VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANASANGAMA,BELAGAVI–590018

A DBMS MINI PROJECT
REPORT ON

"CRICKET DATABASE MANAGEMENT SYSTEM"

Submitted in partial fulfillment of the requirements for the IV Semester of degree of **Bachelor of Engineering in Information Science and Engineering** of Visvesvaraya Technological University,Belagavi

Submitted by

1RN23IS015 Amrutha Hegde
1RN23IS016 Amrutha V
1RN23IS023 Apeksha TR
1RN23IS029 Aruna Sanjana KA
1RN23IS054 Brunda R

Under the Guidance of

Ms.SOWMYASHREE S
Assistant Professor
Dept.of ISE,RNSIT

ESTD . 2001
An Institute with a Difference

Department of

Information Science and Engineering

RNS Institute of Technology

Dr.Vishnuvaradhan Road,Rajarajeshwari Nagar post, Channasandra, Bengaluru-560098

2024-2025

RNS INSTITUTE OF TECHNOLOGY

Dr.Vishnuvaradhan Road,Rajarajeshwari Nagar post, Channasandra,

Bengaluru - 560098

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the Database Management System Mini project work entitled has been successfully completed by **Amrutha Hegde(1RN23IS015), Amrutha V(1RN23IS016), Apeksha TR(1RN23IS023), Aruna Sanjana KA(1RN23IS029), Brunda R(1RN23IS054),** bonafide students of **RNS Institute of Technology, Bengaluru** in partial fulfillment of the requirements for the award of degree in **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during academic year **2024-2025.** The Database Management System Mini project report has been approved as it satisfies the academic requirements in respect of Mini project work for the said degree.

Ms. Sowmyashree S

Faculty In charge

Dr. Suresh L

Professor & HoD -ISE

DECLARATION

We, **Amrutha Hegde(1RN23IS015), Amrutha V(1RN23IS016), Apeksha TR(1RN23IS023), Aruna Sanjana KA(1RN23IS029), Brunda R(1RN23IS054),** students of IV Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Database Mini projectwork entitled has been carried out and submitted in partial fulfillment of the requirements for the IV *Semester degree of* ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi* during academic year 2024-2025.

Place:Bengaluru

Date:

1RN23IS015 Amrutha Hegde
1RN23IS016 Amrutha V
1RN23IS023 Apeksha TR
1RN23IS029 Aruna Sanjana KA
1RN23IS054 Brunda R

ACKNOWLEDGMENT

The fulfillment and rapture that go with the fruitful completion of any assignment would be inadequate without mentioning the people who made it possible, whose steady guidance and support crowned the efforts with success.

We would like to profoundly thank the Management of RNS Institute of Technology for providing such a healthy environment to carry out this project work. We would like to express our thanks to our Director, **Dr. M.K. Venkatesha**, and Principal, **Dr. Ramesh Babu H.S**., for their support and inspiration towards the attainment of knowledge.

We wish to place on record our gratitude **to Dr. Suresh L**., Professor and Head of the Department, Information Science and Engineering, for being the driving force and mastermind behind our project work. Our heartfelt thanks go to **Mrs. Sowmyashree S**, Assistant Professor, Department of Information Science and Engineering, for guiding the project and to all the staff members of the Department of Information Science and Engineering for their help at all times.

Additionally, we extend our gratitude to all other teaching and non-teaching staff of the Information Science & Engineering department, RNSIT, Bangalore, who have directly or indirectly helped us carry out the project work.

Place: Bengaluru
Date: 23/05/2025

1RN23IS015 Amrutha Hegde
1RN23IS016 Amrutha V
1RN23IS023 Apeksha TR
1RN23IS029 Aruna Sanjana KA
1RN23IS054 Brunda R

ABSTRACT

The Cricket Database Management System (CDBMS) is designed to efficiently manage and organize information related to cricket matches, teams, players, and statistics. With the increasing amount of data generated in cricket—from local tournaments to international leagues—this system provides a structured and scalable solution for storing, retrieving, and analyzing match-related data. The project implements a relational database model to store information such as player profiles, team details, match fixtures, results, and performance statistics. Key features include player-wise and team-wise performance tracking, real-time score updates, and historical match data management. The system also supports queries for generating customized reports like top scorers, best bowlers, and win/loss ratios.

