



COMP4801 Final Year Project [2023/24] Final Report



KickInsights: A Football App with Machine Learning

| | | |
|-------------------|--------------|------------|
| Supervisor | Dr. Chim T W | |
| Student | Fung Hon Yin | 3035784497 |

Date of submission: 2024-04-26

Abstract

In the modern world of technology, there is an ever-rising demand from football fans' to conveniently access comprehensive football data. However, the mobile applications on the market suffer from common downsides including the inconsistency of data, a lack of functions for insights extraction, and the absence of an objective match-winner prediction system. In this project, a brand-new mobile app – KickInsights, has been developed to elevate fans' football experience by tackling the inefficiencies of current alternatives. It has brought comprehensive football data from online databases onto mobile devices and has provided tools for statistical comparison of football clubs and players, together with a data-driven match prediction system powered by machine learning algorithms.

This report will first introduce the project's background and motivation, then cover the objectives, scope of data, and core functionalities of the app. For the unique machine learning prediction system, step-by-step demonstrations will be provided. Moreover, the predicting power of the system will be compared with benchmarks to evaluate its effectiveness.

The methodologies for frontend and backend development will then be elaborated, in which platforms like React Native and Firestore are used. The data collection, wrangling, storing, and training process for the prediction system will also be explained. Finally, the challenges faced throughout the project and the future directions will be discussed.

Acknowledgment

I would like to express my sincere and heartfelt gratitude to Dr. TW Chim for supervising this final year project. His continuous support and guidance are crucial for this project. Without the feedback and encouragement, this project would not have proceeded as smoothly.

Moreover, I would like to thank my CAES 9542 lecturer Mr Gagandeep Singh for his help in the project plan and progress reports.

Table of Contents

| | Page |
|--|-----------|
| 1. Introduction | 1 |
| 1.1 Background | 1 |
| 1.2 Motivation | 1 |
| 1.3 Existing Football Applications | 4 |
| 1.4 Proposed Solution | 4 |
| 2. Project Objective and Scope | 5 |
| 2.1 Objective | 5 |
| 2.2 Scope of Data | 5 |
| 2.3 Core Functionalities | 5 |
| 2.3.1 League Table | 5 |
| 2.3.2 Club Information | 6 |
| 2.3.3 Club Comparison | 10 |
| 2.3.4 Player Information | 11 |
| 2.3.5 Player Comparison | 14 |
| 2.3.6 Matches Tab | 16 |
| 2.3.7 Match Centre | 17 |
| 2.4 Machine Learning Prediction System | 18 |
| 2.4.1 User Authentication | 18 |
| 2.4.2 Creating a model | 19 |
| 2.4.3 Applying a model | 24 |
| 2.4.4 Saving and Publishing a model | 25 |
| 2.4.5 Community Tab | 26 |
| 2.4.6 Saved Tab | 28 |
| 2.4.7 Interactivity of the prediction system | 30 |
| 2.4.8 Ability of the prediction system | 30 |
| 3. Methodology | 32 |

| | |
|--|-----------|
| 3.1 Frontend Development | 32 |
| 3.1.1 Prototype | 32 |
| 3.1.2 UI and UX | 33 |
| 3.1.3 Application Development | 34 |
| 3.2 Backend Development | 34 |
| 3.2.1 Data Collection | 34 |
| 3.2.2 Data Wrangling and Standard | 35 |
| 3.2.3 Machine Learning Prediction System | 37 |
| 3.4.5 Backend Infrastructure | 38 |
| 3.3 Difficulties and Mitigations | 39 |
| 4. Project Budget and Schedule | 41 |
| 4.1 Budget | 41 |
| 4.2 Schedule | 42 |
| 5. Future Works | 43 |
| 5.1 Comments for Community Models | 43 |
| 5.2 Offline Access | 43 |
| 5.3 Sorting and Filtering Functions | 43 |
| 5.4 AI chatbot | 43 |
| 6. Conclusion | 44 |
| 7. Project Resources | 45 |
| 8. References | 46 |
| 9. Appendices | 48 |

List of Figures

| | Page |
|---|------|
| <i>Figure 1. Club and Player information page from the LiveScore app</i> | 1 |
| <i>Figure 2. Club data from the FotMob app</i> | 2 |
| <i>Figure 3. Player data from the GOAL app</i> | 2 |
| <i>Figure 4. The sequence of actions to locate players' statistics on a web browser</i> | 2 |
| <i>Figure 5. Predictions and match result suggestions offered by the LiveScore app</i> | 3 |
| <i>Figure 6. Match insights from the FotMob app</i> | 3 |
| <i>Figure 7. Prediction polls from the OneFootball and the GOAL app</i> | 3 |
| <i>Figure 8. Table Tab</i> | 6 |
| <i>Figure 9. Home Table</i> | 6 |
| <i>Figure 10. Away Table</i> | 6 |
| <i>Figure 11. Table Legend</i> | 6 |
| <i>Figure 12. Refreshing the league table</i> | 6 |
| <i>Figure 13. Clubs Tab</i> | 7 |
| <i>Figure 14. Club Information page</i> | 7 |
| <i>Figure 15. A postponed game</i> | 7 |
| <i>Figure 16. Club Statistics page with each section expanded</i> | 8 |
| <i>Figure 17. Club Players page and player information</i> | 9 |
| <i>Figure 18. Comparing two teams in Club Comparison</i> | 10 |
| <i>Figure 19. Comparing a new aspect and a new club in Club Comparison</i> | 10 |
| <i>Figure 20. Players List</i> | 11 |
| <i>Figure 21. Players' Information page</i> | 11 |
| <i>Figure 22. Information page for a goalkeeper</i> | 12 |
| <i>Figure 23. Locating a player with the search and filter function</i> | 12 |
| <i>Figure 24. Locating a player with the club and position filter</i> | 13 |
| <i>Figure 25. Comparison interface</i> | 14 |
| <i>Figure 26. Adding a player for comparison</i> | 14 |
| <i>Figure 27. "Player Added" alert</i> | 14 |
| <i>Figure 28. "Compare list is full" alert</i> | 14 |
| <i>Figure 29. Comparing an aspect in Player Comparison</i> | 15 |
| <i>Figure 30. Matches Tab</i> | 16 |
| <i>Figure 31. Previous and future matches</i> | 16 |
| <i>Figure 32. Refreshing the match list</i> | 17 |

| | |
|---|----|
| <i>Figure 33. Delayed or rescheduled matches</i> | 17 |
| <i>Figure 34. Match Centre</i> | 18 |
| <i>Figure 35. Sign In, Sign Up, Sign Out, and the prediction home page</i> | 19 |
| <i>Figure 36. Prediction Tab for a guest user</i> | 19 |
| <i>Figure 37. “Data” page of the model creation form</i> | 20 |
| <i>Figure 38. Statistics input fields and list of the “Data” page</i> | 21 |
| <i>Figure 39. “Training” page of the model creation form</i> | 22 |
| <i>Figure 40. “Evaluation” page of the model creation form</i> | 23 |
| <i>Figure 41. Prediction match list</i> | 25 |
| <i>Figure 42. Prediction legend</i> | 25 |
| <i>Figure 43. Naming a model</i> | 26 |
| <i>Figure 44. Community Tab</i> | 26 |
| <i>Figure 45. Sorted model list in Community Tab</i> | 26 |
| <i>Figure 46. Filtered model list in Community Tab</i> | 27 |
| <i>Figure 47. Rating a model</i> | 27 |
| <i>Figure 48. Refreshing the model list</i> | 27 |
| <i>Figure 49. View and apply the published community models</i> | 28 |
| <i>Figure 50. Saved Tab before and after saving a community model</i> | 29 |
| <i>Figure 51. The interface of a model inside the Saved Tab</i> | 29 |
| <i>Figure 52. Model performance graphs of different algorithms and parameters</i> | 30 |
| <i>Figure 53. Prototype of KickInsights created with Figma</i> | 33 |
| <i>Figure 54. The system architecture of KickInsights</i> | 34 |
| <i>Figure 55. Cloud functions for data wrangling</i> | 36 |
| <i>Figure 56. Cloud scheduling for functions</i> | 36 |
| <i>Figure 57. Training status during the training progress</i> | 37 |
| <i>Figure 58. Loading screen of KickInsights</i> | 38 |

List of Tables

| | Page |
|---|------|
| <i>Table 1. Comparison among popular football apps and KickInsights</i> | 4 |
| <i>Table 2. Comparison of clicks required to locate a club's statistic among popular football apps and KickInsights</i> | 9 |
| <i>Table 3. Comparison of the average clicks required to compare club statistics among popular football apps and KickInsights</i> | 11 |
| <i>Table 4. Comparison of clicks required to locate a player's statistic among popular football apps and KickInsights</i> | 13 |
| <i>Table 5. Comparison of the average clicks required to compare player statistics among popular football apps and KickInsights</i> | 15 |
| <i>Table 6. Training duration for Machine Learning models (in seconds)</i> | 31 |
| <i>Table 7. User Interfaces of KickInsights on various devices</i> | 40 |
| <i>Table 8. Budget for the project</i> | 41 |
| <i>Table 9. Project Schedule Timetable</i> | 42 |

List of Abbreviations

| Abbreviation | Definition |
|--------------|------------------------------------|
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| BaaS | Backend-as-a-Service |
| GiB | Gibibyte |
| GUI | Graphical User Interface |
| IDE | Integrated Development Environment |
| iOS | iPhone Operating System |
| JSON | JavaScript Object Notation |
| KNN | K-Nearest Neighbor |
| MB | Megabyte |
| ML | Machine Learning |
| SVM | Support Vector Machine |
| UI | User Interface |
| URL | Uniform Resource Locator |
| UTC | Coordinated Universal Time |
| UX | User Experience |
| vCPU | Virtual Central Processing Unit |

1. Introduction

1.1 Background

In the ever-evolving world of football, where there is enormous passion from millions of fans across the globe, there is a growing demand for convenient yet comprehensive access to football data. Supporters crave detailed statistics and insights to keep up with their favourite clubs and players [1]. In particular, 46% of Generation Z's audience regularly use apps while watching sports, highlighting the rise of digital connectivity with sports content [2].

Hence, under today's technological revolution, a variety of mobile apps with popular options including Livescore, OneFootball, and FotMob, have been developed as virtual tools for football fans [3].

1.2 Motivation

Despite the availability of football applications on mobile platforms, most of the existing options share the following downsides.

Firstly, a lack of functions for insights extraction from static data. These apps only provide pages loaded with fixed information with no tools for evaluation. For example, the LiveScore app provides a club or player information page listing their statistics, as shown in Figure 1. However, users cannot take advantage of the data provided to perform further evaluation, including seasonal comparison between clubs, or cross-club player comparison. Without functions for further data exploration, under-utilization of the data would have resulted.

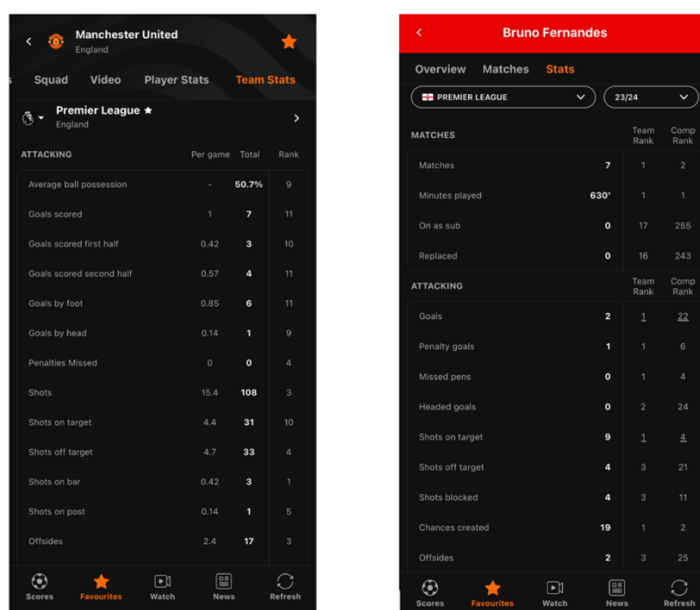


Figure 1. Screenshots from the LiveScore app, showing the Club and Player information page

Secondly, inconsistency and low comprehensiveness of the data provided. For instance, Figure 2 shows that the FotMob app only scratches the surface and offers general club information like points and goal types, while Figure 3 shows that the GOAL app offers only a few columns of player data represented by symbols without clear notation. This leads to a poor overall experience as users are unsure whether the data they are looking for is available in the app.

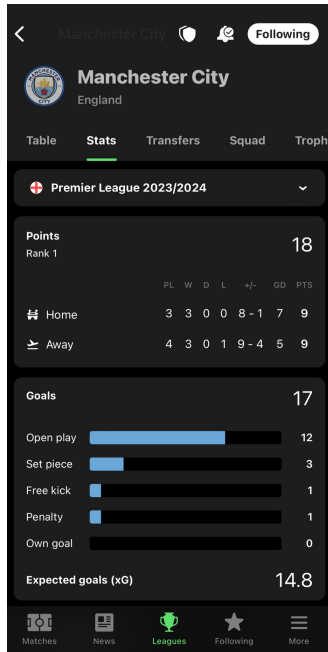


Figure 2. Screenshot from the FotMob app, showing the Club data

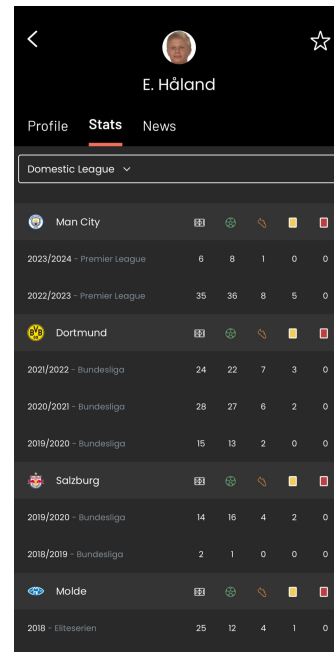


Figure 3. Screenshot from the GOAL app, showing the Player data

For users who desire detailed in-game statistics, like a player's long pass completion rate in the current season, or a goalkeeper's launch rate during goal kicks, it leaves them no choice but to visit large-scale online databases, which is time-consuming and difficult to navigate on a mobile browser, as shown in Figure 4 using iOS's Safari as an example.

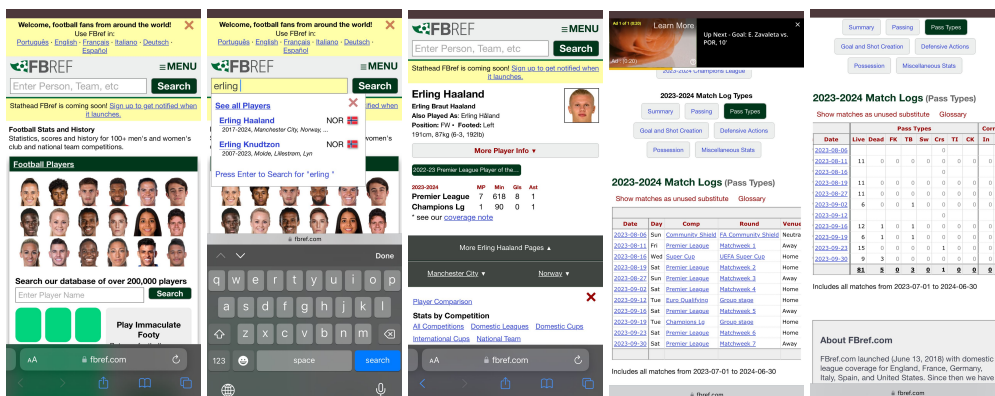


Figure 4. Screenshots from iOS's Safari, showing the sequence of actions to locate players' statistics on a web browser

Thirdly, the absence of an objective match-winner prediction system. Although some apps with prediction features, like LiveScore, offer expert tips and suggestions as shown in Figure 5, they are subjective and prone to errors.

