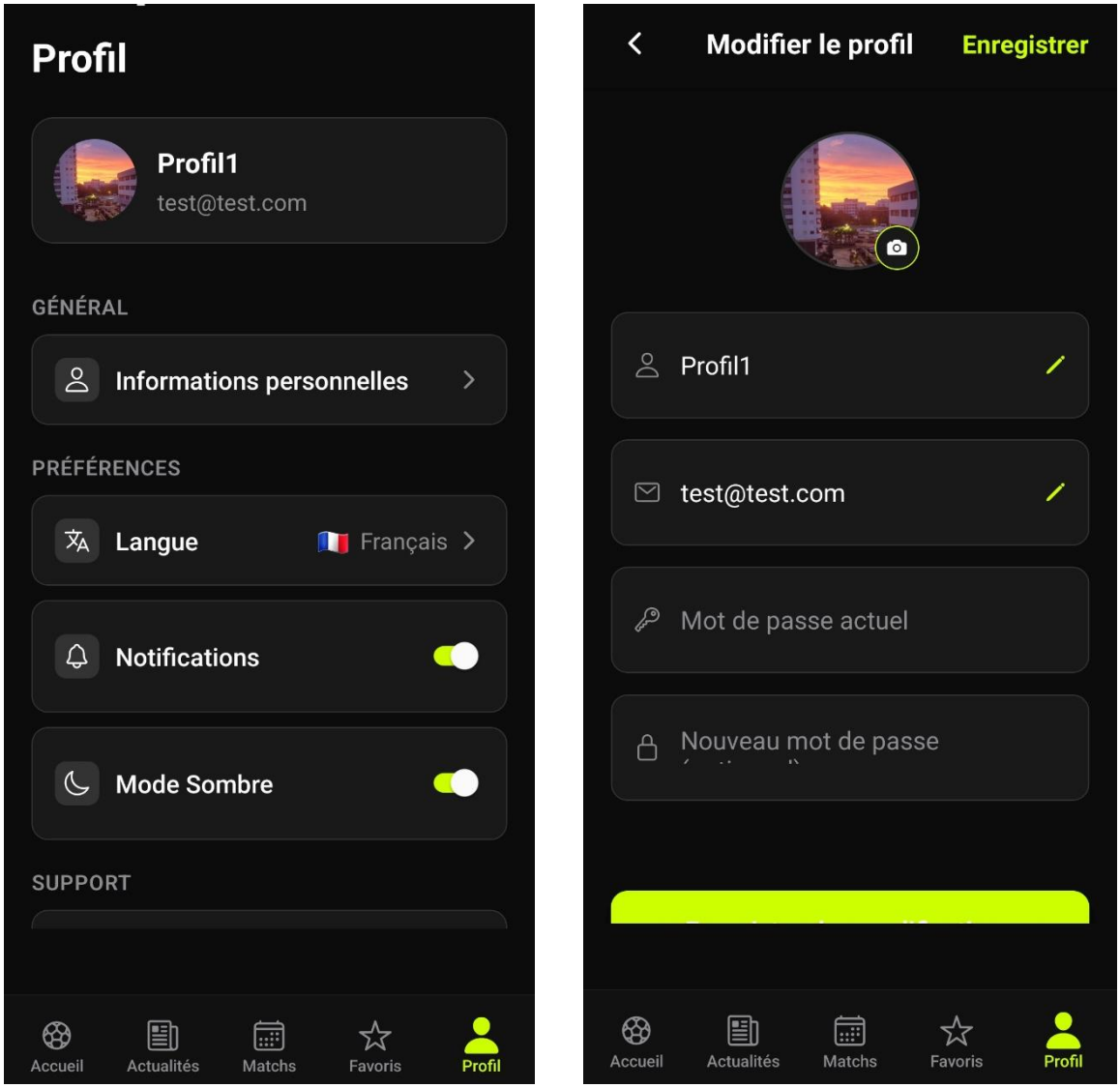

Figure 8:Interface "Profil" permettant à l'utilisateur de personnaliser l'application (langue et thèmes) et de pouvoir modifier des données relatives à son compte.

5. Conclusion

La phase de réalisation technique détaillée dans ce chapitre a abouti à une version stable et fonctionnelle de l'application **KooraGoal**. L'utilisation conjointe de React Native, Firebase et SQLite a permis de répondre efficacement aux spécifications fonctionnelles, offrant une expérience utilisateur fluide, connectée et personnalisable. Maintenant que le produit est opérationnel, il est essentiel d'analyser le déroulement du projet. Le chapitre suivant dressera le bilan de cette expérience, en mettant en lumière les défis techniques surmontés et les enseignements tirés de ce développement.