Developed using SQL and integrated with a user-friendly front end (PHP, JavaScript, CSS). This project aims to demonstrate the practical application of database management principles in a real-world sports environment and can be extended for use by sports analysts, teams, or cricket organizations.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVATION

API	-	Application Programming Interface
CSS	-	Cascading style sheets
DBMS	-	Database Management System
ER	-	Entity Relationship
HTML	-	Hypertext Markup Language
HTTP	-	Hypertext Transfer Protocol
JS	-	JavaScript
PHP	-	PHP Hypertext Preprocessor
SQL	-	Structured Query Language

Chapter 1

INTRODUCTION

1.1 Background

A database is an organized collection of data, generally stored and accessed electronically from a computer system. Where databases are more complex they are often developed using formal design and modeling techniques.

The database management system (DBMS) is the software that interacts with end users, applications,the database itself to capture and analyze the data and provides facilities to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a "database system". Often the term "database" is also used to loosely refer to any of the DBMS, the data base system or an application associated with the database. The DBMS manages three important things: the data,the database engine that allows data to be accessed, locked and modified and the database schema, which defines the database's logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform administration procedures. Typical database administration tasks supported by the DBMS include change management, performance monitoring/tuning and backup and recovery. Many database management systems are also responsible for automated rollbacks, restarts and recovery as well as the logging and auditing of activity.

1.2 Introduction to Cricket Database Management System

Cricket, one of the most popular sports in the world, generates a vast amount of data every time a match is played—ranging from player statistics and team performance to match results and tournament details. With the rise of domestic leagues like the Indian Premier League (IPL), Big Bash League (BBL), and international tournaments, the need to efficiently manage and analyze this data has become more critical than ever. A Cricket Database Management System (CDBMS) is developed to address this requirement by offering a systematic way to store, access, and manage cricket-related data using database technologies.

The main objective of this project is to design and implement a relational database that can handle complex data structures associated with cricket. The system is built to manage a wide array of data such as player profiles, team information, match details and

individual performance records (runs scored, wickets taken, strike rate, economy rate, etc.).

The Cricket DBMS is designed not only to store data but also to allow efficient retrieval through Structured Query Language (SQL) queries. This enables the generation of meaningful insights like top scorers, most economical bowlers, head-to-head records between teams, and performance trends over time. The system may also include normalization to reduce redundancy and maintain data integrity across multiple tables.

To enhance usability, the system can be integrated with a front-end application built using web or desktop development tools (like PHP, Java, or Python). This interface allows users such as sports analysts, cricket boards, journalists, or even fans to interact with the data more intuitively—whether it's for live updates, match predictions, or historical analysis.

In summary, the Cricket Database Management System is a real-world application of database design principles aimed at organizing and analyzing cricket data efficiently. It serves as a foundation for more advanced systems that could incorporate data visualization, machine learning for performance prediction, or API integrations for real-time score updates. This project not only enhances technical skills in database management but also demonstrates how data-driven systems can add value to the sports industry.