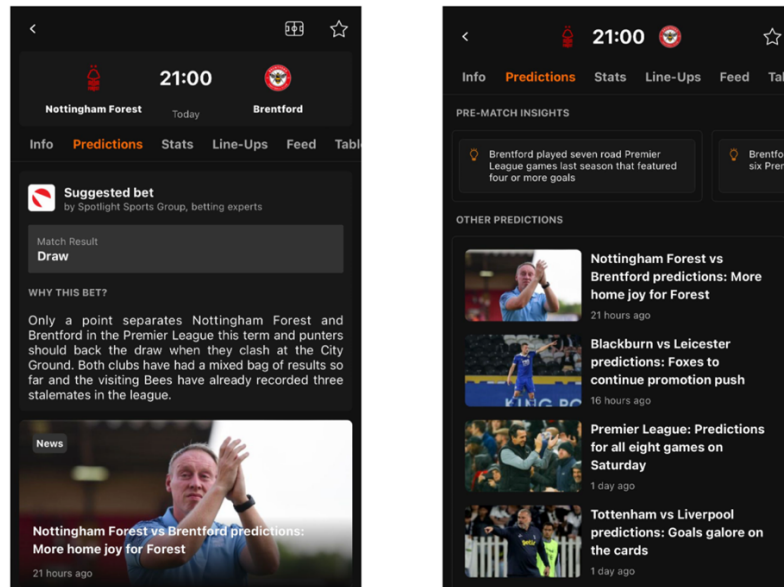


Figure 5. Screenshots from the LiveScore app, showing the predictions and match result suggestions offered by the app

Some apps, like FotMob, only provide unorganized selective insights with a low reference value to users, as shown in Figure 6, while the OneFootball and the GOAL app create polls as an indicator of sentiment, as shown in Figure 7. However, polls may be subject to bias [4]. One possible reason may be due to the different amount of supporters for each club.

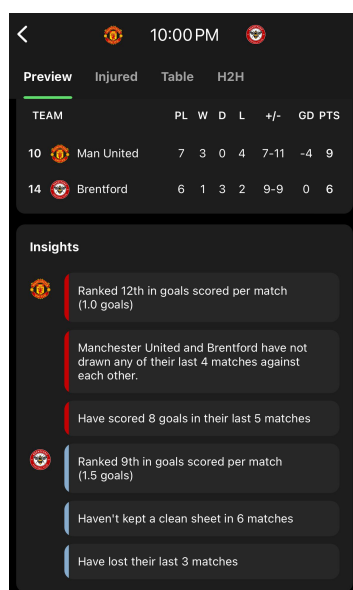


Figure 6. Screenshot from the FotMob app, showing the Match insights

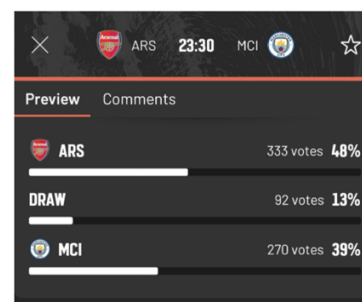
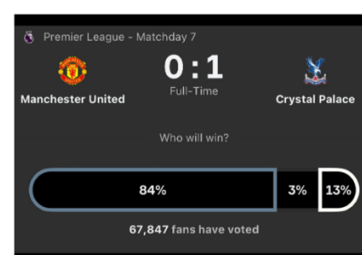


Figure 7. Screenshots from the OneFootball and the GOAL app, showing the prediction polls

1.3 Existing Football Applications

As shown in Table 1, even the best alternatives for obtaining club and player data, LiveScore and OneFootball, only partially provide the data. Moreover, not one of these apps provides an interface for direct club or player comparison, implying that in order to compare these statistics, users would have to navigate between pages to make comparisons in a one-by-one manner. Lastly, match prediction by these apps (if offered) is not statistical-based in general.



| Features / Functions | AiScore | FotMob | GOAL | LiveScore | One-Football | KickInsights |
|--------------------------------------|---------|--------|------|-----------|--------------|--------------|
| In-depth team data [1] | X | X | X | ✓ | ✓ | ✓ |
| In-depth player data [2][3] | ✓ | ✓ | X | ✓ | ✓ | ✓ |
| Team/Player Comparison | X | X | X | X | X | ✓ |
| Match prediction (opinion based) | ✓ | X | ✓ | ✓ | ✓ | ✓ |
| Match prediction (statistical based) | X | ✓ | X | ✓ | X | ✓ |

✓ - Provided , ✓ - Partially Provided , X - Not Provided

[1] See Appendix A

[2][3] See Appendix B, C

Table 1. Comparison among popular football apps and KickInsights (as of 2024-04-22)

1.4 Proposed Solution

The goal of this project is to develop KickInsights, a brand-new mobile app that brings comprehensive football data onto mobile devices. Football enthusiasts are no longer required to access in-depth statistics in online databases via personal computers, or by going through time-consuming and repetitive navigation on mobile web browsers.

In addition to offering the data, KickInsights also allows users to extract insights from them via interfaces for statistical comparison, and even further utilize them with a data-driven match prediction system powered by ML algorithms.

In the preceding Section 2, the objectives, scope, and design for the core functionalities of the app will be discussed, followed by the methodology for frontend and backend development in Section 3. Finally, the project budget, schedule, and future directions will be explained in Sections 4 and 5.

2. Project Objective and Scope

2.1 Objective

This project aims to develop KickInsights to solve the inefficiencies of current mobile application alternatives, and to integrate ML components with the app to enhance the utilization of the football data provided. To achieve the outcome, the scope of data will be discussed in Section 2.2 and the core functionalities of the app will be elaborated upon in Section 2.3. Section 2.4 then provides a step-by-step demonstration of the machine learning prediction system, before evaluating its interactivity, training duration, and predicting power.

2.2 Scope of Data

This project focuses on the Premier League for three reasons. Firstly, it is the most popular football league in the world followed by an estimated 3.2 billion global fanbase [5]. Secondly, it has extensive and reliable data across multiple online databases and APIs, including FBREF and FootApi, readily available for scraping. Thirdly, it has been consistently generating high-quality match content for the past decades, assuring that the data possess significant reference value representing football as a whole.

2.3 Core Functionalities

This section introduces the functionalities provided in four tabs in KickInsights, namely the Table, Clubs, Players, and Matches tab. Visuals in this section are captured from the KickInsights app launched on the native environment (iOS). Data displayed in the app is as of April 24th, 2024.

2.3.1 League Table

The live Premier League table of the current season is provided in the Table tab in the menu bar, as shown in Figure 8, so that users can easily access it. Each row contains a club's ranking, name, matches played (PL), wins(W), draws (D), losses(L), goal difference (GD), and points (Pts). To evaluate the home and away performance of each club, users can select respective tables on the navigation bar on top of the screen, as shown in Figures 9 and 10. Moreover, when a club is clicked, users will be redirected to the information page of that club (see Section 2.3.2).

20:37 Premier League 2023/24

| # | CLUB | PL | W | D | L | GD | Pts |
|----|----------------|----|----|----|----|-----|-----|
| 1 | Arsenal | 34 | 24 | 5 | 5 | 56 | 77 |
| 2 | Liverpool | 33 | 22 | 8 | 3 | 43 | 74 |
| 3 | Man City | 32 | 22 | 7 | 3 | 44 | 73 |
| 4 | Aston Villa | 34 | 20 | 6 | 8 | 21 | 66 |
| 5 | Spurs | 32 | 18 | 6 | 8 | 16 | 60 |
| 6 | Newcastle | 32 | 15 | 5 | 12 | 17 | 50 |
| 7 | Man United | 32 | 15 | 5 | 12 | -1 | 50 |
| 8 | West Ham | 34 | 13 | 9 | 12 | -9 | 48 |
| 9 | Chelsea | 32 | 13 | 8 | 11 | 4 | 47 |
| 10 | Brighton | 32 | 11 | 11 | 10 | 2 | 44 |
| 11 | Wolves | 33 | 12 | 7 | 14 | -7 | 43 |
| 12 | Fulham | 34 | 12 | 6 | 16 | -4 | 42 |
| 13 | Bournemouth | 33 | 11 | 9 | 13 | -12 | 42 |
| 14 | Crystal Palace | 33 | 9 | 9 | 15 | -14 | 36 |
| 15 | Brentford | 34 | 9 | 8 | 17 | -7 | 35 |
| 16 | Everton | 33 | 10 | 8 | 15 | -14 | 30 |

Figure 8. Table Tab

20:37 Premier League 2023/24

| # | CLUB | PL | W | D | L | GD | Pts |
|----|----------------|----|----|---|---|----|-----|
| 1 | Liverpool | 17 | 13 | 3 | 1 | 28 | 42 |
| 2 | Man City | 17 | 12 | 5 | 0 | 29 | 41 |
| 3 | Arsenal | 17 | 13 | 2 | 2 | 28 | 41 |
| 4 | Aston Villa | 17 | 12 | 2 | 3 | 20 | 38 |
| 5 | Newcastle | 17 | 11 | 3 | 3 | 23 | 36 |
| 6 | Spurs | 16 | 12 | 0 | 4 | 13 | 36 |
| 7 | Fulham | 17 | 9 | 1 | 7 | 11 | 28 |
| 8 | Chelsea | 16 | 8 | 4 | 4 | 10 | 28 |
| 9 | Brighton | 15 | 7 | 6 | 2 | 9 | 27 |
| 10 | Man United | 15 | 8 | 2 | 5 | 1 | 26 |
| 11 | West Ham | 17 | 6 | 7 | 4 | 1 | 25 |
| 12 | Wolves | 16 | 7 | 3 | 6 | -2 | 24 |
| 13 | Bournemouth | 17 | 6 | 6 | 5 | -3 | 24 |
| 14 | Brentford | 17 | 5 | 6 | 6 | -3 | 21 |
| 15 | Nott'm Forest | 17 | 5 | 5 | 7 | 0 | 20 |
| 16 | Crystal Palace | 16 | 5 | 4 | 7 | 0 | 19 |

Figure 9. Home Table

20:37 Premier League 2023/24

| # | CLUB | PL | W | D | L | GD | Pts |
|----|----------------|----|----|---|----|-----|-----|
| 1 | Arsenal | 17 | 11 | 3 | 3 | 28 | 36 |
| 2 | Man City | 15 | 10 | 2 | 3 | 15 | 32 |
| 3 | Liverpool | 16 | 9 | 5 | 2 | 15 | 32 |
| 4 | Aston Villa | 17 | 8 | 4 | 5 | 1 | 28 |
| 5 | Spurs | 16 | 6 | 6 | 4 | 3 | 24 |
| 6 | Man United | 17 | 7 | 3 | 7 | -2 | 24 |
| 7 | West Ham | 17 | 7 | 2 | 8 | -10 | 23 |
| 8 | Wolves | 17 | 5 | 4 | 8 | -5 | 19 |
| 9 | Chelsea | 16 | 5 | 4 | 7 | -6 | 19 |
| 10 | Everton | 17 | 5 | 4 | 8 | -14 | 19 |
| 11 | Bournemouth | 16 | 5 | 3 | 8 | -9 | 18 |
| 12 | Brighton | 17 | 4 | 5 | 8 | -7 | 17 |
| 13 | Crystal Palace | 17 | 4 | 5 | 8 | -14 | 17 |
| 14 | Brentford | 17 | 4 | 2 | 11 | -4 | 14 |
| 15 | Newcastle | 15 | 4 | 2 | 9 | -6 | 14 |
| 16 | Fulham | 17 | 3 | 5 | 9 | -15 | 14 |

Figure 10. Away Table

A legend is provided under the table, as shown in Figure 11, to indicate the league positions that will be qualified for the Champions League or the Europa League in the following season or those that would be relegated to a lower division. To get live updates during matchdays, users can refresh the table by pulling it downwards, as shown in Figure 12.

16 Everton 33 10 8 15 -14 30

17 Nott'm Forest 34 7 9 18 -18 26

18 Luton 34 6 7 21 -28 25

19 Burnley 34 5 8 21 -32 23

20 Sheffield Utd 33 3 7 23 -57 16

Qualification / Relegation

- Champions League group stage
- Europa League group stage
- Relegation

Figure 11. Table Legend

20:37 Premier League 2023/24

| # | CLUB | PL | W | D | L | GD | Pts |
|---|-------------|----|----|---|----|----|-----|
| 1 | Arsenal | 34 | 24 | 5 | 5 | 56 | 77 |
| 2 | Liverpool | 33 | 22 | 8 | 3 | 43 | 74 |
| 3 | Man City | 32 | 22 | 7 | 3 | 44 | 73 |
| 4 | Aston Villa | 34 | 20 | 6 | 8 | 21 | 66 |
| 5 | Spurs | 32 | 18 | 6 | 8 | 16 | 60 |
| 6 | Newcastle | 32 | 15 | 5 | 12 | 17 | 50 |

20:56 Premier League 2023/24

| # | CLUB | PL | W | D | L | GD | Pts |
|---|-------------|----|----|---|---|----|-----|
| 1 | Arsenal | 34 | 24 | 5 | 5 | 56 | 77 |
| 2 | Liverpool | 33 | 22 | 8 | 3 | 43 | 74 |
| 3 | Man City | 32 | 22 | 7 | 3 | 44 | 73 |
| 4 | Aston Villa | 34 | 20 | 6 | 8 | 21 | 66 |

Figure 12. Refreshing the league table

2.3.2 Club Information

In the “Clubs” tab, there is a list containing all 20 clubs in the current Premier League season, as shown in Figure 13. Their home stadium, city, establishment year, and logo are displayed.

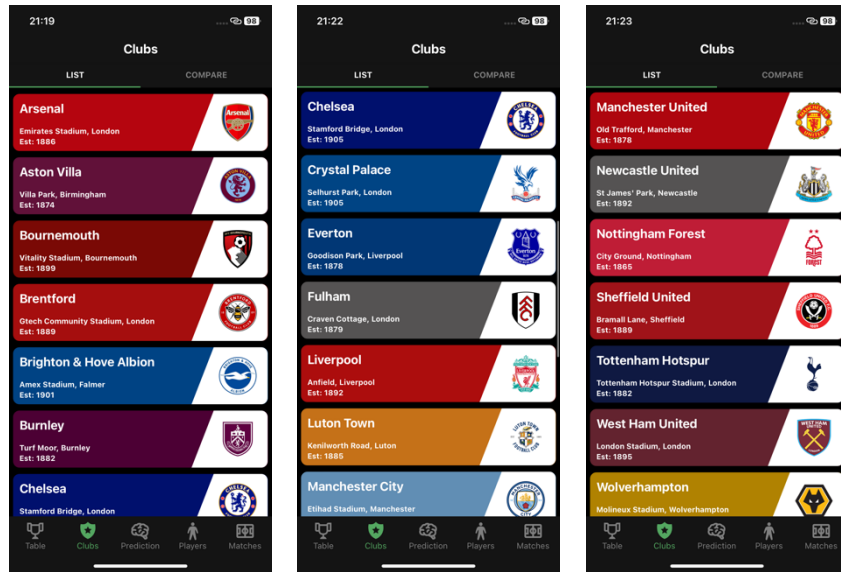


Figure 13. Clubs Tab

Users can click on a club for its information page, which was designed with the club's theme color. The page includes the overview, the kits, the last and the next 3 games of the club, as shown in Figure 14 using Liverpool as an example. For the "Kits" section, the home, away, and the third kit in the current season are displayed. For the "Last 3 Games" section, the scorelines are colored in green, yellow, or red, corresponding to a win, draw, or loss for the team respectively. For the "Next 3 Games" section, the starting times of the games are shown. Figure 15 illustrates that if a game is postponed or rescheduled, it will be reflected in the section.

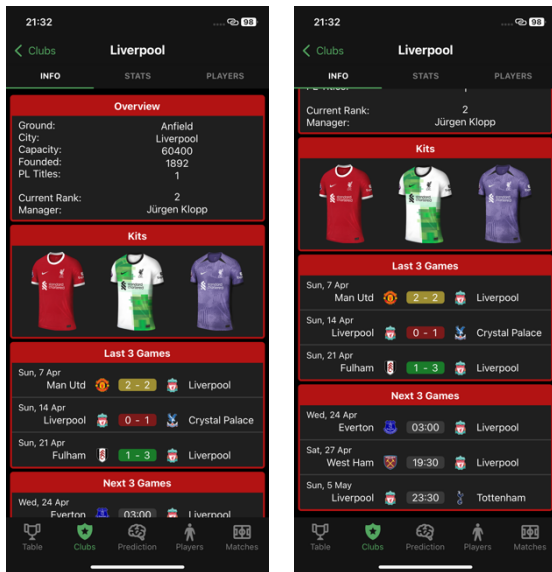


Figure 14. Club Information page

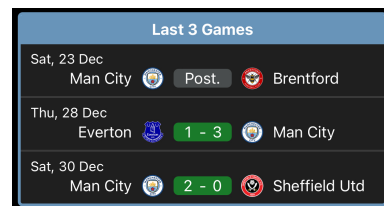


Figure 15. A postponed game

The “Stats” page provides a club’s in-depth statistics in the current season, which includes a wide range of football data segmented into General, Shooting, Passing, Pass Types, Goal and Shot Creation, Defensive Actions, Possession, Goalkeeping, and Miscellaneous data, as shown in Figure 16 using Chelsea as an example. Altogether, the page contains 117 rows of data, the most comprehensive among existing mobile apps, yet remains organized within the interface. Navigation actions are minimized as users can directly expand the section they are interested in with one click, instead of a series of clicking, scrolling, and zooming actions as required on a web database.



