FPL Assistant – Final Project Report

1. Introduction

FPL Assistant is a cross-platform mobile application developed using **Flutter** to support **Fantasy Premier League (FPL)** managers in making informed and timely decisions regarding their teams.
The application aggregates official Fantasy Premier League data together with real Premier League statistics and presents them through an intuitive, visually appealing, and highly interactive user interface.

The system provides users with access to league standings, fixtures, match details, and team-planning tools, all integrated within a modern dark-themed design that follows official FPL branding guidelines.
Additionally, the application incorporates **Firebase services** for authentication, cloud data storage, and synchronization, as well as **local notifications** to improve user engagement.

This report presents the objectives of the application, identifies the target users, describes the features implemented throughout the three development phases, discusses the tools and technologies used, and provides an overview of the final system architecture and user interface.

---

2. Project Overview

2.1 Application Purpose

The primary objective of FPL Assistant is to serve as a comprehensive companion tool for Fantasy Premier League players.
The application aims to help users:

- Monitor **real-time Premier League standings and statistics**.
- Analyze **fixtures and fixture difficulty** across all gameweeks.
- View **detailed match information**, including scores, statistics, and lineups.
- Plan and manage **Fantasy Premier League squads and transfers**.
- Receive **notifications and reminders** for important or starred matches.

By centralizing all relevant information into a single mobile application, FPL Assistant minimizes the need for users to rely on multiple external platforms while managing their FPL teams.

2.2 Target Users

The application targets the following user groups:

- **Casual and competitive Fantasy Premier League managers** seeking efficient team-management support.
- **Premier League football fans** interested in real-time match data and league standings.
- **Sports analytics enthusiasts** who enjoy comparing team and player performance.
- **Mobile-first users** who prefer dedicated applications over web-based tools.

2.3 Overall System Functionality

At a high level, FPL Assistant provides the following core functionalities:

- A live **Premier League table** with detailed statistics and visual indicators.
- **Fixtures and results** organized by gameweek with kickoff times and difficulty ratings.
- **Match detail views** displaying scores, statistics, lineups, and match bookmarking.
- **Team planning and transfer tools** to assist FPL decision-making.
- **User authentication** using Firebase Authentication.
- **Cloud data synchronization** via Cloud Firestore.
- **Local notifications** for starred matches.
- A **modern, animated, dark-themed interface** consistent with FPL branding.

---

3. Features Implemented Across Development Phases

3.1 Phase 1 – Core Foundation

The first development phase focused on establishing the technical foundation of the application and enabling reliable access to football data.

Key deliverables included:

- **Project setup and architecture**
    - Creation of a Flutter project following a clean, modular structure.

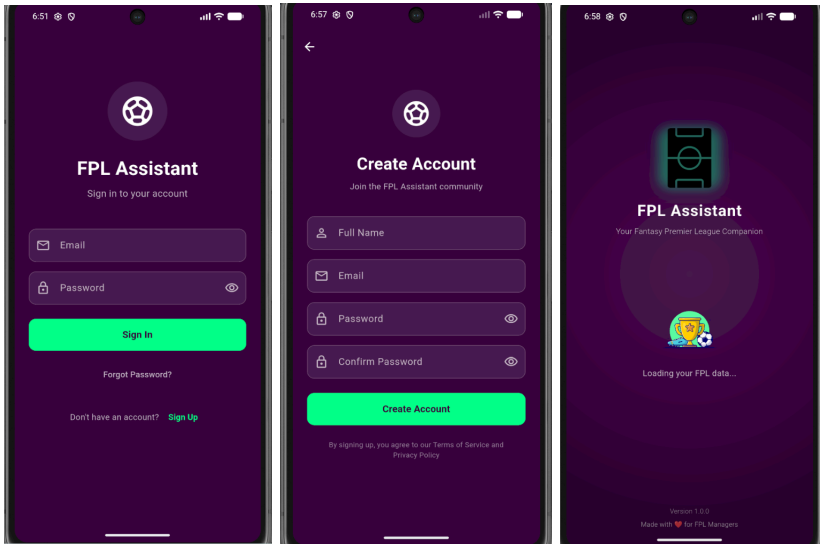
    - Definition of core folders such as `models`, `services`, `providers`, `screens`, and `core`.
    - Implementation of a base dark theme aligned with FPL colors.
- **API integration**
    - Integration with the **official Fantasy Premier League API** to retrieve teams, players, fixtures, and gameweeks.
    - Initial integration of a **secondary football data API** to enhance league standings information.
- **State management**
    - Adoption of the **Provider** pattern to manage application state.
    - Implementation of a central `FPLProvider` responsible for loading and coordinating football data.
- **Data modeling**
    - Definition of key data models including `Team`, `Fixture`, `Gameweek`, and `Player`.
- **Basic user interface**
    - Development of initial screens for the league table and fixtures using Material Design components.

