Functional Requirements Document (FRD) by Govind Pillai

Cricket Performance and Insights System

1. Document Purpose

This Functional Requirements Document (FRD) defines the functional specifications and system behavior for generating cricket insights using the datasets `deliveries.csv` and `matches.csv`.

The purpose of this system is to derive player performance metrics, stadium analysis, and identify potential coaching candidates based on data trends post-2020.

---

2. System Overview

The proposed analytics system will:

- Process match-level and delivery-level data.
- Compute and visualize player and stadium statistics.
- Identify top performers and potential coaches based on defined conditions.
- Enable decision-making for player selection, strategy formulation, and staff recruitment.

---

3. Data Sources

Dataset	Description
matches.csv	Contains match-level data such as match ID, season, date, venue, teams, toss decision, result, and winner.
deliveries.csv	Contains ball-by-ball information including batsman, bowler, fielder, runs scored, extras, and dismissal details.

4. Functional Scope

The system will compute insights and generate reports based on the following 11 requirements.

5. Functional Requirements

5.1 Player Performance Analysis

ID	Function	Description	Input Data	Output / Result
FR1	**Identify top-scoring batsmen post-2020**	Compute total runs scored by each batsman in matches after 2020. Exclude players who played in initial seasons (e.g., 2008–2012).	`deliveries.csv`, `matches.csv`	List of batsmen ranked by total runs.
FR2	**Identify power hitters with highest strike rate (≥50 balls faced)**	Calculate strike rate = (total runs / balls faced) × 100 for players after 2020. Only include players who have faced ≥50 balls.	`deliveries.csv`	List of players with strike rate and balls faced.
FR3	**Identify top fielders by number of catches**	Count dismissals of type `'caught'` per fielder across all matches.	`deliveries.csv`	List of fielders ranked by catch count.
FR4	**Identify top bowlers by wickets post-2020**	Count dismissals credited to each bowler in matches after 2020. Exclude bowlers who participated in initial seasons.	`deliveries.csv`, `matches.csv`	List of bowlers ranked by wickets.

5.2 Stadium and Match Outcome Analysis

ID	Function	Description	Input Data	Output / Result
FR5	**Stadium-wise analysis: batting first win probability**	Calculate the percentage of matches where the team	`matches.csv`	Table: Stadium vs. Batting First Win Probability.

		batting first won per stadium.		
FR6	**Stadium-wise month-wise batting first win probability**	Extract match month and compute batting first win probability per stadium per month.	`matches.csv`	Table: Stadium × Month × Batting First Win %.
FR7	**Stadium-wise analysis: fielding first win probability**	Calculate percentage of matches where team fielding first won per stadium.	`matches.csv`	Table: Stadium vs. Fielding First Win Probability.
FR8	**Stadium-wise month-wise fielding first win probability**	Extract match month and compute fielding first win probability per stadium per month.	`matches.csv`	Table: Stadium × Month × Fielding First Win %.

5.3 Coaching Candidate Identification

ID	Function	Description	Input Data	Output / Result
FR9	**Identify potential Batting Coach candidates**	Identify players with good run performance in their last active season but who have not played in the last two seasons.	`deliveries.csv`, `matches.csv`	List of potential batting coaches.
FR10	**Identify potential Bowling Coach candidates**	Identify bowlers with strong wicket performance in their last active season but not active in the past two seasons.	`deliveries.csv`, `matches.csv`	List of potential bowling coaches.
FR11	**Identify potential Fielding Coach candidates**	Identify players with high catch counts in their last active season but not active in the past two seasons.	`deliveries.csv`	List of potential fielding coaches.

6. System Workflow

1. **Data Ingestion**

- Load and merge `matches.csv` and `deliveries.csv` using `match_id`.

2. **Data Cleaning & Preprocessing**
    - Handle missing values and duplicates.
    - Standardize player names and stadiums.
    - Extract year and month from match date.
3. **Filtering Logic**
    - For “post-2020” analyses → Include matches with `year > 2020`.
    - For “not in starting seasons” → Exclude players who appeared in seasons ≤2012.
    - For “not played in last 2 seasons” → Identify last active season and check inactivity for 2+ years.
4. **Computation Modules**
    - **Batting module:** Runs, strike rate, boundaries.
    - **Bowling module:** Wickets, economy, strike rate.
    - **Fielding module:** Catches, run-outs.
    - **Match outcomes module:** Batting/fielding first probabilities.
    - **Coaching candidate module:** Player inactivity and prior performance.
5. **Reporting & Visualization**
    - Generate summarized CSVs and dashboards for each analysis.
    - Visualize trends using charts (e.g., bar charts for top players, heatmaps for stadium-month outcomes).

---

7. Non-Functional Requirements

Category	Requirement
Performance	The system must process up to 1M records in under 30 seconds for aggregation queries.
Scalability	Should handle addition of new seasons without schema modification.
Data Integrity	Ensure accurate join between `matches.csv` and `deliveries.csv` on `match_id`.
Usability	Output should be exportable in CSV or Excel formats.
Maintainability	Modular code to allow easy updates for new analytics metrics.

8. Assumptions

- `matches.csv` and `deliveries.csv` contain consistent `match_id` keys.
- Match data includes complete records up to the current season (2025).
- Player names are unique per dataset (if not, standardization is applied).

9. Deliverables

1. Data processing scripts or pipelines (Python/Pandas).
2. Output CSV or dashboards for each requirement (FR1–FR11).
3. Documentation of metrics and calculation logic.
4. Versioned datasets post-cleaning.