Chapter 2

E R DIAGRAM AND RELATIONAL SCHEMA DIAGRAM

2.1 Description of ER Diagram

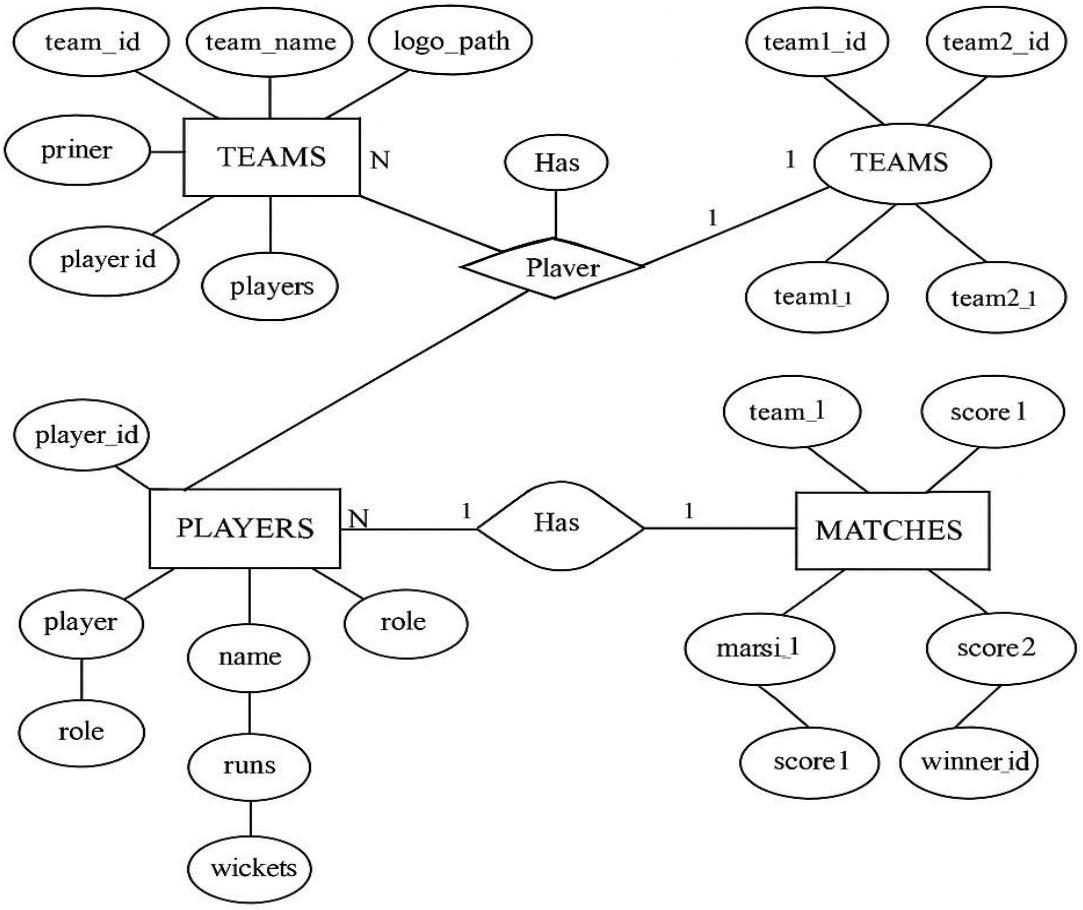


Figure 2.1: E-R Diagram for Cricket Database Management System

- **Entities**:

- **Team**: Represents a cricket team, with attributes such as team_id , team_name , matches_played , wins , losses , nrr , and points .

- **Player**: Represents individual players, with attributes including player_id , team_id (foreign key), player_name , role , runs , and wickets .

- **Match**: Represents matches played, with attributes such as match_id , team1_id , team2_id , match_date , venue , team1_score , team2_score , and winner_id .

• **Relationships**:

- **Team to Player:** A one-to-many relationship where one team can have multiple players. This is represented by the team_id foreign key in the Player entity.

- **Match to Team:** A many-to-many relationship where each match involves two teams. This is represented by the team1_id and team2_id foreign keys in the Match entity.

• **Diagram Representation:** The ER diagram visually represents these entities and relationships, using rectangles for entities, diamonds for relationships, and lines to connect them. This diagram serves as a blueprint for the database structure, guiding the implementation of the relational schema.

• **Normalization:** The ER diagram is designed to follow normalization principles, ensuring that data redundancy is minimized and data integrity is maintained. Each entity is focused on a specific aspect of the cricket database, allowing for efficient data management.

2.2 Description of Relational Schema Diagram

• **Overview**: The Relational Schema Diagram provides a detailed view of how the entities defined in the ER diagram are translated into tables within the database. It outlines the structure of each table, including the fields, data types, and relationships.