3.2 Phase 2 – Feature Development

Phase 2 focused on implementing the main **user-facing features**, enhancing **data visualization**, and introducing **user authentication flows** required for personalized access and data persistence. This phase transformed the application from a basic data viewer into an interactive and user-centered system. The key features implemented during this phase are summarized below.

Authentication Screens (Sign In / Sign Up / Forgot Password)

- Implementation of a complete **authentication flow** using Firebase Authentication.
- **Sign In screen** allowing existing users to log in using email and password credentials.
- **Sign Up screen** enabling new users to create accounts securely.
- **Forgot Password screen** providing password reset functionality via email.
- Input validation and user feedback for incorrect credentials and empty fields.
- Seamless navigation between authentication screens and the main application upon successful login.

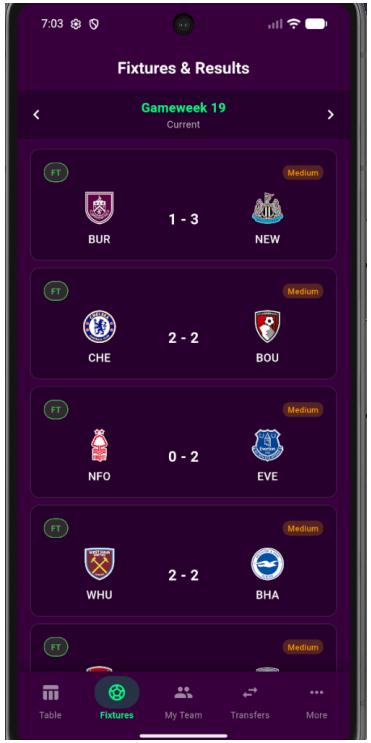


League Table Screen

- Display of the **complete Premier League standings** in a structured table format.
- Presentation of detailed team statistics, including matches played, wins, draws, losses, goals scored, goals conceded, goal difference, and total points.
- Visual indicators highlighting teams qualifying for the **Champions League**, **Europa League**, and those in **relegation zones**.
- Support for **pull-to-refresh** functionality to retrieve the most recent league data.

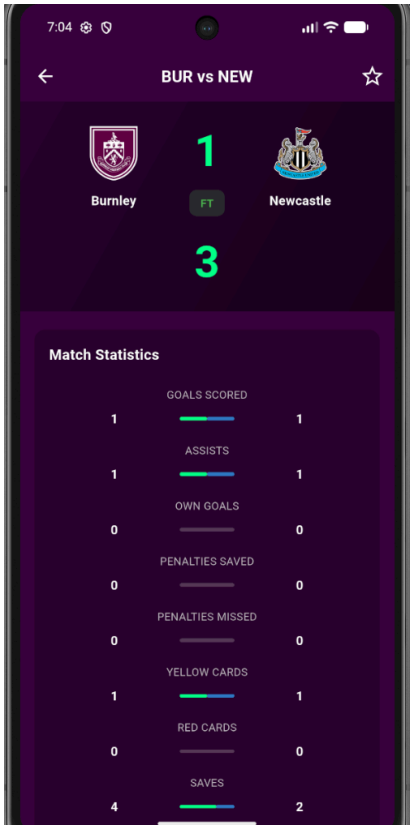
Fixtures Screen

- Gameweek-based fixture listing covering **all gameweeks (1–38)**.
- Clear differentiation between **finished**, **live**, and **upcoming** matches.
- Display of **team badges**, match scores for completed games, and kickoff times for scheduled fixtures.
- **Color-coded fixture difficulty ratings** to assist users in planning their Fantasy Premier League strategies.