Figure 16. Club Statistics page with each section expanded

The “Players” page stores players of the club sorted according to their playing positions, from Forwards (FW), Midfielders (MF), Defenders (DF) to Goalkeepers (GK), as shown in Figure 17 using Arsenal as an example. Users can view the players’ general information, like kit number, nationality, age, height, and market value, by tapping on their row.

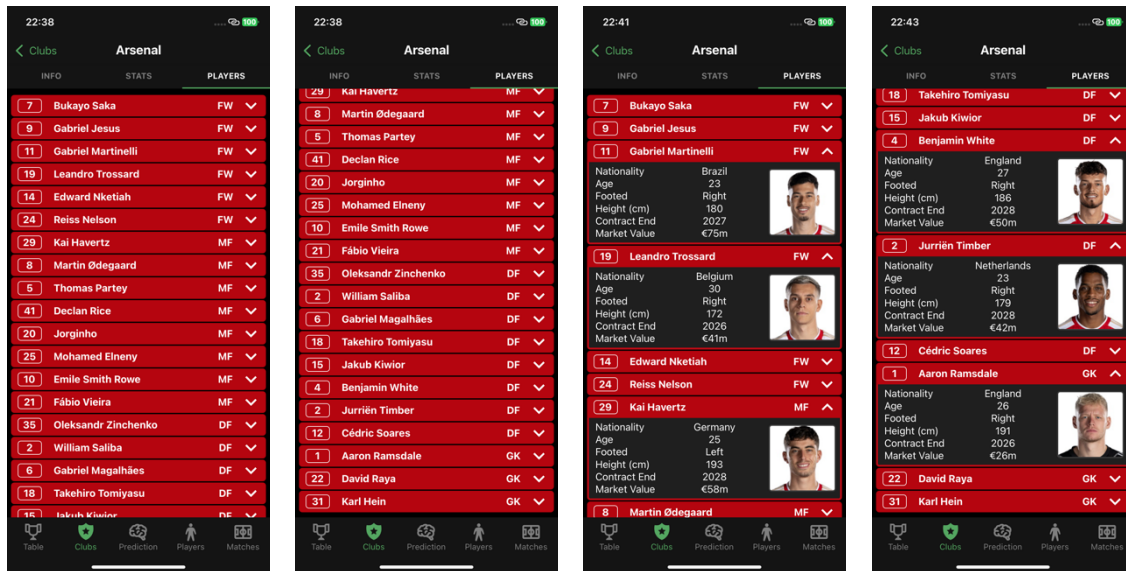


Figure 17. Club Players page and player information

To conclude, the “Clubs” tab together with the “Information”, “Statistics”, and “Player” pages provide a simple navigation to access club details and statistics. Table 2 compares the number of clicks required to locate a club’s statistics starting from the apps’ home page, in which KickInsights has equalled the best-performing alternative, FotMob, while offering much more comprehensive data.








| |  |  |  |  |  |  |  |
|--|---|---|---|---|---|---|---|
| Clicks required to locate a club's statistics | Browser | AiScore | FotMob | GOAL | LiveScore | One-Football | KickInsights |
| Minimum | 6 | NA | 4 | NA | 6 | 6 | 4 |
| Average (Approximate) | 7 | NA | 5 | NA | 7 | 7 | 5 |

Table 2. Comparison of clicks required to locate a club’s statistic among popular football apps* and KickInsights

* AiScore and GOAL do not provide club statistics.

2.3.3 Club Comparison

Next to the club list on the top navigation bar is the page for club statistics comparison. Users can pick any two teams in the premier league from the logos panel and directly compare their seasonal data in a side-by-side manner, as shown in Figure 18 using Brighton and Luton Town as examples. Data is segmented in the same way as in the club statistics page.

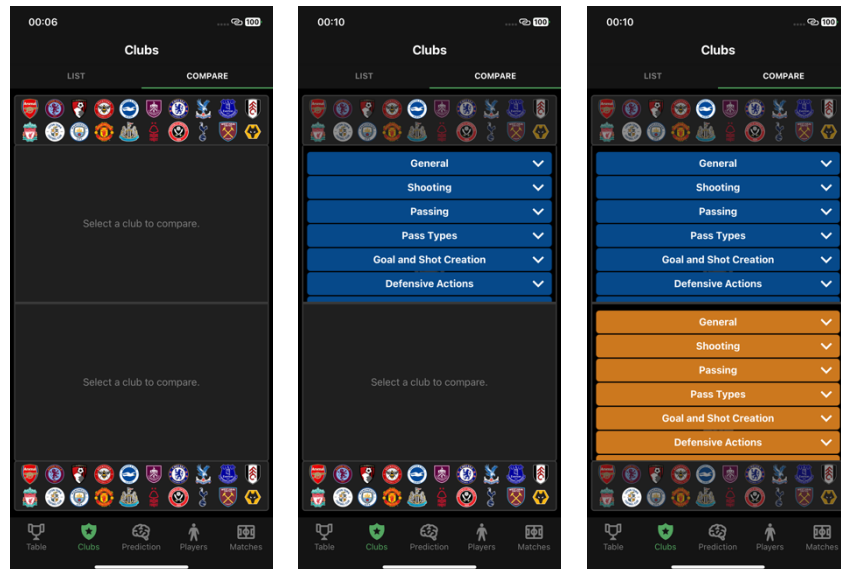


Figure 18. Compare two teams in Club Comparison

To compare an aspect between the two clubs, users can expand their respective sections. Another aspect or another club can be selected for comparison with just two more clicks, as shown in Figure 19, which switches from comparing “Shooting” to “Pass types”, and then from Luton Town to Sheffield United at the bottom half of the interface.

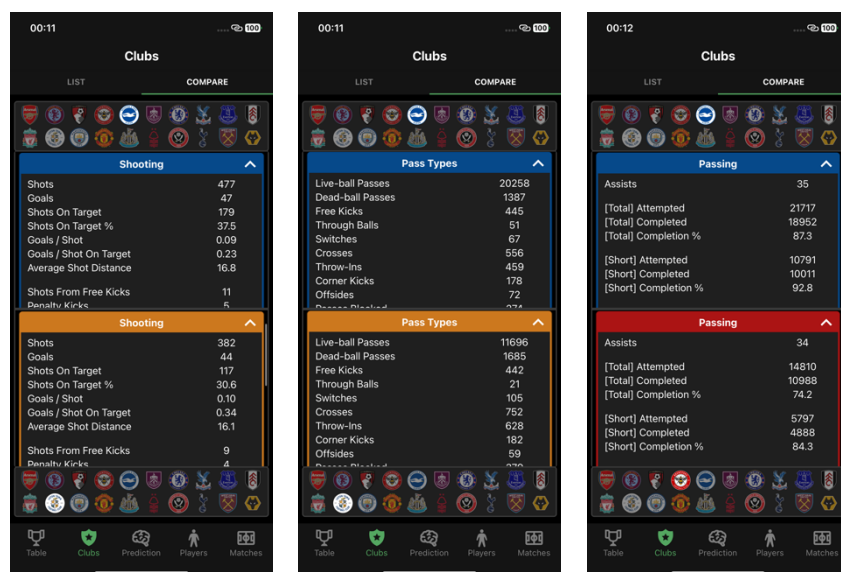


Figure 19. Comparing a new aspect and a new club in Club Comparison

This single-screen interface not only supports comparing multiple attributes on the same screen, but also requires only two extra clicks to compare new statistics, avoiding the need to switch between tabs or going back and forth for each new comparison, as in other football apps and the mobile browser. Therefore, KickInsights significantly reduces the effort needed to compare seasonal statistics between two clubs in the premier league, as summarized in Table 3.



| Average clicks required to compare club statistics | Browser | AiScore | FotMob | GOAL | LiveScore | One-Football | KickInsights |
|--|---------|---------|--------|------|-----------|--------------|--------------|
| First comparison | 14 | NA | 10 | NA | 14 | 14 | 6 |
| Second comparison | 28 | NA | 20 | NA | 28 | 28 | 8 |
| Third comparison | 42 | NA | 30 | NA | 42 | 42 | 10 |

Table 3. Comparison of the average clicks required to compare club statistics among popular football apps* and KickInsights

* AiScore and GOAL do not provide club statistics.

2.3.4 Player Information

In the “Players” tab, there is a list containing all 533 Premier League players, as of April 24th, 2024, sorted by alphabetical order of their clubs’ names, and then by their playing positions. For each player, the club, kit number, and playing position are displayed, as shown in Figure 20. By tapping on the player's row in the list, users can access the player information page, which contains both the general information and in-depth statistics of a player in the current season, as shown in Figure 21, using Declan Rice from Arsenal as an example.

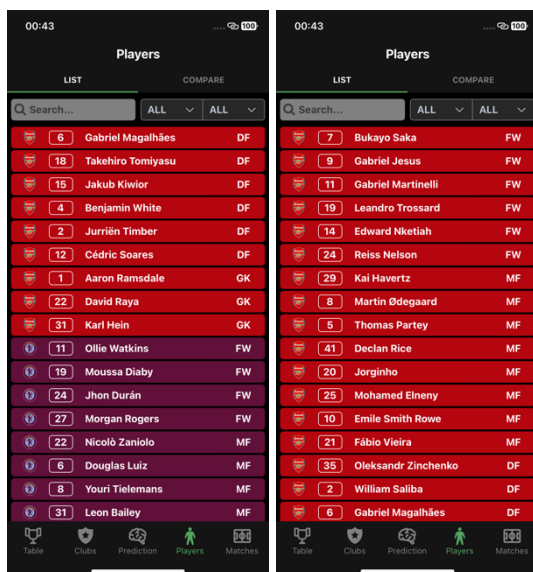


Figure 20. Players List



Figure 21. Players' Information page

For goalkeepers, the information page contains specific statistics like “Goalkeeping” and “Advanced Goalkeeping” on top of the standard statistics for on-field players. Aaron Ramsdale from Arsenal is used as an example in Figure 22.

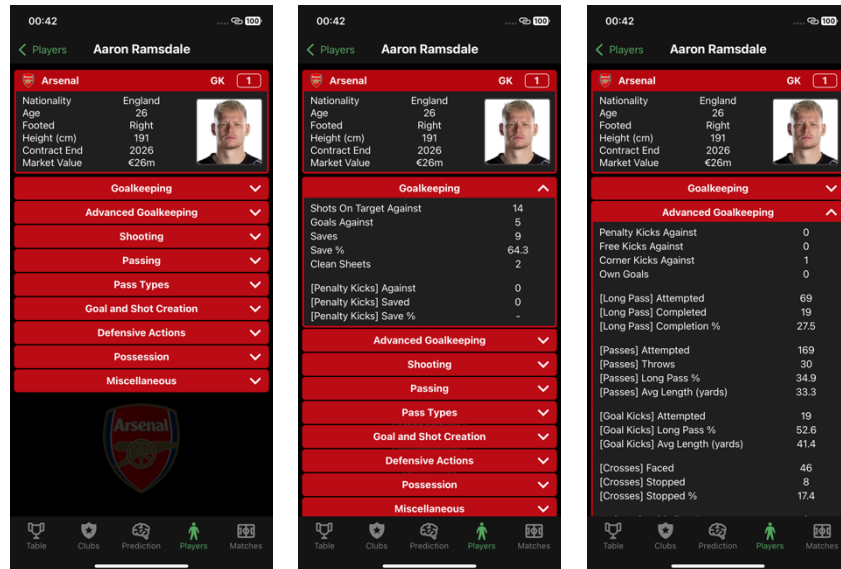


Figure 22. Information page for a goalkeeper

To facilitate navigation, the “Players” tab provides the search and filter functions, which are both absent in other existing football apps. These functions can be combined in any sequence as shown in Figure 23. For example, by inputting “tom” into the search bar, every player whose name contains “tom” will appear on the list. Then, any player positions like “Defender” (DF) can be shortlisted with the position filter.

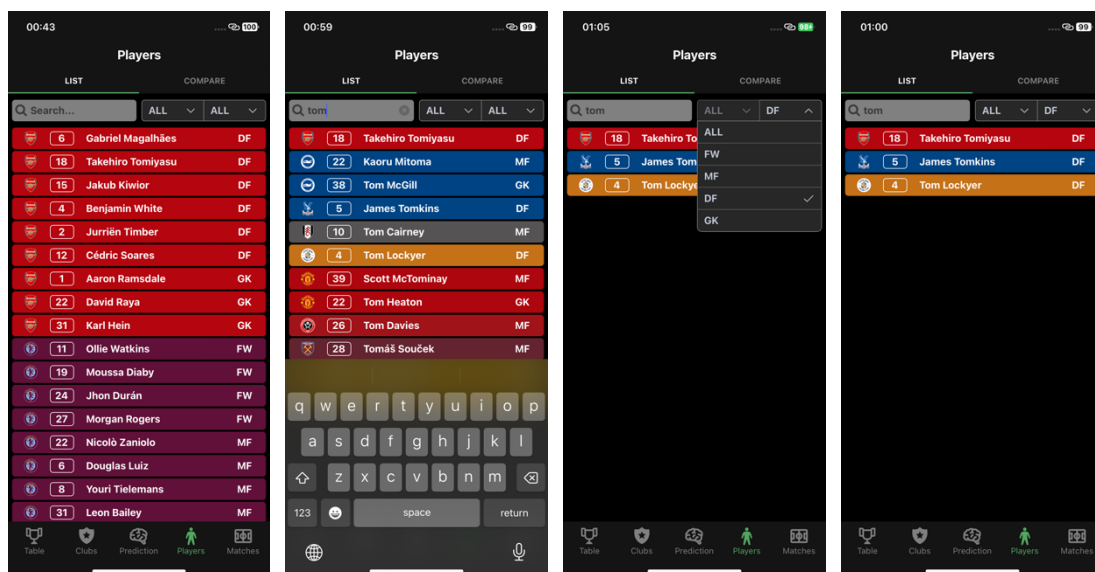


Figure 23. Locating a player with the search and filter function

Since there are two transfer windows in each season when players join new clubs, users may not be aware of the latest club a transferred player belongs to. While it may be difficult to locate the player page in other apps without explicitly searching for transfer news on the internet, users of KickInsights can easily locate the player with the search function.

Moreover, the club and position filters allow users to locate any player in just four clicks, in which two are used for filtering the club and two are used for filtering the position, as illustrated in Figure 24, where the “Manchester City” (MCI) and “Midfielders” (MF) filters have been applied.

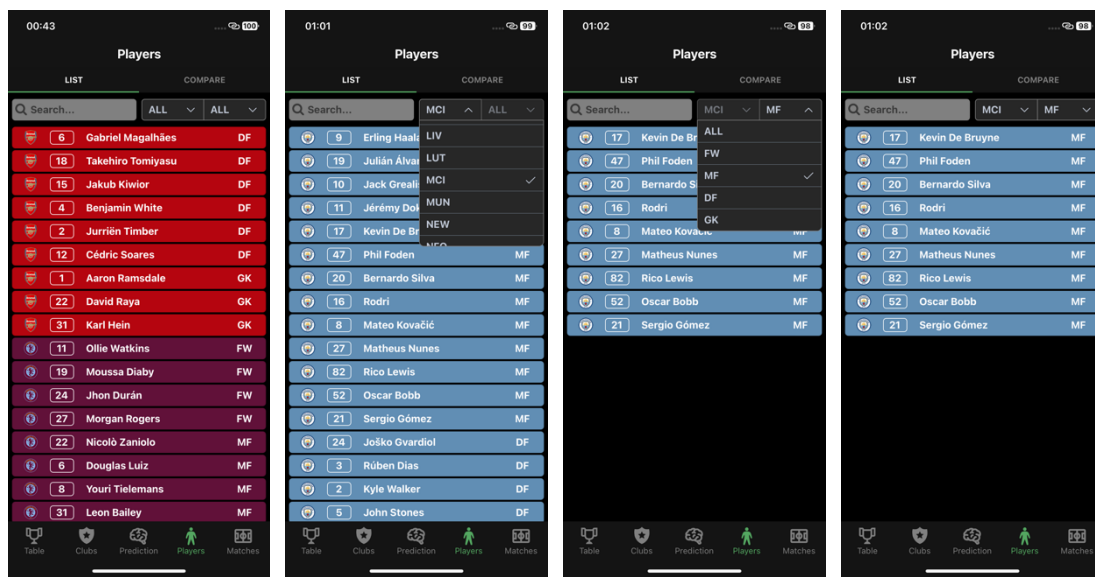


Figure 24. Locating a player with the club and position filter

To conclude, the “Players” tab offers flexibility and ease for locating the information pages of all players, which contain comprehensive player statistics specific to various playing positions. As summarized in Table 4, KickInsights beats other football apps in terms of the number of clicks required to locate a player’s statistics, starting from the home page.



| Clicks required to locate a player's statistics | Browser | AiScore | FotMob | GOAL | LiveScore | One-Football | KickInsights |
|---|---------|---------|--------|------|-----------|--------------|--------------|
| Minimum | 6 | 6 | 7 | NA | 7 | 6 | 3 |
| Average (Approximate) | 12 | 8 | 9 | NA | 9 | 9 | 7 |

Table 4. Comparison of clicks required to locate a player’s statistic among popular football apps* and KickInsights

* GOAL does not provide player statistics.