• **Tables**:

- Teams Table:
  - Structure:
    - team_id : INT, Primary Key, Auto Increment
    - team_name : VARCHAR(100), Unique
    - matches_played : INT
    - wins : INT
    - losses : INT
    - nrr : FLOAT
    - points : INT

- **Players Table:**
  - Structure:
    - player_id : INT, Primary Key, Auto Increment
    - team_id : INT, Foreign Key referencing Teams(team_id)

- player_name : VARCHAR(100)
- role : ENUM('Batsman', 'Bowler', 'All-rounder', 'Wicket-keeper')
- runs : INT
- wickets : INT

- **Matches Table:**
  - Structure:
    - match_id : INT, Primary Key, Auto Increment
    - team1_id : INT, Foreign Key referencing Teams(team_id)
    - team2_id : INT, Foreign Key referencing Teams(team_id)
    - match_date : DATE
    - venue : VARCHAR(100)
    - team1_score : INT
    - team2_score : INT
    - winner_id : INT, Foreign Key referencing Teams(team_id)

• **Relationships**:
  - The Players table has a foreign key relationship with the Teams table, indicating which team each player belongs to.
  - The Matches table has two foreign key relationships with the Teams table, representing the two teams participating in each match.

• **Diagram Representation:** The Relational Schema Diagram visually represents the tables, their fields, and the relationships between them. It uses rectangles to represent tables, with lines connecting foreign keys to their corresponding primary keys in related tables.

• **Normalization:** The relational schema is designed to adhere to normalization rules, ensuring that each table contains only relevant data and that relationships are properly defined. This design minimizes redundancy and enhances data integrity.

The ER and Relational Schema Diagrams serve as foundational elements in the design of the Cricket Database Management System. They provide a clear structure for the database, ensuring that data is organized efficiently and can be accessed easily.

2.1.1 Components of Cricket Database Management System, E-R Diagram

A **Cricket Database Management System (CDBMS)** is designed to store, manage, and retrieve all critical information related to cricket tournaments, teams, players, and matches. The system uses a structured relational model to handle the complex relationships between entities. Below are the key components of the system:

1. Teams
- Stores data about each team participating in the tournament.
- Attributes include: `team_id`, `team_name`, `nrr` (net run rate), `matches_played`, `wins`, `losses`, and `points`.

2. Players
- Contains personal and performance data of players.
- Attributes include: `player_id`, `player_name`, `role` (batsman, bowler, all-rounder), `runs`, and `wickets`.

3. Matches
- Stores information about scheduled or completed matches.
- Attributes include: `match_id`, `match_date`, `score1`, `score2`, and `winner`.

4. Relationships
- `HAS_PLAYERS`: A one-to-many relationship between **Teams** and **Players** (one team has many players).
- `PARTICIPATES_AS_TEAM1` and `PARTICIPATES_AS_TEAM2`: Represent the participation of teams in matches.
- `WINS`: Indicates which team has won a given match.

2.1.2 E-R Diagram RelationshipsDescription

The **E-R (Entity-Relationship) Diagram** of the Cricket Database Management System illustrates how the core entities—**Teams**, **Players**, and **Matches**—are interconnected.

1. HAS_PLAYERS (1:N)
- **Relationship** between **Teams** and **Players**.
- **Meaning**: One team can have multiple players, but each player belongs to only one team.
- **Type**: One-to-Many (1:N) from `TEAMS` to `PLAYERS`.
- **Purpose**: To assign a set of players to a particular team.