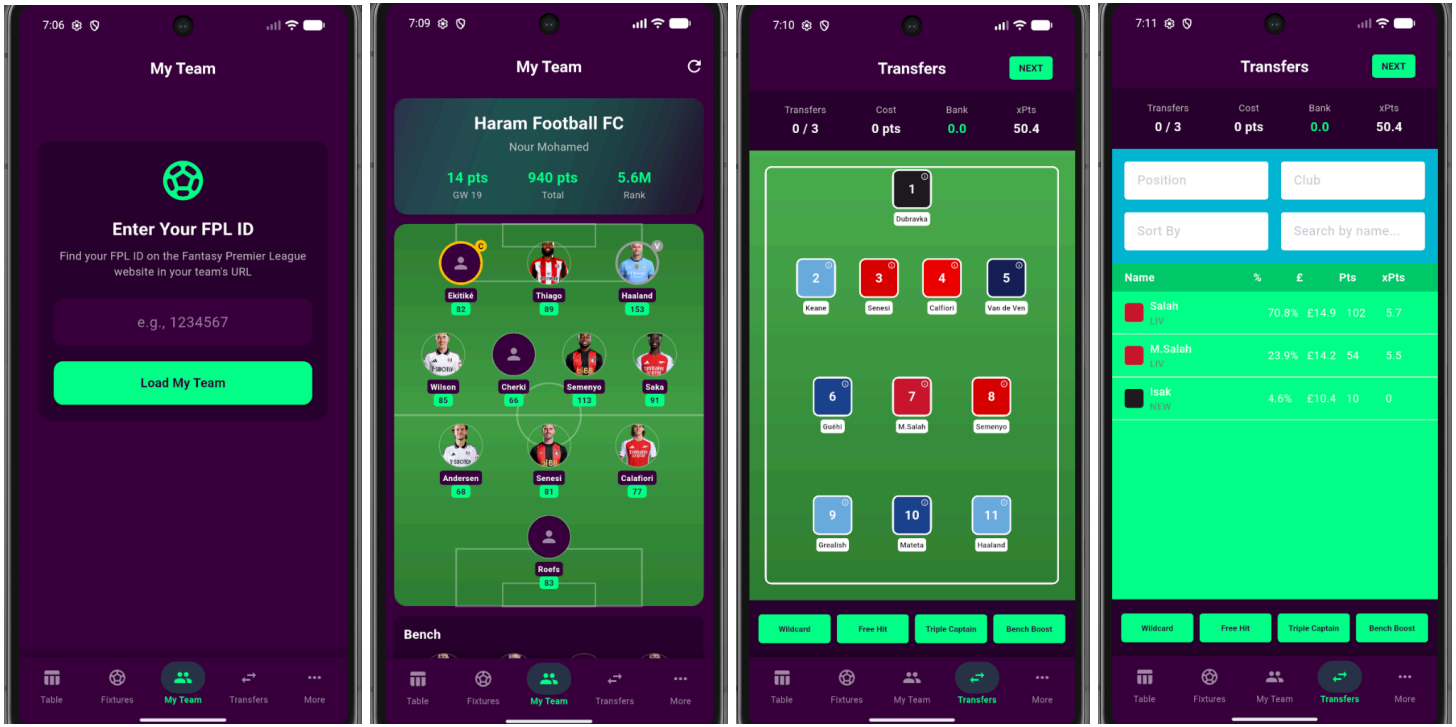
Match Details Screen

- Detailed match view presenting team logos, current or final scorelines, and key match statistics.
- Display of **confirmed or probable lineups** when available.
- Ability for users to **star or favorite matches** for quick access and future notifications.