2.3.5 Player Comparison

Similar to the club comparison function, an interface is also provided in the “Players” tab for direct comparison of player statistics in the current season, as shown in Figure 25. Thus, cross-club and cross-positional comparisons can be made on a single screen. Comparison of players in the same club will also be supported. To add a player for comparison, simply long-press the player in the player list, as shown in Figure 26 using Kevin De Bruyne from Manchester City and Bruno Fernandes from Manchester United as an example.

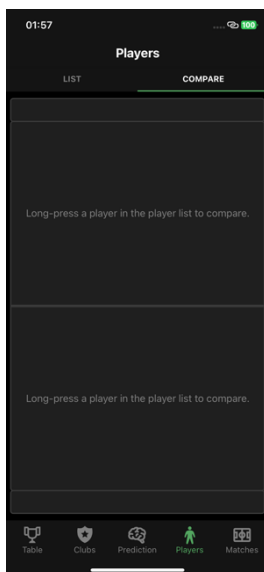


Figure 25. Comparison interface

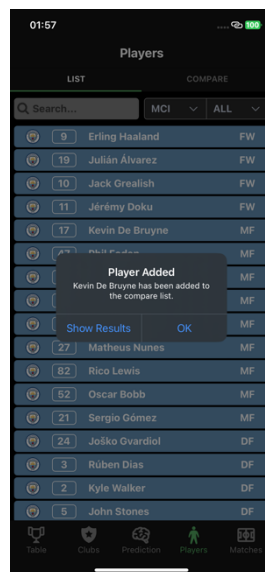
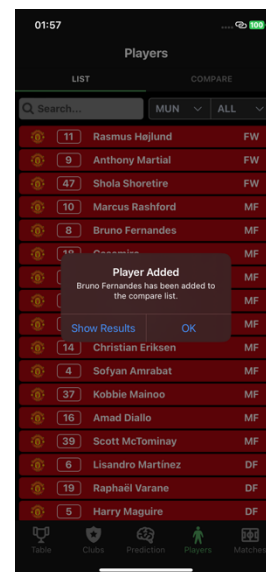


Figure 26. Adding a player for comparison



As shown in Figure 27, besides manually navigating to the comparison interface, users can also view the comparison results by clicking the alert. Figure 28 illustrates that if there are already two players being compared, the alert will require the user to replace one of them.

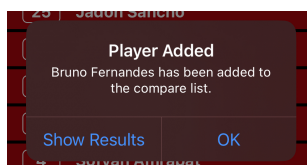


Figure 27. “Player Added” alert

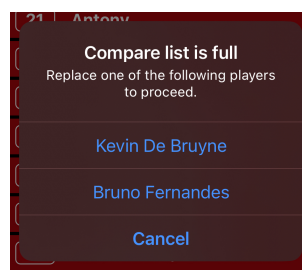


Figure 28. “Compare list is full” alert

Figure 29 shows the comparison interface after two players were added. A side-by-side frontend design consistent with the club comparison screen has been implemented to enhance the UX. To compare an aspect between the two players or switch to another aspect, the respective sections can be expanded with two clicks.

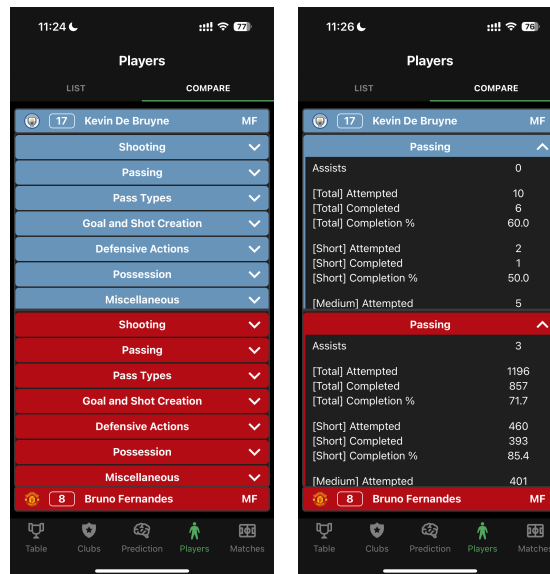


Figure 29. Comparing an aspect in Player Comparison

Similar to the club comparison function, this interface supports comparing multiple attributes on the same screen and avoids the need to switch between tabs, as required in other football apps. Therefore, once again, KickInsights greatly reduces the effort needed to compare seasonal statistics between two players in the premier league, as summarized in Table 5.








| |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
| Average clicks required to compare player statistics | Browser | AiScore | FotMob | GOAL | LiveScore | One-Football | KickInsights |
| First comparison | 24 | 16 | 18 | NA | 18 | 18 | 14 |
| Second comparison | 48 | 32 | 36 | NA | 36 | 36 | 16 |
| Third comparison | 72 | 48 | 52 | NA | 52 | 52 | 18 |

Table 5. Comparison of the average clicks required to compare player statistics among popular football apps* and KickInsights

* GOAL does not provide player statistics.

2.3.6 Matches Tab

All matches in the current season from matchweek 1 to 38 will be stored in the match tab. Matches during the live matchweek (matchweek 34 as of April 24th, 2024) will be displayed by default, as shown in Figure 30. With the top navigation bar, users can view previous matches (on or before matchweek 33) and future matches (on or after matchweek 35), as shown in Figure 31. In each matchweek, matches are sorted according to their starting date and time. Matches in the past will be labelled as full-time (FT), while upcoming matches will be shown with their starting times.

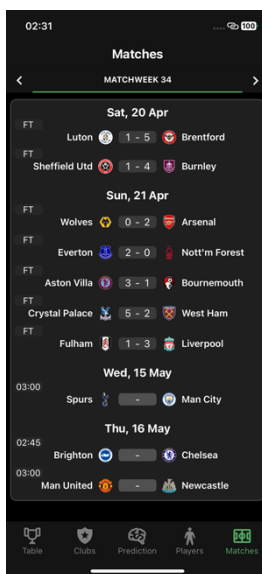


Figure 30. Matches Tab

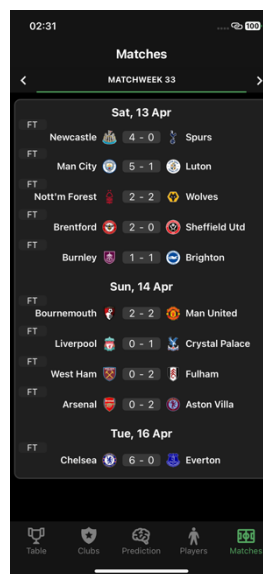
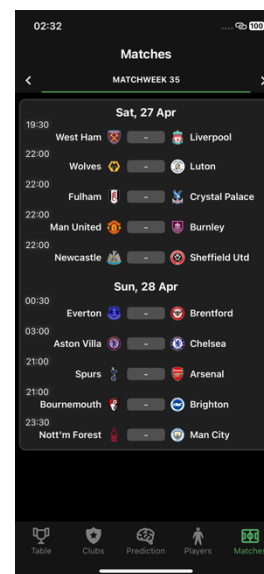


Figure 31. Previous and future matches



To get live updates during matchdays, users can refresh the match list by pulling it downwards, as shown in Figure 32. In addition, to provide a systematic manner for displaying matches, matches that are delayed or rescheduled due to circumstances like unfavourable weather or event crashes would remain in their original matchweeks. For example, the match in matchweek 26 between Chelsea and Spurs would originally be played on February 27th, 2024. After being rescheduled to May 3rd, 2024, it could still be located inside matchweek 26, as shown in Figure 33. With this approach, no matchweek's page will be packed with matches from previous matchweeks.

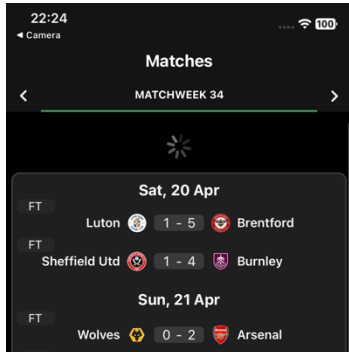


Figure 32. Refreshing the match list

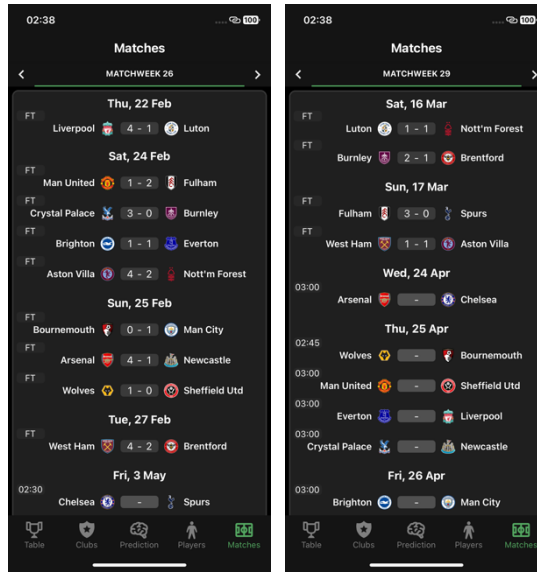


Figure 33. Delayed or rescheduled matches

2.3.7 Match Centre

For each match, users can access the match centre by clicking the respective match in the “Matches” tab. It contains three pages as shown in Figure 34, using Manchester United versus Liverpool as an example. The information page includes the general details of the match, like the date, time, and venue, with the managers and captains of both teams. Their league standings are provided and users can navigate to the club information page (see Section 2.3.2) by clicking on the team. Moreover, the officials refereeing the match are listed. The statistics page summarizes the match’s scoreline, goal scorers, scoring timestamp, and head-to-head in-game statistics, including corners, interceptions, long balls, etc. Various major attributes like possession, passing accuracy, shots on target, and saves are provided with bars indicating the total and successful count of that attribute. Finally, the squad page outlines the formation, first team, and substitute players selected by both sides for the match.

For matches in the future, the match centre only includes information announced prior like the matchweek, date, time, etc. Users can navigate to the prediction page via the “Predict” button on the top-right corner, to apply a ML model for predicting the match’s winner, as explained in Section 2.3.8.

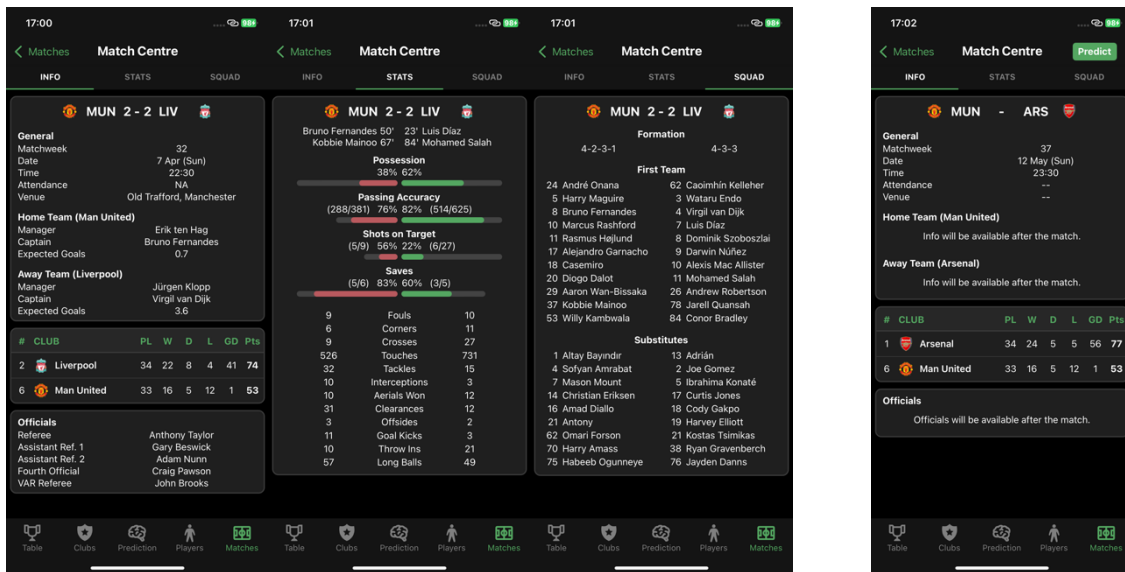


Figure 34. Match Centre. Left: past matches. Right: Future matches.

With this simple yet comprehensive interface, users can conveniently explore past matches with the aid of statistics visualization, and predict any future matches directly.

2.4 Machine Learning Prediction System

This section demonstrates the unique and interactive functionalities provided in the Prediction tab, which is the app's default home page. It starts from user authentication, creating and applying an ML model, to viewing and saving models published by other users. Finally, the interactivity and ability of the prediction system will be discussed. Visuals in this section are captured from the KickInsights app launched on the native environment (iOS). Data displayed in the app is as of April 24th, 2024.

2.4.1 User Authentication

Before creating an ML model or saving others' models, users should first sign up for a KickInsights account or sign in to an existing one. Persistence storage is implemented so users only need to sign in once and the account state will be retained between sessions. Figure 35 shows the sign-up, sign-in, and sign-out pages. The user icon will be displayed on the top-left corner of the prediction home page.

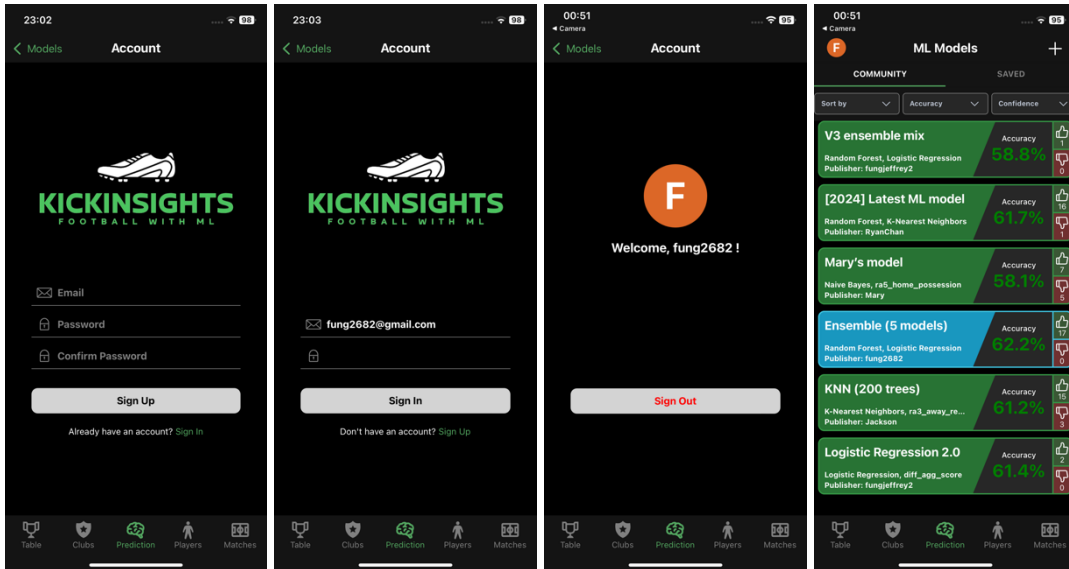


Figure 35. Sign In, Sign Up, Sign Out, and the prediction home page

Figure 36 shows that if a KickInsights user is not signed in, he or she will visit the prediction tab as a guest user, who can still view the community models but will not be able to create a model or save models published in the community.