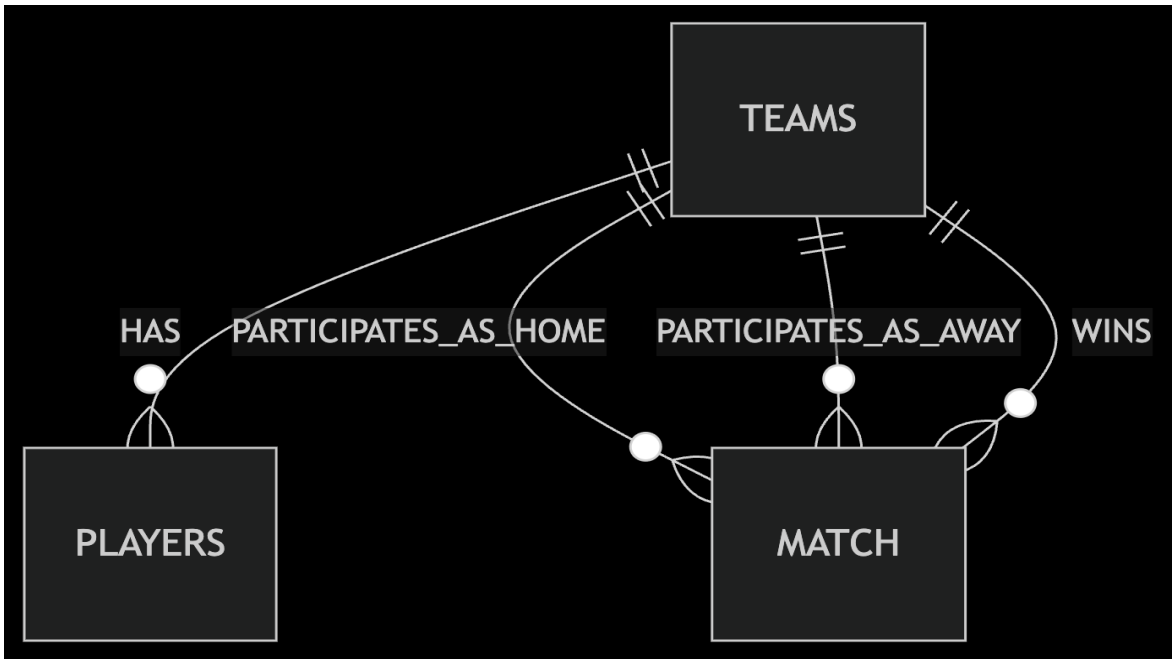
2. PARTICIPATES_AS_TEAM1 (1:N)
- **Relationship** between **Teams** and **Matches**.
- **Meaning**: A team can participate as *Team 1* in multiple matches.
- **Type**: One-to-Many from `TEAMS` to `MATCH`.
- **Purpose**: To track the involvement of a team as the first team in a match.

3. PARTICIPATES_AS_TEAM2 (1:N)
- **Relationship** between **Teams** and **Matches**.
- **Meaning**: A team can participate as *Team 2* in multiple matches.
- **Type**: One-to-Many from `TEAMS` to `MATCH`.
- **Purpose**: To track the involvement of a team as the second team in a match.

4. WINS (1:N)
- **Relationship** between **Teams** and **Matches**.
- **Meaning**: A team can win multiple matches, but each match has only one winner.
- **Type**: One-to-Many from `TEAMS` to `MATCH`.
- **Purpose**: To identify the winning team of each match.

These relationships establish clear connections among the entities, ensuring that the database reflects the real-world structure of a cricket tournament. By defining roles such as player

membership, match participation, and match outcomes, the E-R diagram supports accurate data retrieval, analysis, and reporting.
Let me know if you'd like this styled for a report or presentation.



2.1.3 General Constraints

A well-designed database must ensure **data accuracy, consistency, and integrity**. In the Cricket DBMS, various types of constraints are applied to enforce business rules and relationships among entities like teams, players, and matches.

1. Key Constraints

These constraints ensure that each record in a table is uniquely identifiable and that relationships between tables are accurately maintained.

Constraint Type	Implementation	Purpose
Primary Key	`team_id` (TEAMS), `player_id` (PLAYERS), `match_id` (MATCH)	To uniquely identify each record
Foreign Key	`PLAYERS.team_id → TEAMS.team_id`	To link players to their respective teams
Composite Key	*Not used currently*	N/A

2. Domain Integrity Constraints

These restrict the values allowed in specific fields, ensuring that data entered into the database is valid and meaningful.