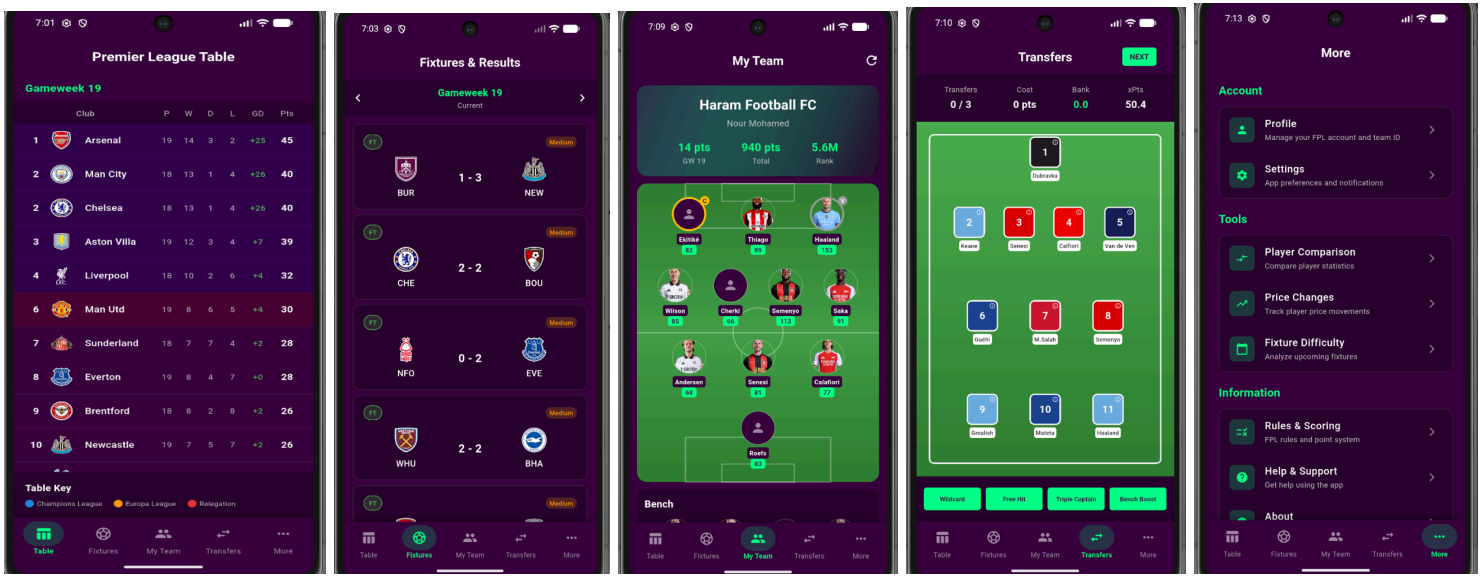
Team Planning and Transfers

- **“My Team” screen** displaying the user’s squad, selected formation, and overall team structure.
- **Transfers screen** providing a simplified player market view to support planning and decision-making for upcoming transfers.



Navigation

- Implementation of a **bottom navigation bar** providing intuitive access to the main application sections.
- Five primary tabs included:
    - League Table
    - Fixtures
    - My Team
    - Transfers
    - More



APK : https://drive.google.com/drive/folders/1zOYHDZsXa9xOn0E1BiF5QmSl20L2Hlls

---

3.3 Phase 3 – Finalization and Deployment

The final phase focused on polishing the application, improving user experience, and preparing the app for release.

Key improvements included:

- **User experience and animations**
    - Animated splash screen using official FPL colors.
    - Smooth page transitions and micro-animations to enhance usability.
- **Firebase Authentication**
    - Email and password authentication.
    - Persistent login sessions and secure sign-out functionality.
- **Cloud data synchronization**
    - Storage of starred matches and user preferences in Cloud Firestore.
    - Synchronization of user data across multiple devices.
- **Local notifications**
    - Integration of Flutter Local Notifications.

    - Runtime permission handling for notifications.
    - Match reminder alerts for starred fixtures.
- **Branding and deployment**
    - Custom launcher icons for all supported platforms.
    - Creation of an optimized release build with code shrinking enabled.

---

4. Tools, Technologies, and Frameworks

4.1 Core Technologies

- **Flutter** – Cross-platform framework for building the application UI and logic.
- **Dart** – Programming language used throughout the project.

4.2 State Management and Architecture

- **Provider** – Manages application state and UI updates.
- **Modular architecture** – Ensures maintainability and scalability.

4.3 Backend and Cloud Services

- **Firebase Core** – Firebase initialization and configuration.
- **Firebase Authentication** – User login and session management.
- **Cloud Firestore** – Storage of user data, starred matches, and preferences.

4.4 Networking and APIs

- **HTTP package** – Used for RESTful API communication.
- **Fantasy Premier League API** – Source of official FPL data.
- **External football data API** – Provides league standings and statistics.
- **Cached Network Image** – Efficient loading and caching of images.

4.5 Notifications and Local Storage

- **Flutter Local Notifications** – Match reminders and alerts.
- **Permission Handler** – Runtime permission management.
- **Shared Preferences** – Lightweight local data storage.

4.6 Development Tools

- **Visual Studio Code** – Primary development environment.

- **Android Studio** – Emulator and SDK management.
- **Git and GitHub** – Version control and collaboration.

---

5. System Architecture

The application follows a **layered architecture** that separates responsibilities across distinct components:

- **Models Layer** – Defines data structures.
- **Services Layer** – Handles API communication and Firebase operations.
- **Providers Layer** – Manages business logic and application state.
- **Presentation Layer** – UI screens and navigation.

This design improves code readability, maintainability, and extensibility.

---

6. User Interface Overview

The application follows a clear navigation flow:

Splash Screen → Authentication → Main Screen with Bottom Navigation

Key screens include the league table, fixtures, match details, team planning, transfers, and profile/settings pages.
The interface uses a consistent dark theme, official FPL colors, animations, and clear visual indicators to enhance usability.

7. Conclusion and Future Work

FPL Assistant successfully delivers a modern and functional companion application for Fantasy Premier League managers.
The project evolved through three structured development phases, resulting in a complete system that integrates real-time football data, authentication, cloud synchronization, notifications, and a polished user interface.

Future enhancements may include:

- Advanced analytics and transfer recommendations.
- Historical performance charts and player comparison tools.
- Offline data support.
- Multi-language localization.

Overall, the project demonstrates effective application of **Flutter**, **Firebase**, **state management**, and **API integration**, while following sound software engineering principles appropriate for a final academic project.