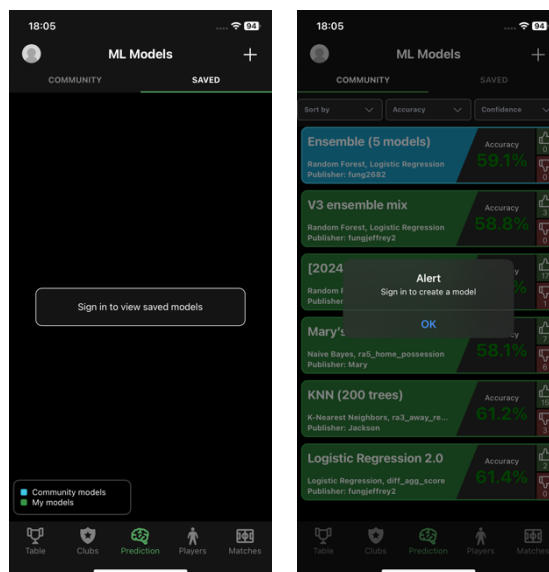
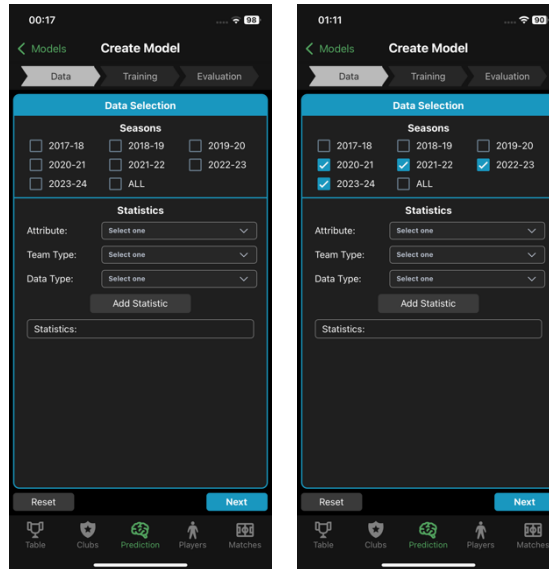


Figure 36. Prediction Tab for a guest user

2.4.2 Creating a model

KickInsights offers an input form for users to conveniently select data and algorithms to train and optimize an ML model. The form consists of three pages, “Data”, “Training”, and “Evaluation”, as shown in Figure 37. Users can navigate through the pages by the “Next” and “Previous” buttons, and reset form inputs to the default version in each page.



*Figure 37. “Data” page of the model creation form.
Left: Default form. Right: Form with seasons selected*

On the “Data” page, users can select one or more premier league seasons as the training data for their model. Some users may prefer using all available seasons while others may want to exclude certain seasons for various factors, one being the COVID-19 pandemic in which matches were played under empty stadiums and thus game outcomes were affected. To include a season, simply tick its checkbox, as shown in Figure 37.

Figure 38 shows the next section which allows users to choose the statistics for training, by selecting from the dropdown pickers an attribute, team type, and data type. For attributes, a wide variety of figures in a football match are provided including the match result, goals scored, possession, and corners (see Appendix D for the full list of attributes). For team type, the own team’s or the opponent team’s data, as well as the differences between both teams, can be selected for training. For data type, the aggregate value or the rolling averages in different timeframes (see Appendix D for the available timeframes) of the selected attribute can be chosen. Altogether, these three input fields offer 450 combinations of statistics for users to accurately pick the ones they desire. On the right side of the list, users can also remove a statistic easily with the remove button.

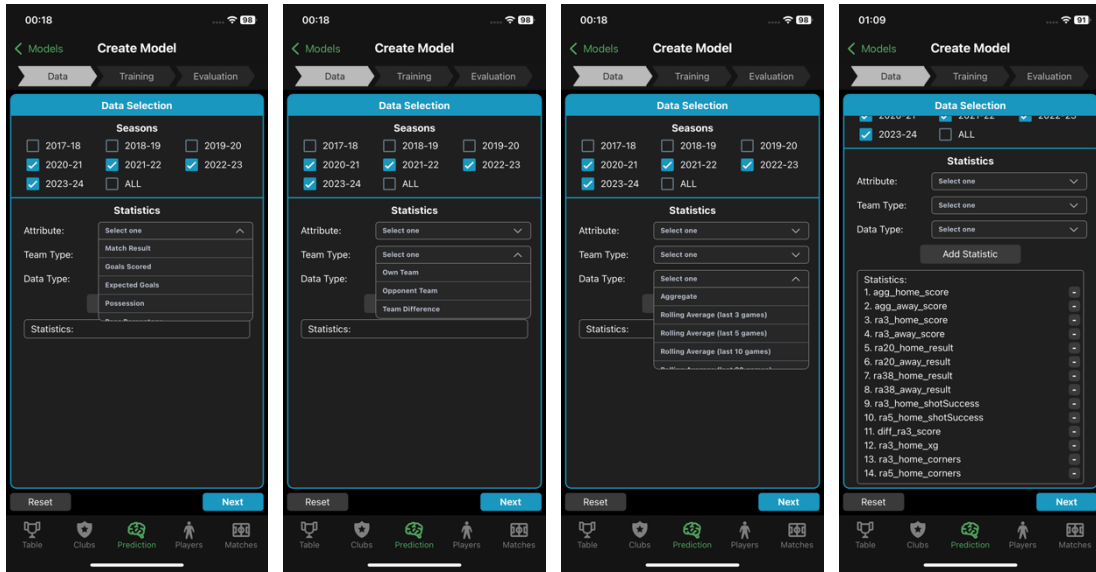


Figure 38. Statistics input fields and list of the “Data” page

If the three dropdown inputs are provided, users can add the statistic to the list and repeat until all desired statistics for training are included. For example, Figure 38 shows a model under creation, with the four most recent seasons (2020-2024) selected and 14 match statistics added to the list.

After data selection, users are then provided with the “Training” page, in which one or more ML algorithms can be selected for the model. As shown in Figure 39, users simply need to tick a checkbox to include the corresponding algorithm. Sliders can also be used to adjust the specific parameters for each model, like the number of neighbors for KNN, and the regularization parameter C for SVM. Additional parameters like the solver and the learning rate are required for Logistic Regression and AdaBoost respectively. If more than one model is selected, an ensemble model combining the algorithms and parameters will be created automatically. Continuing the previous example, a random forest with 130 trees is selected as the algorithm.

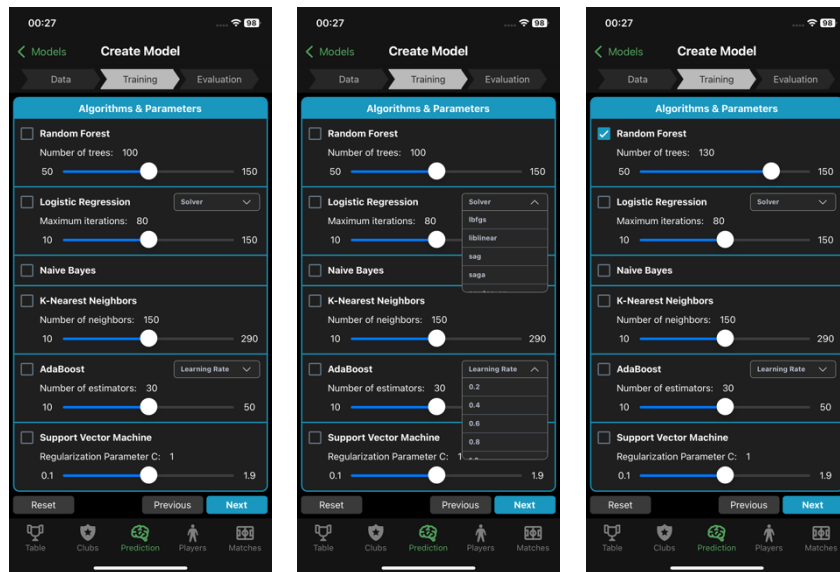


Figure 39. "Training" page of the model creation form

The six algorithms including Random Forest, Logistic Regression, Naive Bayes, KNN, AdaBoost, and SVM, were provided on this page because not only are they some of the most commonly used algorithms, so users are more likely to understand their training logic and parameters, but more importantly, they also provide the confidence of each prediction made. It is crucial for the next part, model evaluation.

The "Evaluation" page consists of four major parts, including the results under various metrics, a confusion matrix, a model performance graph, and a feature importance chart, as shown in Figure 40. Before evaluating the results, users should first select the confidence level required for the model to predict a match's outcome, by choosing a level ranging from 0.3 to 0.95 with the form's top dropdown picker.

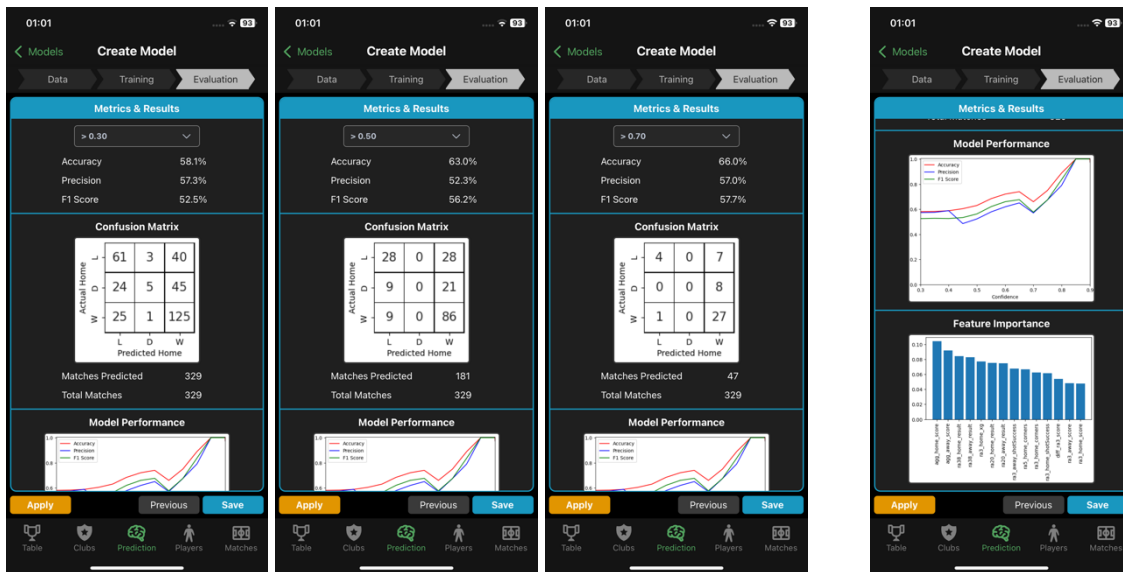


Figure 40. "Evaluation" page of the model creation form.
 Left: Metrics, results, and the confusion matrix.
 Right: Model performance graph and feature importance chart.

For example, with a confidence level of 0.3, the random forest model created earlier predicted all 329 past matches in the current season (2023-2024). It achieved an accuracy of 58.1%, a precision of 57.3%, and a F1 Score of 52.5%. The confusion matrix then presents the actual and the predicted match results for the home team. For instance, the model successfully predicted 125 matches out of 151 matches in which the home team won. Note that only the home team's results are needed as the away team's result can be derived from it.

In general, when the confidence level required is higher, fewer matches will be predicted but the accuracy will increase. For example, Figure 40 also shows that when the confidence level is set as 0.5 and 0.7, the accuracy is raised to 63.0% and 66.0% respectively. From the confusion matrix, it can be observed that a higher portion of matches appeared inside the diagonal column from top-left to bottom-right, indicating that there are more correctly predicted match results.

A performance graph for the model is displayed below the confusion matrix. It captures the aforementioned trends and provides a simple visualization for users to determine how well a model performs under different confidence levels. Generally, the accuracy, precision, and F1 score will be positively sloped, indicating a better performance at higher confidence.

Lastly, a feature importance chart is provided. A feature with a higher importance score indicates that it has a larger effect on the model in predicting the match outcomes [6]. For the previously created random forest model, the four most significant features were the aggregate home score, aggregate away score, and rolling averages of home and away team's match results in the last 38 games. It indicates that the long-term performance of a team had provided strong predicting power for this model. On the other hand, the five features with the least significance were all having the data type of rolling averages of the last 3 games, suggesting that the short-term recent performance of a team had less effect on predicting the future match outcomes under the model.

With this ML model creation interface, users can navigate between data selection, training algorithms, and model evaluation with ease. If the prediction results are not satisfactory, users can modify any input, re-select statistics for training, and fine-tune the model parameters. All previous user inputs will be saved to provide a convenient way for further model optimization.

2.4.3 Applying a model

Users can view the match predictions made by the model created with the “Apply” button in the bottom-left corner of the “Evaluation” page, as shown in Figure 40. Then, an interface identical to the Matches tab will appear, showing the current matchweek as default. Users can visit the past or future matchweeks with the top navigation bar. In each matchweek, the predicted match result for the home team (if any) will be shown on the left, with capitals “W”, “D”, and “L” standing for “Win”, “Draw”, and “Loss” respectively, as shown in Figure 41. For games that are in the past, a green box indicates that the prediction was correct and a red box indicates otherwise, as explained in the legend under the match list, shown in Figure 42.

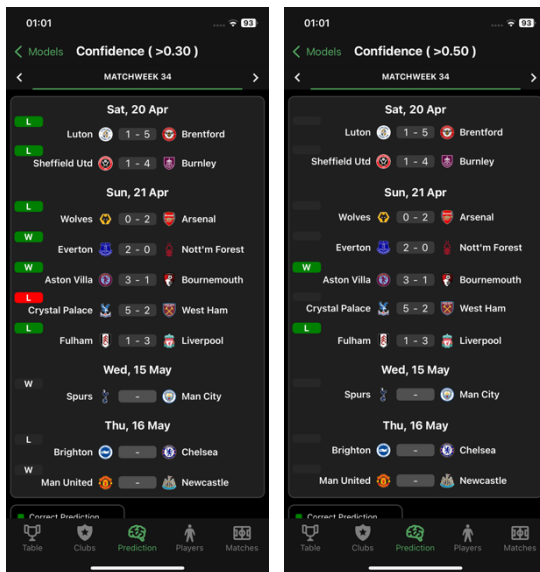


Figure 41. Prediction match list.
Left: Confidence of 0.3. Right: Confidence of 0.5.

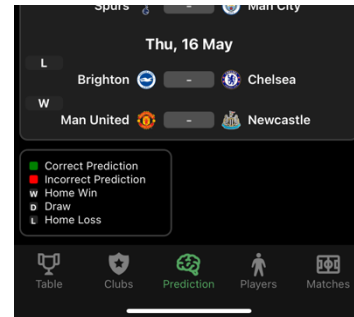


Figure 42. Prediction legend.

Depending on the confidence level selected on the “Evaluation” page, there will be different numbers of matches predicted. For example, as showcased in Figure 40, with a required confidence of 0.3, the model predicted all matches in matchweek 34 (the current matchweek as of 24th April, 2024), with 6 predictions correct, 1 prediction incorrect, and 3 predictions for future matches. If the confidence level is raised to 0.5, only 2 predictions were made.

2.4.4 Saving and Publishing a Model

As shown in Figure 43, users will be prompted to name the model after the model is created. Then, they can either save and publish the model to the “Community Tab” (see Section 2.4.5) or only save the model privately in the “Saved Tab” (see Section 2.4.6). These choices can be changed later. Continuing the previous example, the model was named “My Random Forest” and was published. It then appears on top of the ML models list in the “Community Tab”, as shown in Figure 44 (compare with Figure 35).

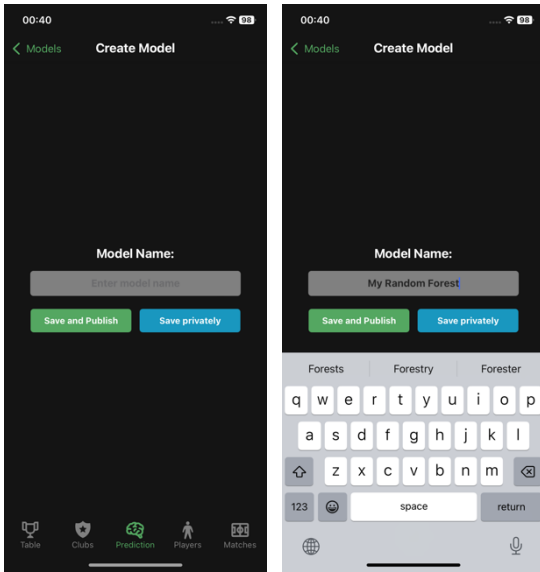


Figure 43. Naming a model

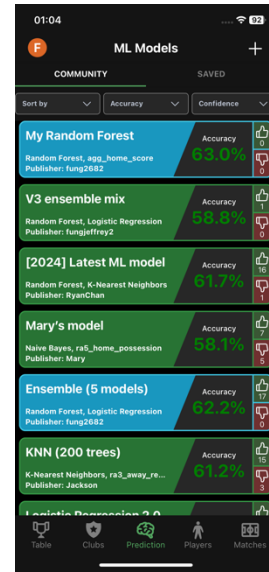


Figure 44: Community Tab

2.4.5 Community Tab

The community tab contains a list of ML models published by different KickInsights users. The ones in blue are uploaded by the user himself or herself, and the ones in green are uploaded by other users. The cover of these models includes the basic information of the model like the model name, algorithms and statistics used for training, and the publisher.

The ML models are sorted by publish date by default. Users can also sort them by their performance or number of likes via the dropdown picker, as shown in Figure 45 (compare with Figure 44).