Attribute	Constraint	Validation Rule
`TEAMS.nrr`	`DECIMAL(4,3)`	Net run rate must be between -1.999 and +1.999
`PLAYERS.role`	`ENUM`	Must be one of: ‘Batsman’, ‘Bowler’, ‘All-rounder’, ‘Wicket-keeper’
`MATCH.match_date`	Temporal	Cannot be a future date
`MATCH.score1, score2`	`INT UNSIGNED`	Scores must be non-negative

3. Referential Integrity Constraints

These maintain the logical relationships between tables.

Relationship	Constraint	Action on Violation
`TEAMS → PLAYERS`	`ON DELETE CASCADE`	Deleting a team deletes its players
`TEAMS → MATCH (team1, team2)`	`ON DELETE RESTRICT`	Prevent match deletion if team exists
`TEAMS → MATCH (winner)`	`ON DELETE SET NULL`	Sets winner to NULL if the team is deleted

4. Business Rule Constraints

These constraints enforce specific real-world rules of cricket tournaments.

```
-- Teams cannot play against themselves
ALTER TABLE MATCH ADD CONSTRAINT chk_teams_diff CHECK (team1 <> team2);

-- Winner must be either team1 or team2 (or NULL)
ALTER TABLE MATCH ADD CONSTRAINT chk_valid_winner
CHECK (winner IS NULL OR winner IN (team1, team2));

-- Player performance must be non-negative
ALTER TABLE PLAYERS ADD CONSTRAINT chk_runs_wickets
```

CHECK (runs >= 0 AND wickets >= 0);

5. Nullability Constraints

Attribute	Nullable?	Condition
MATCH.winner	Yes	NULL for drawn/abandoned matches
TEAMS.nrr	Yes	Initially NULL until first match
PLAYERS.wickets	No	Must default to 0 if no wickets taken

6. Cardinality Constraints

Relationship	Minimum	Maximum
TEAMS → PLAYERS	11 (Playing XI)	25 (Squad limit)
TEAMS → MATCHES	0	Unlimited
MATCH → TEAMS	Exactly 2	Exactly 2

These constraints ensure the logical soundness, data integrity, and real-world accuracy of the Cricket Database Management System. They also enhance reliability and support error-free data entry and automated rule enforcement in managing cricket tournaments.

2.4.1 Schema Description

The Cricket DBMS is designed to manage data related to cricket teams, players, matches, and their outcomes. It consists of multiple relational tables that are connected via primary and foreign keys to ensure referential integrity.

1. TEAMS Table

Stores information about each team participating in the tournament.

Field Name	Data Type	Constraint	Description
team_id	INT	PRIMARY KEY	Unique identifier for each team
team_name	VARCHAR(50)	NOT NULL, UNIQUE	Official name of the team
nrr	DECIMAL(4,3)	NULLABLE	Net Run Rate (updated after matches)
matches_played	INT	DEFAULT 0	Total matches played
wins	INT	DEFAULT 0	Total matches won
losses	INT	DEFAULT 0	Total matches lost
points	INT	DEFAULT 0	Tournament points based on wins

2. PLAYERS Table

Contains personal and performance data of players.

Field Name	Data Type	Constraint	Description
player_id	INT	PRIMARY KEY	Unique identifier for each player
player_name	VARCHAR(50)	NOT NULL	Full name of the player
role	ENUM	CHECK (role in (Batsman, Bowler, All-rounder, Wicket-keeper))	Player's role
runs	INT	DEFAULT 0	Total runs scored
wickets	INT	DEFAULT 0	Total wickets taken
team_id	INT	FOREIGN KEY REFERENCES TEAMS	Team to which the player belongs

3. MATCH Table

Stores details of each match played.