CHAPITRE 4

BILAN FINAL DU PROJET

1. Introduction

La phase de réalisation technique étant désormais achevée, il convient de porter un regard rétrospectif et critique sur l'ensemble du déroulement du projet **KooraGoal**. Ce chapitre a pour vocation de dresser un bilan analytique de notre travail. Nous y exposerons les principaux obstacles techniques et méthodologiques rencontrés durant le développement, ainsi que les stratégies de résolution adoptées pour les surmonter. Enfin, nous conclurons par une synthèse des compétences acquises et des enseignements tirés de cette expérience.

2. Difficultés rencontrées

Tout projet de développement logiciel s'accompagne inévitablement de défis techniques et organisationnels. Durant la réalisation de **KooraGoal**, la première difficulté majeure a concerné la gestion de l'asynchronisme et la latence réseau. La consommation d'une API de football en temps réel implique de gérer des délais de réponse variables qui peuvent impacter l'expérience utilisateur, notamment l'effet de gel de l'interface lors du chargement des données.

Un second obstacle technique s'est présenté lors de l'intégration de la base de données locale SQLite avec Expo. La configuration de la persistance des données a généré des erreurs de compatibilité initiales, particulièrement lors de la gestion des migrations de tables et de la récupération des images stockées en local pour les favoris.

Enfin, la gestion globale de l'état (State Management) pour le thème sombre et l'internationalisation s'est avérée plus complexe que prévu. Assurer que chaque composant, même le plus imbriqué, réagisse instantanément au changement de langue ou de couleur sans provoquer de re-rendus inutiles a nécessité une architecture rigoureuse des contextes React.

3. Solutions apportées

Face à ces problématiques, plusieurs solutions techniques ont été mises en œuvre. Pour pallier les problèmes de latence, nous avons systématiquement implémenté des indicateurs de chargement et utilisé le chargement paresseux. Cela permet d'afficher l'interface squelette avant même l'arrivée des données, améliorant ainsi la perception de fluidité.

Concernant la base de données SQLite, la solution a consisté à isoler toute la logique de base de données dans un service dédié. Nous avons également utilisé des transactions asynchrones

pour sécuriser les opérations d'écriture et de lecture, garantissant ainsi l'intégrité des données sans bloquer le fil d'exécution principal.

Pour la gestion de l'état, l'utilisation optimisée de l'API Context de React, a permis de centraliser la logique du thème et de la langue. Cela a simplifié le code et a rendu la maintenance du code beaucoup plus aisée.

4. Bilan final

Au terme de ce projet, le bilan est extrêmement positif. D'un point de vue technique, ce développement a permis de consolider la maîtrise du framework React Native et de comprendre en profondeur le cycle de vie des applications mobiles. L'utilisation de Firebase pour l'authentification et de SQLite pour le stockage local a offert une vision complète d'une architecture hybride Cloud/Local.

Sur le plan personnel et méthodologique, ce projet a renforcé nos compétences en gestion du temps et en résolution de problèmes. L'obligation de respecter un cahier des charges précis tout en s'adaptant aux contraintes techniques a été une expérience formatrice, nous préparant efficacement aux exigences du milieu professionnel de l'ingénierie logicielle.

Conclusion générale et perspectives

Le projet **KooraGoal** avait pour ambition de proposer une application mobile complète dédiée aux amateurs de football, centralisant les scores en direct, les actualités et une gestion personnalisée des favoris. Au terme de ce cycle de développement, nous pouvons affirmer que les objectifs initiaux ont été atteints. L'application est fonctionnelle, fluide et respecte les standards actuels de développement mobile grâce à l'utilisation de **React Native** et de l'écosystème **Expo**.

L'architecture technique mise en place, combinant une API REST performante, la robustesse de **Firebase** pour la gestion des utilisateurs et la flexibilité de **SQLite** pour le stockage local, a permis de créer une solution stable et évolutive. L'intégration de fonctionnalités avancées telles que le mode sombre et le support multilingue témoigne du soin apporté à l'expérience utilisateur.

Cependant, le développement d'une application est un processus itératif et continu. Plusieurs perspectives d'évolution sont envisageables pour enrichir **KooraGoal** dans le futur :

- **Système de Notifications Push :** Intégrer un service de notifications pour alerter l'utilisateur en temps réel lors d'un but ou du début d'un match favori.
- **Fonctionnalités Sociales :** Ajouter un espace de commentaires sous les matchs ou un forum pour permettre aux utilisateurs d'interagir entre eux.
- **Publication sur les Stores :** Finaliser les étapes de signature numérique pour déployer l'application sur le Google Play Store et l'Apple App Store.

En conclusion, ce projet de fin d'année a été une opportunité concrète de synthétiser nos connaissances académiques et de les appliquer à un cas réel, aboutissant à un produit dont nous sommes fiers.

Webographie

- Documentation de React Native : https://reactnative.dev/docs
- Documentation d'Expo : https://docs.expo.dev/
- Documentation de Firebase Authentification : https://firebase.google.com/docs/auth
- Documentation d'Expo SQLite : https://docs.expo.dev/versions/latest/sdk/sqlite/