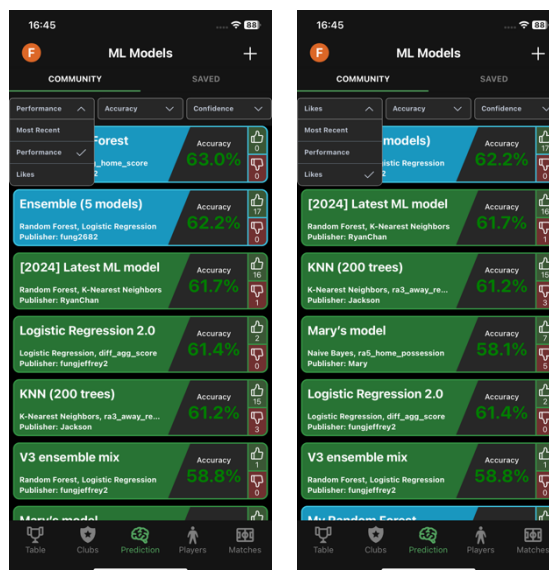


Figure 45. Sorted model list in Community Tab
Left: By model performance. Right: By number of likes

In addition, the displayed metrics and confidence level can also be selected from the dropdown pickers. Users can freely combine the filters to view any of the models' accuracy, precision, and F1 scores under different confidence level requirements, as shown in Figure 46.

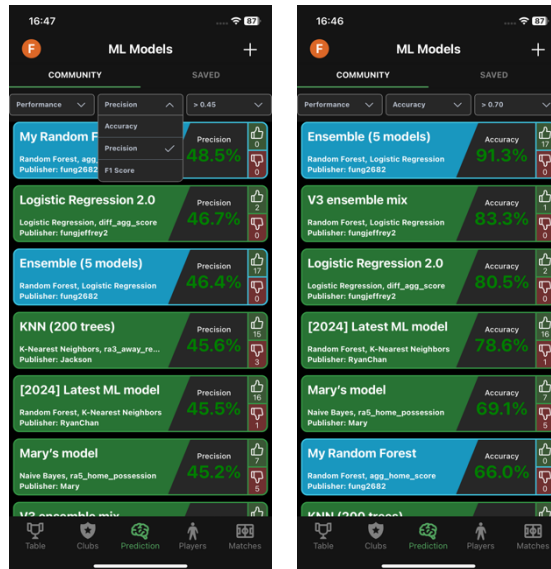


Figure 46. Filtered model list in Community Tab
Left: By precision. Right: By confidence level

Moreover, users can rate a model with the “Like” and “Dislike” buttons on the right of each model, as shown in Figure 47 in which the first two models are liked and the third model is disliked. Figure 48 illustrates that to update the model list, users can simply pull the list downwards to fetch the latest models published.



Figure 47. Rating a model

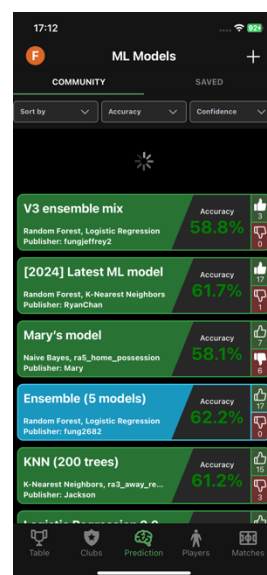


Figure 48. Refreshing the model list

Users can click on any model to view its in-depth setup like the statistics, algorithms, and evaluation results. To enhance UX, the interface is consistent with the model creation form, as shown in Figure 49, except that it is shown in green and no user input is accepted. Users can also apply others' models for match prediction. Lastly, they can save the model inside the "Saved Tab" (see Section 2.4.6).

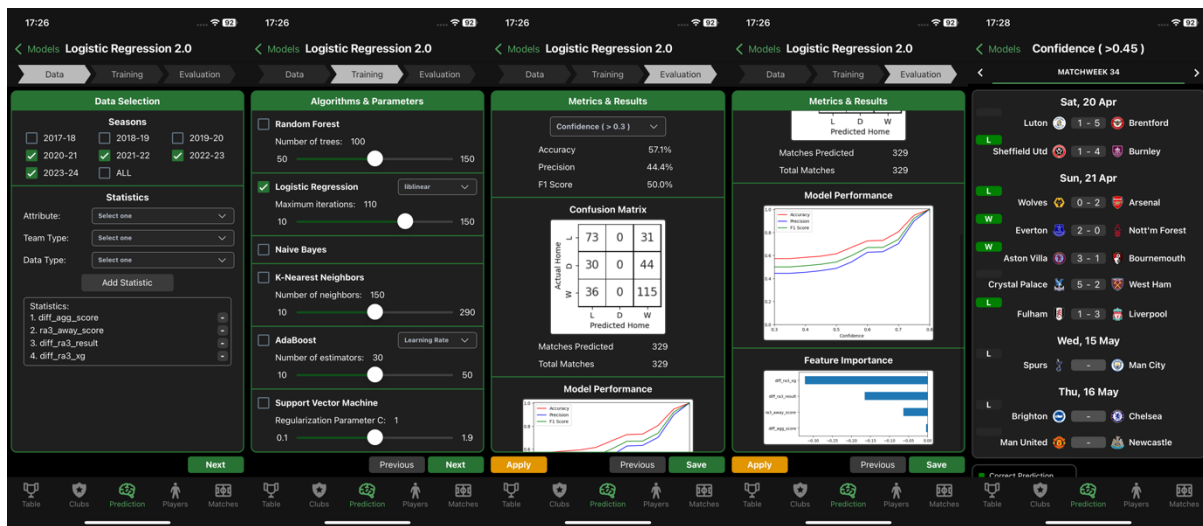


Figure 49. View and apply the published community models

2.4.6 Saved Tab

The saved tab contains models saved by the local user, including those created by him or her, and the ones saved from the community, as shown in Figure 50. For example, the model named "Logistic Regression 2.0" from Figure 49 is saved. Similar to the Community tab, the ML model list can be refreshed by pulling the list downwards. Sorting and filtering functions for these saved models are also provided.

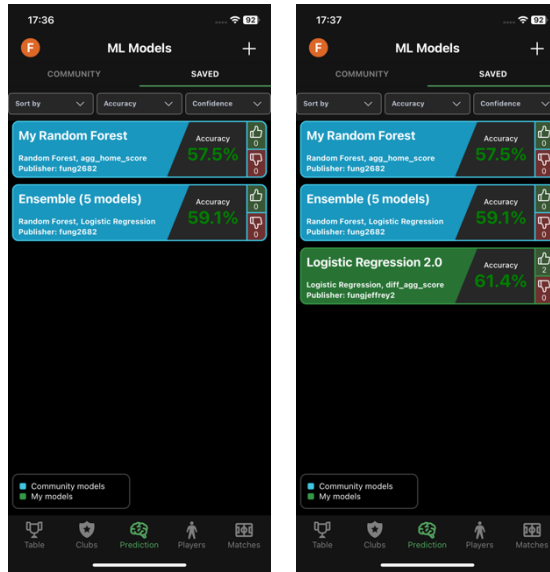


Figure 50. Saved Tab before and after saving a community model

Figure 51 shows that for a model created by the local user, its model inputs can be modified and trained again in the same manner as in the original creation form. Moreover, it can be renamed, published, and unpublished. With the button on the top-right corner, a user-created model can be deleted, and a model saved from the community list can be unsaved. Then, the affected models will be deleted from the server, or removed from the user account's saved list.

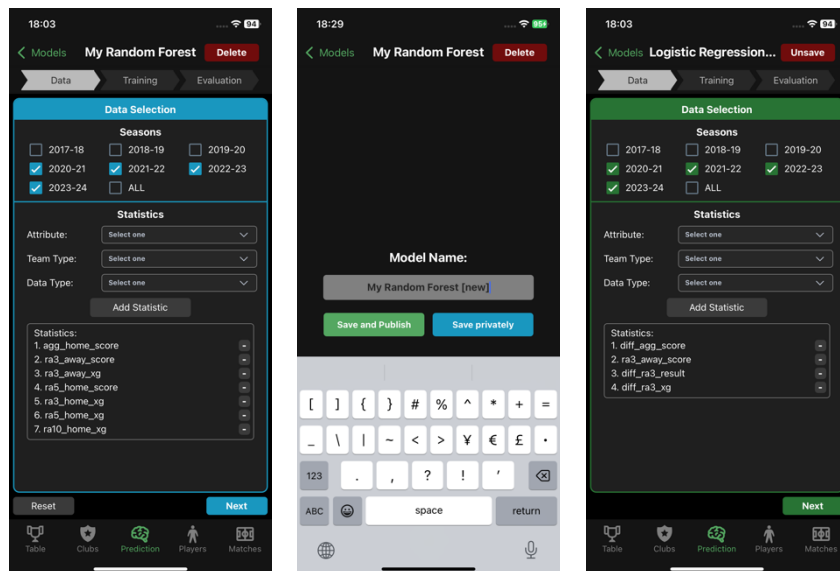


Figure 51. The interface of a model inside the Saved Tab

2.4.7 Interactivity of the prediction system

The ML prediction system allows users to conveniently create, apply, and view others' models. Users with less experience in ML can first explore and distinguish accurate models in the community, observe performance trends and feature importance, and then create their own model by trial and error. Throughout the process, they may learn about the relations between different training attributes and parameters with the prediction results. For example, models using logistic regression will not predict a draw for match results. Weaknesses of each model may also be identified, including the confidence-accuracy tradeoff. Figure 52 shows a variety of model performance graphs generated from different algorithms and configurations with distinct patterns that can be analyzed. On the other hand, experienced ML users may opt for creating and optimizing their own models to achieve higher accuracy and consistency.

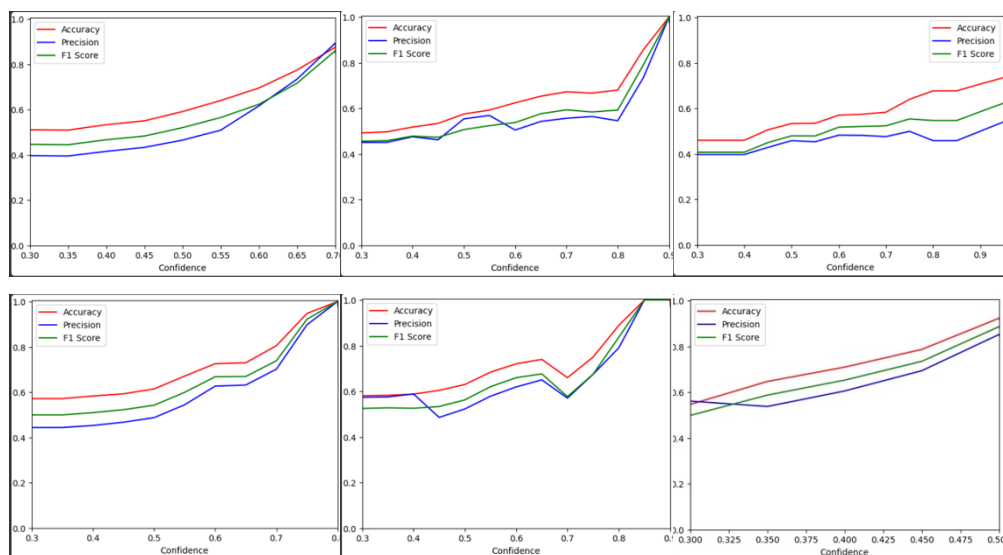


Figure 52. Model performance graphs of different algorithms and parameters

2.4.8 Ability of the prediction system

The ability of the system can be examined in two aspects, training duration and predicting power. Table 6 summarizes the average training duration needed for the six algorithms (with 4GiB memory and 4 vCPUs). It can be shown that most models are trained within 15 seconds, with an average of 12.3 seconds, except for SVM which suffers from timeouts on Google's Cloud Run, the backend engine, since its training cannot be completed within 9 minutes. Overall, the training duration required is considered satisfactory for mobile app usage.

| Number of Seasons | Number of Statistics | Random Forest | Logistic Regression | Naive Bayes | KNN | AdaBoost | SVM |
|-------------------|----------------------|---------------|--------------------------------------|-------------|----------------|--------------------------------------|---------------------------------|
| 1 | 5 | 12.3 | 12.1 | 12.8 | 7.3 | 9.2 | 15.1 |
| 1 | 10 | 12.7 | 12.5 | 14.4 | 8.8 | 9.7 | timeout |
| 5 | 5 | 17.6 | 12.2 | 12.7 | 10.1 | 11.8 | timeout |
| 5 | 10 | 18.3 | 12.5 | 14.4 | 10.3 | 11.9 | timeout |
| Configuration | | Trees: 100 | Max iterations: 110 Solver: lbfgs | | Neighbors: 150 | Estimators: 30 Learning rate: 0.6 | Regularization parameter C: 1.0 |

Table 6. Training duration for Machine Learning models (in seconds)

In terms of the predicting power, since no model can be proved as the best model with the highest accuracy achievable with the system. Let us consider the model “My Random Forest” created in Section 2.4.2, with 4 seasons and 14 statistics for training, as a reasonable representative of the system. It achieved an accuracy of 58.1% and a precision of 57.3%, predicting all 329 past matches in the current season (as of 24th April, 2024), as shown in Figure 40.

Since the distribution of match results is skewed, the home team has a higher chance of winning, meaning that the expected value of 33.3% (a result over win/draw/lose) cannot be used as the benchmark. Instead, being a solution to motivation 3 in Section 1.2, the objective prediction system should be compared with subjective predictions from other football experts and media. Therefore, the premier league predictions made by experts at the National Broadcast Company (NBC), the leading television broadcaster in the United States [7], are selected as the benchmark.

Nicholas Mendola , Joe Prince-Wright, and Andy Edwards from NBC predicted 327 matches in the current season (as of 24th April, 2024). Surprisingly, only 163 of them were correct [8], which is equivalent to an accuracy of 49.8%, 8.3% lower than “My Random Forest”.

To conclude, the ML prediction system successfully fulfilled its purpose, providing a data-driven alternative for match prediction to replace the existing prediction provided in other apps and platforms, which are subjective and prone to bias. At the same time, most of the algorithms available can be trained in under 15 seconds, so no significant extra time costs are incurred for making the predictions.

3. Methodology

This chapter provides a detailed discussion of the implementation of the project. To start with, Section 3.1 covers the frontend development tools that are used for prototype design, UI and UX, and application development. Section 3.2 explores the platforms and methods that are used in backend development in aspects of data collection, wrangling, standard, and the ML prediction system. The backend infrastructure is also discussed. Finally, Section 3.3 reviews the difficulties faced in the development process and explains the solution adopted to mitigate the impact.

3.1 Frontend development

3.1.1 Prototype

Figure 53 shows the prototype of the app created with Figma, a web-based design tool. With the support for interactive flows which allows designers to explore and test user interactions with the interface [9], it improves the efficiency of the design process. The GUI designs were then used as a reference for the initial frontend development. They were modified and optimized at later stages during the frontend development on the native platform (iOS).

The narrated prototype demonstration can be accessed via the following link.

<https://youtu.be/ymzIQAWuqTk>



Figure 53. Prototype of KickInsights created with Figma

3.1.2 UI and UX

To streamline UI and UX within the app, the React Navigation library is used because it allows developers to easily implement navigation systems like stacks, drawers, and tabs in their mobile apps [10]. The code is highly customizable and can be implemented in JavaScript, which is the main programming language for app development.

Within the library, the Bottom Tab Navigators and Top Tab Navigators are used for traversal between the five main screens and their sub-screens respectively, while the Stack Navigator is used to display detailed club and player statistics. These navigators provide an overall

convenient and consistent user interface, with smooth loading screens and transition animations to sustain users' immersiveness in the app.

3.1.3 Application development

React Native is used with Visual Studio Code as the IDE to first develop code in JavaScript, and then convert the code into native languages. It provides high code reusability because only a single codebase is required for both iOS and Android development, allowing future expansion to other platforms. Moreover, development time can be shortened as the open-source framework offers abundant pre-developed components with relevant community support [11].

Expo is used to facilitate the development process. It contains a well-established set of tools and services oriented around React Native for app developers to write and build mobile apps with ease [12]. It also supports hot reloading on emulators and mobile devices, allowing developers to see the results in real-time after changes in the React Native code [11].

3.2 Backend development

The backend system architecture is as follows. Each entity and relation, from data collection, wrangling, to application, will be explained in Section 3.2.1 to Section 3.2.3.