Field Name	Data Type	Constraint	Description
match_id	INT	PRIMARY KEY	Unique match identifier
match_date	DATE	CHECK (≤ CURRENT_DATE)	Date when the match was played
team1	INT	FOREIGN KEY REFERENCES TEAMS	First participating team
team2	INT	FOREIGN KEY REFERENCES TEAMS	Second participating team
score1	INT UNSIGNED	NOT NULL	Runs scored by team1
score2	INT UNSIGNED	NOT NULL	Runs scored by team2
winner	INT	FOREIGN KEY REFERENCES TEAMS	Winning team (nullable for draws)

4. Constraints and Rules Embedded

- **Player-Team Relationship**: Enforced using foreign key `team_id` in `PLAYERS`.
- **Match Participation**: `team1` and `team2` in `MATCH` reference the `TEAMS` table.
- **Winner Validity**: Must be either `team1` or `team2`, or NULL in case of a draw.
- **Business Logic Checks**:
- Teams cannot play against themselves.
- A player's `runs` and `wickets` must be ≥ 0.
- **Dynamic Constraints** (via triggers):
- Update `points` and `wins` when a match result is recorded.
- Update `nrr` based on score comparison.

Chapter 3

SYSTEM DESIGN

3.1 Table Description

1. TEAMS

Field	Type	Null?	Key	Default
team_id	INT	NO	PRIMARY	AUTO_INCREMENT
team_name	VARCHAR(50)	NO	UNIQUE	–
nrr	DECIMAL(4,3)	YES		NULL
matches_played	INT	NO		0
wins	INT	NO		0
losses	INT	NO		0
points	INT	NO		0

2. PLAYERS

Field	Type	Null?	Key	Default
player_id	INT	NO	PRIMARY	AUTO_INCREMENT
player_name	VARCHAR(50)	NO		–
role	ENUM	NO		'Batsman'
runs	INT	NO		0
wickets	INT	NO		0
team_id	INT	NO	MULTI	–

3. MATCH

Field	Type	Null?	Key	Default
match_id	INT	NO	PRI	AUTO_INCREMENT
match_date	DATE	NO		–
team1	INT	NO	MUL	–
team2	INT	NO	MUL	–
score1	INT UNSIGNED	NO		–
score2	INT UNSIGNED	NO		–
winner	INT	YES	MUL	NULL

4. HAS_PLAYERS (optional normalized mapping)

Field	Type	Null?	Key	Default
team_id	INT	NO	PRI	–
player_id	INT	NO	PRI	–

5. PARTICIPATES_AS_TEAM1

Field	Type	Null?	Key	Default
team_id	INT	NO	PRI	–
match_id	INT	NO	PRI	–

6. PARTICIPATES_AS_TEAM2

Field	Type	Null?	Key	Default
team_id	INT	NO	PRI	–
match_id	INT	NO	PRI	–

7. WINS ***(optional derived relationship)***

Field	Type	Null?	Key	Default
team_id	INT	NO	PRI	–
match_id	INT	NO	PRI	–

3.2 Stored Procedures

A stored procedure is a compiled set of SQL statements that can be reused multiple times. In the context of this Cricket Database Project, stored procedures help automate frequently needed queries, such as retrieving player statistics or filtering matches by date.
We have created two stored procedures in this project:

1. get_matches_between_dates
This procedure returns all matches that occurred between two given dates. It is especially useful when the user wants to filter fixtures to view past or upcoming games based on a specific time frame.

```
DELIMITER $$
DROP PROCEDURE IF EXISTS get_matches_between_dates$$

CREATE PROCEDURE get_matches_between_dates(IN start_date DATE, IN end_date
DATE)
BEGIN
   SELECT * FROM match
   WHERE match_date BETWEEN start_date AND end_date;
END$$
DELIMITER ;
```

2. get_team_players_stats
This procedure returns all players of a specific team, along with their roles, total runs, and total wickets. It helps in quickly viewing a summary of a team's squad and performance.

```
DELIMITER $$
DROP PROCEDURE IF EXISTS get_team_players_stats$$
CREATE PROCEDURE get_team_players_stats(IN teamID INT)
BEGIN
   SELECT player_name, role, runs, wickets
   FROM players
   WHERE team_id = teamID;
END$$
DELIMITER ;
```

These stored procedures increase the modularity and maintainability of your database operations, and are especially useful in larger systems or while integrating a web-based frontend where these procedures can be called via PHP.

