

Premier League 2025/6 Dashboard

A learning project in Dash

August 2025

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Name | Signature | Position | Date |
| Prepared by | Richard Wood |  | Head Experimentation | 06 Aug 2025 |

## Executive Summary

This document outlines the design for a web-based dashboard for the English Premier League 2025/26 season.

The dashboard, built with Dash (Python), will allow users to select home and away teams, view match details, and add or save commentary for specific games.

The target audience includes analysts, football enthusiasts, and journalists who need to view and annotate match data.

This project is a learning exercise in building a functional, responsive dashboard.

## Functional Requirements

Users must be able to select a ‘HomeTeam’ and an ‘AwayTeam’ from two separate dropdown menus, each containing all 20 Premier League teams.

The application will validate that the selected Home and Away teams are not the same.

The dashboard will automatically display the following metadata for the selected match:

* Match Date and Time
* Location
* The previous and next game for both the Home and Away teams.

An editable text area will be available for users to input commentary.

A savefunction will be included to persistently store user commentary in a database.

## Non-Functional Requirements

The application must be responsive and load within 1-2 seconds for typical user interactions.

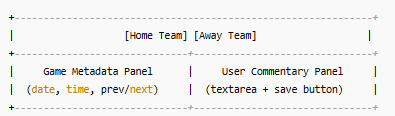
User commentary must persist across different user sessions.

The user interface should be clean and accessible.

## UI Design and Layout

The dashboard layout will be divided into three sections to provide a clear and intuitive user experience:

* **Top Bar:** Contains the dropdown menus for selecting the Home Team and Away Team.
* **Left Panel**: Displays the "Game Info," including match date, time, and the previous and next games for both teams. This data will be sourced from a CSV file.
* **Right Panel**: Features the User Commentary Panel, which includes the editable text area and a "Save" button.



## Technical Design

### Technology Stack

| **Component** | **Technology** |
| --- | --- |
| Dashboard | Dash (Python) |
| Data Processing | Pandas |
| Persistence | SQLite |
| Web Server | Flask (via Dash) |
| Deployment | Local environment |

### Data Sources and Persistence

**Match Data:** Initial match data will be loaded from a CSV file (epl-2025-GMTStandardTime.csv).

This file is assumed to contain 380 rows, with columns for

* Match Number
* Date (including time) e.g. “23/08/2025 15:00:00”
* Location
* Home Team
* Away Team.

This data will be read at startup and stored as a pandas DataFrame in memory for fast retrieval.

**Commentary Data:** User-generated commentary will be stored in a SQLite database for persistence. The database schema for the

game\_commentary table is as follows:

* id: INTEGER, PRIMARY KEY
* home\_team: TEXT
* away\_team: TEXT
* commentary: TEXT
* UNIQUE(home\_team, away\_team)

## Architecture & Data Flow

The system architecture follows a logical flow from the user interface to the data layers. The flow is:

1. A user selects a Home and Away team, which triggers a Dash Callback.
2. The callback retrieves match information for the selected teams from the in-memory pandas DataFrame.
3. The application then queries the SQLite database to load any existing commentary for that specific match.
4. This data is then displayed on the dashboard.
5. When a user adds or edits commentary and clicks "Save," a callback saves the new commentary to the SQLite database. The system will use the unique combination of Home and Away teams to identify the match. A confirmation message is then shown to the user.

## Future Enhancements

The design provides a foundation for future features, including:

* Calendar view of fixtures (month/week).
* Commentary history (versioning).
* User login and personalization.
* REST API for external access.

## Risks and Mitigations

| **Risk** | **Mitigation** |
| --- | --- |
| Changes in the source CSV file structure could break the application. | Implement input validation and preprocessing for the CSV file to ensure it conforms to the expected format. |
| Multiple users attempting to save commentary simultaneously could lead to data collision. | Future versions could be improved by adding a timestamp or a user-specific key to the commentary data. |