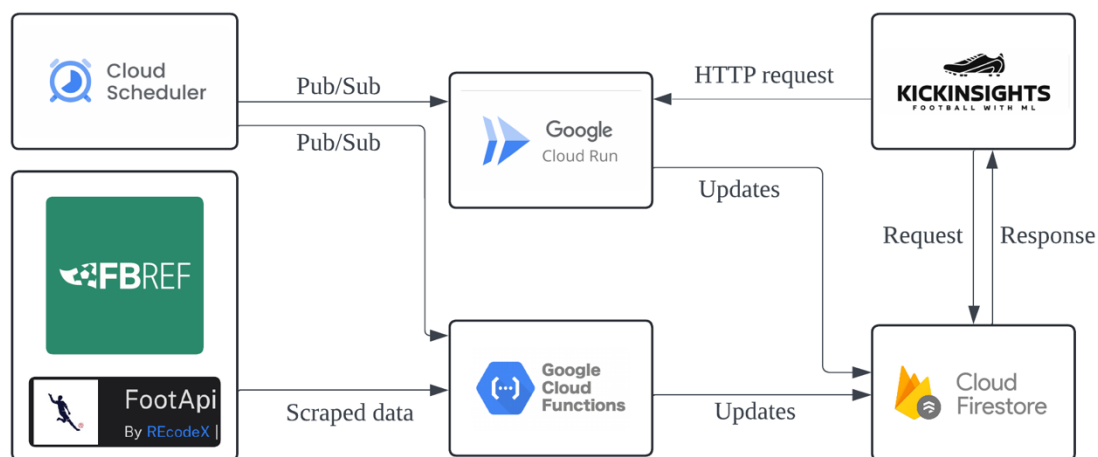


Figure 54. The system architecture of KickInsights

3.2.1 Data Collection

The two main data sources required for the project is firstly, general club and player information used for tables and lists in the main tabs, and secondly, in-depth seasonal and in-game statistics used for the “stats” pages, match centres, and the ML prediction system.

General information is pipelined from RapidAPI, the largest API platform in the world which allows integration of APIs into different applications [13]. It reduces the time and effort needed compared to individual searches for data sources. Additionally, the platform provides code snippets in JavaScript with detailed documentation to facilitate the integration process.

FootApi is selected among various APIs to be the data source for general information since it offers real-time football data like league table standings, fixtures, match detail, etc., with latency within 505 milliseconds and a rate limit of 6 requests per second, which is sufficient for the project scope. Moreover, data is outputted in readily available JSON format.

On the other hand, in-depth statistics are scraped from FBREF, one of the best football data providers constantly collecting team and player statistics, covering the sports comprehensively in a data-driven approach [14]. Its online database is set as the data endpoint for web-scraping, in which Axios, a JavaScript library, is used due to its robust error-handling mechanism.

3.2.2 Data Wrangling and Standard

After obtaining data from FootApi and FBREF, structuring and cleaning are performed on Google Cloud Functions to remove redundant attributes and maintain consistency. For example, players' full names and playing positions are standardized to the FBREF version. Furthermore, Transfermarkt is adopted as the standard for club and player information, as it possesses an extensive database of club and player profiles widely used as the industry benchmark [15]. Additional attributes like the player image URLs, linked to the cloud storage explained in Section 3.2.3, are created in the database for frontend display. Player names with removed diacritics are also used to facilitate the searching function introduced in Section 2.3.4.

A total of 23 scripts for data wrangling are run on Google Cloud Functions, as shown in Figure 55, in which 21 are written in JavaScript, and the remaining two for the ML prediction system are written in Python. Data wrangling functions are triggered by Pub/Sub topics generated from the Google Cloud Scheduler to ensure that the latest data are updated inside the Firestore database. Each function is allocated with 256 MB of virtual memory. The two Python scripts for ML are deployed with Google Cloud Run, as it supports a maximum timeout duration of 60 minutes against the 9 minutes by Cloud Functions [16]. So, the ML wrangling and training functions are provided with a 2nd Gen environment with 6 vCPUs, and allocated with 1 and 4

GiB of memory respectively. This supports high-speed model training after it receives HTTP requests from the KickInsights app for training models.

| Environment | Name ↑ | Last deployed | Region | Recommendation | Trigger | Runtime | Memory allocated |
|-------------|---------|---|-----------------------|----------------|---------|---|----------------------|
| ✓ | 1st gen | fetchClubs_1_daily | 31 Mar 2024, 06:59:02 | asia-east2 | | Topic: firebase-schedule-fetchClubs_1_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchClubs_2_daily | 31 Mar 2024, 06:59:07 | asia-east2 | | Topic: firebase-schedule-fetchClubs_2_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchClubs_3_daily | 31 Mar 2024, 06:59:01 | asia-east2 | | Topic: firebase-schedule-fetchClubs_3_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchClubs_4_daily | 31 Mar 2024, 06:59:01 | asia-east2 | | Topic: firebase-schedule-fetchClubs_4_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchClubStats_1_daily | 31 Mar 2024, 06:59:02 | asia-east2 | | Topic: firebase-schedule-fetchClubStats_1_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchClubStats_2_daily | 31 Mar 2024, 06:59:00 | asia-east2 | | Topic: firebase-schedule-fetchClubStats_2_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchClubStats_3_daily | 31 Mar 2024, 06:59:05 | asia-east2 | | Topic: firebase-schedule-fetchClubStats_3_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchClubStats_4_daily | 31 Mar 2024, 06:59:05 | asia-east2 | | Topic: firebase-schedule-fetchClubStats_4_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchClubStats_5_daily | 31 Mar 2024, 06:59:01 | asia-east2 | | Topic: firebase-schedule-fetchClubStats_5_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchLastNext3_weekdays | 31 Mar 2024, 06:59:02 | asia-east2 | | Topic: firebase-schedule-fetchLastNext3_weekdays-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchLastNext3_weekends | 31 Mar 2024, 06:59:01 | asia-east2 | | Topic: firebase-schedule-fetchLastNext3_weekends-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchMatches_weekdays | 12 Apr 2024, 14:01:06 | asia-east2 | | Topic: firebase-schedule-fetchMatches_weekdays-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchMatches_weekends | 12 Apr 2024, 14:01:39 | asia-east2 | | Topic: firebase-schedule-fetchMatches_weekends-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchPlayerImages_daily | 31 Mar 2024, 06:59:01 | asia-east2 | | Topic: firebase-schedule-fetchPlayerImages_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchPlayers_1_daily | 31 Mar 2024, 06:59:05 | asia-east2 | | Topic: firebase-schedule-fetchPlayers_1_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchPlayers_2_daily | 31 Mar 2024, 07:02:26 | asia-east2 | | Topic: firebase-schedule-fetchPlayers_2_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchPlayers_3_daily | 31 Mar 2024, 06:58:59 | asia-east2 | | Topic: firebase-schedule-fetchPlayers_3_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchPlayers_4_daily | 31 Mar 2024, 06:59:06 | asia-east2 | | Topic: firebase-schedule-fetchPlayers_4_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchPlayers_5_daily | 31 Mar 2024, 06:59:12 | asia-east2 | | Topic: firebase-schedule-fetchPlayers_5_daily-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchTables_weekdays | 31 Mar 2024, 06:59:00 | asia-east2 | | Topic: firebase-schedule-fetchTables_weekdays-asia-east2 | Node.js 18 256 MB |
| ✓ | 1st gen | fetchTables_weekends | 31 Mar 2024, 06:59:00 | asia-east2 | | Topic: firebase-schedule-fetchTables_weekends-asia-east2 | Node.js 18 256 MB |
| ✓ | 2nd gen | ml_data_wrangling | 13 Apr 2024, 12:47:25 | asia-east2 | | Topic: ml_data_wrangling | Python 3.8 1 GiB |
| ✓ | 2nd gen | train_evaluate_model | 17 Apr 2024, 00:52:26 | asia-east2 | | HTTP | Python 3.9 4 GiB |

Figure 55. Cloud functions for data wrangling

As shown in Figure 56, each cloud function has its own schedule for being updated (shown in unix-cron format), depending on the premier league’s match date and time, FBREF and FootApi’s fetch limit, and Firestore’s usage quota, etc. to minimize fetching costs. For example, most matches are played on weekends so the running frequency is higher on Saturdays and Sundays. Also, the schedule adapts to FBREF’s updated time between 6-8 am (UTC +8).

| Name ↑ | Status of last execution | Region | State | Description | Frequency |
|--|--------------------------|------------|---------|-------------|----------------------------------|
| firebase-schedule-fetchClubs_1_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 1 0 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchClubs_2_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 2 0 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchClubs_3_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 3 0 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchClubs_4_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 4 0 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchClubStats_1_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 31 1 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchClubStats_2_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 41 1 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchClubStats_3_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 51 1 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchClubStats_4_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 1 2 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchClubStats_5_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 11 2 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchLastNext3_weekdays-asia-east2 | ✓ Success | asia-east2 | Enabled | | */10 2-6 ** 1-5 (Asia/Hong_Kong) |
| firebase-schedule-fetchLastNext3_weekends-asia-east2 | ✓ Success | asia-east2 | Enabled | | */10 *** 6,7 (Asia/Hong_Kong) |
| firebase-schedule-fetchMatches_weekdays-asia-east2 | ✓ Success | asia-east2 | Enabled | | */15 2-6 ** 1-5 (Asia/Hong_Kong) |
| firebase-schedule-fetchMatches_weekends-asia-east2 | ✓ Success | asia-east2 | Enabled | | */15 *** 6,7 (Asia/Hong_Kong) |
| firebase-schedule-fetchPlayerImages_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 1 1 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchPlayers_1_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 21 2 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchPlayers_2_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 31 2 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchPlayers_3_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 41 2 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchPlayers_4_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 51 2 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchPlayers_5_daily-asia-east2 | ✓ Success | asia-east2 | Enabled | | 1 3 *** (Asia/Hong_Kong) |
| firebase-schedule-fetchTables_weekdays-asia-east2 | ✓ Success | asia-east2 | Enabled | | */5 2-6 ** 1-5 (Asia/Hong_Kong) |
| firebase-schedule-fetchTables_weekends-asia-east2 | ✓ Success | asia-east2 | Enabled | | */5 *** 6,7 (Asia/Hong_Kong) |
| ml_data_wrangling | ✓ Success | asia-east2 | Enabled | | */15 6 *** (Asia/Hong_Kong) |

Figure 56. Cloud scheduling for functions

3.2.3 Machine Learning Prediction System

As shown in the previous section, data wrangling and model training are separated into two functions. This allows a much shorter time required for training the model when users are waiting for the evaluation results. The data wrangling function is triggered by the Cloud Scheduler periodically to update the Firestore database. It first scrapes data of matches from FBREF, filters and transforms the useful data, and then computes the aggregate values and rolling averages of all attributes of each match. This ensures that every time a model is trained, all data are readily available in the database.

When a user submits a training request in the KickInsights app, an HTTP request is sent to trigger the ML training function. The app only needs to collect the user input, like checkboxes ticked and dropdowns selected, and then transfer the request without any actual training data. This minimizes the traffic flow between the app and the backend program. Upon receiving the user inputs, the model training function first selects the required data columns and rows from the Firestore database, then pipeline the matches' data into the involved algorithms, powered by Scikit-learn. It then compares the predicted match outcome against the actual outcome in the testing dataset to calculate the accuracy, precision, and F1 score under different confidence levels. Lastly, the confusion matrices, model performance graph, and feature importance chart are generated and sent back to the KickInsights app together with the predicted results.

To keep users informed about the training progress, each training step will be reflected on the page before the evaluation and prediction results are returned, as shown in Figure 57.

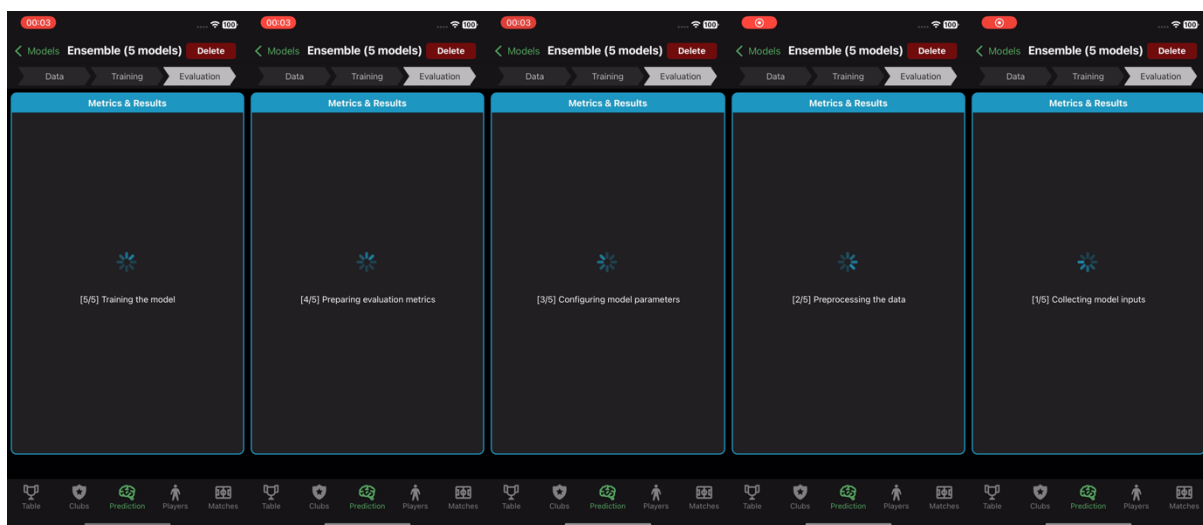


Figure 57. Training status during the training progress

3.2.3 Backend Infrastructure

The application backend and database are constructed with Firebase, Google's BaaS which provides robust infrastructure for data storage. The distributed architecture not only provides load balancing for app usage but also syncs data in real time across data centres, preventing bottlenecks at any specific server.

The wrangled football data is stored in the Google Cloud Firestore Database while player images are stored in the Firebase Storage, as they support offline access of app resources loaded the previous access. They also offer great scalability and data handling capabilities under a cost-effective pricing scheme based on actual usage [17].

Google's Cloud Run API is selected to serve app requests to build, train, and evaluate the ML models because it is fully compatible with the Node.js framework and the Firestore Database, allowing developers to deploy code using a simple command-line interface [18]. For developing the ML components, Python libraries like pandas, numpy, and scikit-learn are used as they cater to diverse models from traditional SVMs and clusterings to advanced artificial neural networks.

When the app is launched, it only needs to interact with the Firestore database by requesting the latest data from it. The following loading screen with loading progress is provided.

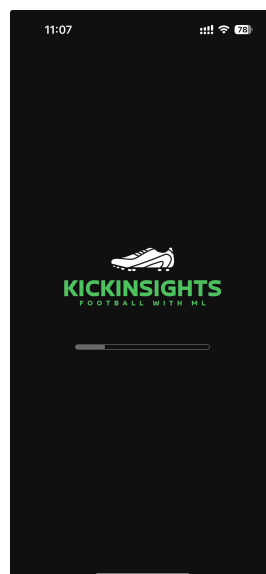


Figure 58. Loading screen of KickInsights

3.3 Difficulties and Mitigations

One of the major challenges faced in the app development process is the compatibility of frontend interfaces with mobile devices. In recent years, devices from various manufacturers have come in a variety of shapes and sizes, making it difficult to apply a single layout onto different physical screens without distortion or other undesired visual effects. On the other hand, it is not time and resource-efficient to customize an interface for each device.

To mitigate the impact, frontend development and testing have been conducted on the iPhone SE (3rd generation), iPhone XR, and iPhone 14 Pro. These devices were selected because of their distinct screen sizes, diagonally measured from 4.7-inch to 6.1-inch [19]. The wide range increases the probability that the app will be compatible with most devices. Moreover, the three iPhones were selected because the project prioritizes launching the app on iOS.

Table 7 shows the variation in UIs among the three devices. The size of the devices is proportional.

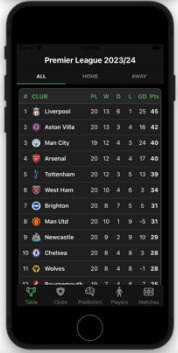

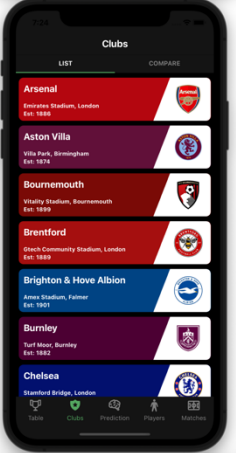
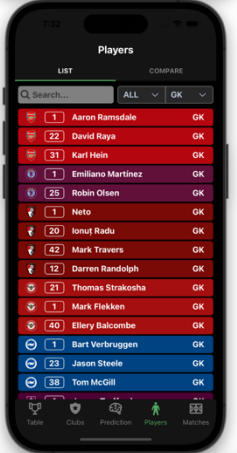
| Device \ Tab | Table | Clubs | Players |
|---|---|--|---|
| <p align="center">iPhone SE (3rd generation)</p> |  |  |  |
| <p align="center">iPhone XR</p> |  |  |  |
| <p align="center">iPhone 14 Pro</p> |  |  |  |

Table 7. User Interfaces of KickInsights on various devices

4. Project Budget and Schedule

This chapter first introduces the budget for the project in Section 4.1. Then, the project schedule will be outlined in Section 4.2.

4.1 Budget

The budget for the project is listed in Table 8. Three services have been purchased including the FootApi Pro Plan, Firebase BaaS, and Google Cloud Platform. For Firebase BaaS, the costs are incurred from User Authentication, Cloud Functions, Cloud Storage, and Cloud Firestore. For Google Cloud Platform, the costs are incurred from Cloud Run and Cloud Scheduler. Billing metrics include the number of upload and download operations, document reads and writes, computing time, etc.

| Service | Provider | Price (HKD) |
|------------------------------------|----------|-----------------|
| FootApi Pro Plan | RapidAPI | \$398.34 |
| Firebase BaaS Blaze Plan* | Google | \$23.80 |
| Google Cloud Platform | Google | \$13.25 |
| Estimated total expenditure | | \$435.39 |

* The service provides a free tier but subsequent usage will be charged.

Table 8. Budget for the project

4.2 Schedule

The schedule of the project is outlined in Table 9. All deliverables are completed on track.

| Time Period | Tasks / Deliverables | Status |
|-------------------------------------|--|-----------|
| September 2023 | <ul style="list-style-type: none"> • Detailed project plan • Project web page • UI / UX design • App prototype | Completed |
| October 2023 | <ul style="list-style-type: none"> • Frontend development: Table tab • Backend and database design • Data collection and wrangling | Completed |
| November 2023 - December 2023 | <ul style="list-style-type: none"> • Frontend development: Clubs, Players tab • Backend and database development • Research on machine learning models | Completed |
| January 2024 | <ul style="list-style-type: none"> • First presentation • Preliminary implementation • Detailed interim report • Frontend development: Machine Learning, Matches tab • Integration of machine learning models | Completed |
| February 2024 | <ul style="list-style-type: none"> • Implementation of Cloud Functions • Implementation of User Account System • Optimization of machine learning models | Completed |
| March 2024 | <ul style="list-style-type: none"> • Application testing • Codebase review • Improvement in UI / UX • Optimization of machine learning models | Completed |
| April 2024 | <ul style="list-style-type: none"> • Final presentation • Final tested implementation • Final report • Project exhibition | Completed |

Table 9. Project Schedule Timetable

5. Future Works

5.1 Comments for Community Models

Currently, the only form of rating on the ML models published on the community is by using likes and dislikes. To allow more useful feedback, a comment section for each model can be developed to encourage constructive idea exchanges within the ML community. For example, users can ask questions on why certain attributes are selected, or suggest alternative parameter settings for the algorithms. This can facilitate exploration and provide more insights.

5.2 Offline Access

At the current stage, the app requires an active internet connection for fetching and updating club, player, match, and ML model data from the Firestore database. Therefore, offline access to limited functions can be provided in the future. For example, users would be able to access club and player statistics during the app's last update, but would not be able to train, publish, or apply the ML models. An offline symbol can be displayed to indicate the connection status.

5.3 Sorting and Filtering Functions

In the same way as how an ML model can be sorted according to upload date, performance, likes, and filtered by confidence level, dropdown pickers can also be implemented for the Club Tab and the Player Tab to allow users to sort and filter the clubs and players by their statistics. For example, clubs can be sorted according to their points scored or games won, and players can be sorted by their goals scored. Filters can be used to display only clubs with more than 20 wins, or players with more than 30 shots. This provides another perspective for users to interact with the football data, and further contribute to its utilization in the app.

5.4 AI chatbot

Large Language Models (LLM), being widely available in the recent past, can be adapted as a great tool for users with less or no experience in ML, by providing definitions and explaining technical terms in simple plain language. For example, metrics like accuracy and precision can be clearly defined, while the training logic and parameters of different algorithms can be explained. Provided with a convenient way to understand the technical terms, the less experienced users can be encouraged to explore other's models or even create their own models.

6. Conclusion

In the modern era of the football world, there is a growing demand for convenient access to comprehensive football data. Fans across the world crave detailed statistics and insights to keep up with their favourite clubs and players. Hence, a variety of mobile apps have appeared on the market as tools for fans to stay connected to the sport. However, they share common downsides like a lack of functions for insights extraction, inconsistent and incomprehensive data provided, and match predictions being subjective and prone to bias.

Therefore, KickInsights has been developed as the solution. It is an app that tackles the inefficiencies of current market alternatives by bringing comprehensive football data onto mobile devices, offering statistical comparison interfaces for users to evaluate the data and extract insights, and finally, implementing a data-driven match prediction system powered by ML algorithms to provide subjective match predictions.

The report first introduces the project's background and motivation to highlight its significance, then covers the objectives, data scope, and designs for the core functionalities of the app, including league tables, club and player statistics, comparison interfaces, match centre, and most importantly, the machine learning prediction system. Some functions are compared with existing apps to highlight improvements in the convenience of accessing statistics.

For the machine learning prediction system, step-by-step demonstrations are provided for creating and applying a model, and viewing models published by other users. Each function of the interface like sorting and filtering is clearly explained. Then, the interactivity of the system is evaluated. The ability of the system in terms of training duration and predicting power are also examined, showing satisfactory results as it has outperformed existing benchmarks.

In addition, methodologies for frontend and backend development are elaborated. Technologies used in the system architecture and backend infrastructure, including Cloud Functions, Firestore, and Cloud Run, are explained and justified. The report then reviews the challenges faced throughout the project and their mitigations. Finally, the project budget and schedule are summarized, followed by a discussion on future works.

7. Project Resources

1. GitHub Repository

- <https://github.com/fung2682/KickInsights>

2. KickInsights Promotional Video

- https://youtu.be/vj2C1eZv_60?si=pONO4pG_EEbLcS9e

3. Project website

- <https://wp2023.cs.hku.hk/fyp23083/>

8. References

- [1] Data Sports Group, “Keeping up with the game: Football data feeds for fans and teams,” Medium, https://medium.com/@marketing_25315/keeping-up-with-the-game-football-data-feeds-for-fans-and-teams-45881110f9db (accessed Nov. 17, 2023).
- [2] Sportico, “Fan demand drives the change in sports media consumption,” Sportico.com, <https://www.sportico.com/business/sports/2022/fan-demand-drives-the-change-in-sports-media-consumption-1234669318/> (accessed Nov. 17, 2023).
- [3] Jsant, “Top 10 football apps for you to have on your phone!,” Real Valladolid Academy, <https://realvalladolidacademy.com/en/the-10-best-football-apps-for-you-to-have-on-your-mobile/> (accessed Nov. 18, 2023).
- [4] L. Chung, “The 7 types of sampling and response bias to avoid in customer surveys,” Delighted, <https://delighted.com/blog/avoid-7-types-sampling-response-survey-bias> (accessed Nov. 18, 2023).
- [5] H. Jain, “Top 10 most watched football leagues in the world,” SportingWiki, <https://www.sportingwiki.com/top-10/top-10-most-watched-football-leagues/> (accessed Nov. 18, 2023).
- [6] T. Shin, “Understanding feature importance in machine learning,” Built In, <https://builtin.com/data-science/feature-importance> (accessed Apr. 24, 2024).
- [7] J. Stoll, “Most watched TV networks in the U.S. 2023,” Statista, <https://www.statista.com/statistics/530119/tv-networks-viewers-usa/#:~:text=In%202023%2C%20NBC%20was%20the,people%20watchig%20the%20TV%20channel.> (accessed Apr. 24, 2024).
- [8] N. Mendola, J. Prince-Wright, and A. Edwards, “Premier League picks: PST’s predictions for Week 35 of 2023-24 season,” NBC Sports, <https://www.nbcsports.com/soccer/premier-league/news/premier-league-picks-psts-predictions-for-week-1-of-2023-24-season> (accessed Apr. 24, 2024).
- [9] F. Flinn, “Figma prototype: What is it and why use it for design?,” Figma Prototype: What is it and why use it for design? | The Design Project, <https://designproject.io/blog/figma-prototype> (accessed Nov. 18, 2023).
- [10] H. Hariyani, “What’s new in react navigation 6,” Medium, <https://medium.com/simform-engineering/whats-new-in-react-navigation-6-8161eefacc8c> (accessed Nov. 20, 2023).
- [11] H. Agarwal, “Advantages and disadvantages of using react native in 2023,” Techexactly, <https://techexactly.com/blogs/advantages-and-disadvantages-of-using-react-native> (accessed Nov. 20, 2023).

- [12] A. Ravichandran, “Building react native apps-expo or not?,” Medium, <https://adhithiravi.medium.com/building-react-native-apps-expo-or-not-d49770d1f5b8> (accessed Nov. 20, 2023).
- [13] K. Haewon, “RapidAPI review: What is rapidapi and how to use it,” Apidog Blog, <https://apidog.com/blog/what-is-rapidapi-and-how-to-use-it/> (accessed Nov. 21, 2023).
- [14] Toby, “Best sites for Free Football Statistics: Top Soccer Stats websites,” Punter2Pro, <https://punter2pro.com/free-football-statistics-soccer-stats/#:~:text=Fbref%20is%20a%20website%20that,match%20breakdowns%20to%20player%20transfers.> (accessed Nov. 20, 2023).
- [15] C. Wheatley, “Secrets behind transfermarkt and how clubs and players use the platform,” Football London, <https://www.football.london/premier-league/secrets-behind-transfermarkt-how-football-21956019> (accessed Nov. 20, 2023).
- [16] “Function timeout | cloud functions documentation | google cloud,” Google, <https://cloud.google.com/functions/docs/configuring/timeout> (accessed Apr. 24, 2024).
- [17] Pathik, “7 reasons to choose google cloud firestore as your database solution,” BlueWhaleApps, <https://bluewhaleapps.com/blog/7-reasons-to-choose-google-cloud-firestore-as-your-database-solution> (accessed Nov. 22, 2023).
- [18] K. Wisniowski, “Cloud run vs app engine vs cloud function (pros and cons),” Cloud Infrastructure Services, <https://cloudinfrastructureservices.co.uk/cloud-run-vs-app-engine-vs-cloud-function/> (accessed Nov. 23, 2023).
- [19] Apple, “Compare iPhone models,” Apple, <https://www.apple.com/ph/iphone/compare/?modelList=iphone-SE-3rd-gen%2Ciphone-14-pro> (accessed Nov. 23, 2023).

7. Appendices

Appendix A

Definition of in-depth team data:

Contains at least one datum from each of the following areas.

General:

Record (W-D-L), Goals For, Goals Against, Clean Sheets

Shooting:

Goals, Shots, Shots on target, Shots on target %, Goals/Shot, Goals/Shot on target, Average Shot Distance, Free kicks, Penalty kicks, Penalty kicks attempted

Passing:

Assists, [Total] Attempted, [Total] Completed, [Total] Completion %, [Short] Attempted, [Short] Completed, [Short] Completion %, [Medium] Attempted, [Medium] Completed, [Medium] Completion %, [Long] Attempted, [Long] Completed, [Long] Completion %

Pass Types:

Live-ball passes, Dead-ball passes, Free Kicks, Through Balls, Switches, Crosses, Throw-Ins, Corner Kicks, Offsides, Passes blocked

Goal and Shot Creation:

Shot Creating Actions: [SCA] Live-Ball passes, [SCA] Dead-Ball passes, [SCA] Take-Ons, [SCA] Shots, [SCA] Fouls, [SCA] Defensive actions

Goal Creating Actions: [GCA] Live-Ball passes, [GCA] Dead-Ball passes, [GCA] Take-Ons, [GCA] Shots, [GCA] Fouls, [GCA] Defensive actions

Defensive Actions:

Tackles, Tackles Won, Tackles in Def 3rd, Tackles in Mid 3rd, Tackles in Att 3rd, Challenges, Challenges Won, Blocks, Shots blocked, Passes blocked, Interceptions, Clearances, Errors

Possession:

Possession, Touches, Touches in Def penalty area, Touches in Def 3rd, Touches in Mid 3rd, Touches in Att 3rd, Touches in Att penalty area, Take-Ons, Take-Ons Success, Take-Ons Tackled, Carries, Total Distance (yards), Progressive Distance (yards), Progressive Carries, Carries into Att 3rd, Carries into Att penalty area, Miscontrols, Dispossessed, Passes Received, Progressive Passes Received

Goalkeeping:

Shot on Target Against, Goals Against, Saves, Save %, Clean Sheets, Long Pass Attempted, Long Pass Completed, Completion %, Passes Attempted, Throws Attempted, Long Pass %, Average Pass Length (yards), Goal Kicks Attempted, Long Goal Kick %, Average GK Length (yards), Crosses Faces, Crosses Stopped, Actions outside penalty area, Average action distance (yards)

Miscellaneous:

Yellow Cards, Red Cards, Second Yellow Cards, Fouls Committed, Fouls Drawn, Penalty Kicks Won, Penalty Kicks Conceded, Own Goals, Ball Recoveries, Aerial Duels Won, Aerial Duels Lost, Aerial Duels Won %

Appendix B

Definition of in-depth on-field player data (Defender, Midfielder, Forward):
Contains at least one datum from each of the following areas.

General:

Nationality, Age, Appearances, Goals, Assists, Tackles

Shooting:

Goals, Shots, Shots on target, Shots on target %, Goals/Shot, Goals/Shot on target, Average Shot Distance, Free kicks, Penalty kicks, Penalty kicks attempted

Passing:

Assists, [Total] Attempted, [Total] Completed, [Total] Completion %, [Short] Attempted, [Short] Completed, [Short] Completion %, [Medium] Attempted, [Medium] Completed, [Medium] Completion %, [Long] Attempted, [Long] Completed, [Long] Completion %

Pass Types:

Live-ball passes, Dead-ball passes, Free Kicks, Through Balls, Switches, Crosses, Throw-Ins, Corner Kicks, Offsides, Passes blocked

Goal and Shot Creation:

Shot Creating Actions: [SCA] Live-Ball passes, [SCA] Dead-Ball passes, [SCA] Take-Ons, [SCA] Shots, [SCA] Fouls, [SCA] Defensive actions
Goal Creating Actions: [GCA] Live-Ball passes, [GCA] Dead-Ball passes, [GCA] Take-Ons, [GCA] Shots, [GCA] Fouls, [GCA] Defensive actions

Defensive Actions:

Tackles, Tackles Won, Tackles in Def 3rd, Tackles in Mid 3rd, Tackles in Att 3rd, Challenges, Challenges Won, Blocks, Shots blocked, Passes blocked, Interceptions, Clearances, Errors

Possession:

Possession, Touches, Touches in Def penalty area, Touches in Def 3rd, Touches in Mid 3rd, Touches in Att 3rd, Touches in Att penalty area, Take-Ons, Take-Ons Success, Take-Ons Tackled, Carries, Total Distance (yards), Progressive Distance (yards), Progressive Carries, Carries into Att 3rd, Carries into Att penalty area, Miscontrols, Dispossessed, Passes Received, Progressive Passes Received

Miscellaneous:

Yellow Cards, Red Cards, Second Yellow Cards, Fouls Committed, Fouls Drawn, Penalty Kicks Won, Penalty Kicks Conceded, Own Goals, Ball Recoveries, Aerial Duels Won, Aerial Duels Lost, Aerial Duels Won %

Appendix C

Definition of in-depth player data (Goalkeeper):

Contains at least one datum from each of the following areas.

General:

Nationality, Age, Appearances, Goals, Assists, Tackles

Goalkeeping:

Shot on Target Against, Goals Against, Saves, Save %, Clean Sheets, Long Pass Attempted, Long Pass Completed, Completion %, Passes Attempted, Throws Attempted, Long Pass %, Average Pass Length (yards), Goal Kicks Attempted, Long Goal Kick %, Average GK Length (yards), Crosses Faces, Crosses Stopped, Actions outside penalty area, Average action distance (yards)

Miscellaneous:

Yellow Cards, Red Cards, Second Yellow Cards, Fouls Committed, Fouls Drawn, Penalty Kicks Won, Penalty Kicks Conceded, Own Goals, Ball Recoveries, Aerial Duels Won, Aerial Duels Lost, Aerial Duels Won %

Appendix D

Full list of statistics for selection in model creation

Attributes:

Match Result,
Goals Scored,
Expected Goals,
Possession,
Pass Percentage,
Pass Success,
Pass Total,
Save Percentage,
Save Success,
Save Total,
Shot Percentage,
Shot Success,
Shot Total,
Aerials Won,
Clearances,
Corners,
Crosses,
Fouls,
Goal Kicks,
Interceptions,
Long Balls,
Offsides,
Tackles,
Throw Ins,
Touches

Team type:

Own Team,
Opponent Team,
Team Difference

Data type:

Aggregate,
Rolling Average (last 3 games),
Rolling Average (last 5 games),
Rolling Average (last 10 games),
Rolling Average (last 20 games),
Rolling Average (last 38 games)