3.3 Trigger

Introduction to Trigger

A trigger is a set of SQL statements that is automatically invoked (or fired) by the database when a specified event occurs on a particular table. Triggers can be used to enforce business rules, automatically log historical changes, validate data, or maintain audit trails.

Triggers are categorized into:

BEFORE triggers:

Execute before the triggering SQL statement (INSERT, UPDATE, DELETE).

AFTER triggers:

Execute after the triggering SQL statement.

Purpose of Triggers in the Cricket Database Project

In this project, triggers are implemented to

- Automatically log changes in player statistics (runs and wickets) to maintain historical performance data.
- Provide transparency and traceability of updates made to players.
- Prepare for future data analytics, such as tracking a player's performance progression over time.

Trigger Use Case: Player Performance Logging

Whenever a player's performance data (i.e., runs or wickets) is updated, the trigger automatically inserts the old and new values into a dedicated history table. This ensures that no performance change goes unrecorded, even if edited manually via phpMyAdmin or through backend operations

Trigger Design:

History Table Structure

First, we create a new table player_stats_history to store the logs:

```
CREATE TABLE IF NOT EXISTS player_stats_history (
    id INT AUTO_INCREMENT PRIMARY KEY,
    player_id INT,
    old_runs INT,
    new_runs INT,
    old_wickets INT,
    new_wickets INT,
    changed_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (player_id) REFERENCES players(id)
);
```

This table records:

- The player's ID.
- Their previous and new runs.
- Their previous and new wickets.
- The timestamp of the change.

Trigger SQL Code:

```
DELIMITER $$

DROP TRIGGER IF EXISTS log_player_stats_update$$

CREATE TRIGGER log_player_stats_update

AFTER UPDATE ON players

FOR EACH ROW

BEGIN

   -- Check if runs or wickets have changed

   IF OLD.runs <> NEW.runs OR OLD.wickets <> NEW.wickets THEN

      INSERT INTO player_stats_history (

         player_id,

         old_runs,

         new_runs,

         old_wickets,

         new_wickets

      )
```

```
VALUES (
        OLD.id,
        OLD.runs,
        NEW.runs,
        OLD.wickets,
        NEW.wickets
    );
  END IF;
END$$
DELIMITER ;
```

Benefits of Trigger Usage :

- Automatic Logging: No manual intervention needed for maintaining history.
- Data Integrity: Helps in validating incorrect data entries or rollback scenarios.
- Audit Trail: Useful during presentations or reports to show how player data has evolved.
- Analytics-Ready: History table can be used to visualize player progress or regressions.

Future Possibilities Using Triggers

- Add a trigger to log match score changes in a match_history table.
- Track team name/logo updates for branding audits.
- Automatically update team’s total runs or points when a new match is inserted.

Chapter 4

IMPLEMENTATION

4.1 Front-end Development

The front-end is built using a combination of technologies such as Hypertext Markup Language (HTML), JavaScript and Cascading Style Sheets (CSS). Front-end developers design and construct the user experience elements on the web page or app including buttons, menus, pages, links, graphics and more.

4.1.1 Hypertext Markup Language

HTML is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in what it allows you to create. HTML is the standard markup language for creating Webpages.It stands for Hyper Text Markup Language. It describes the structure of a Web page. It consists of a series of elements. It elements tell the browser how to display the content. It elements are represented by tags. HTML tags label pieces of content such as "heading", "paragraph", "table", and so on. Browsers do not display the HTML tags, but use them to render the content of the page.

4.1.2 Cascading style sheets

CSS stands for Cascading Style Sheets.It is a style sheet language which is used to describe the look and formatting of a document written in markup language.It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces. CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications. Before CSS, tags like font, color, background style, element alignments,border and size had to be repeated on every webpage.This was a very long process. CSS solved that issue. CSS style definitions are saved in external CSS files so it is possible to change the entire website by changing just one file. CSS provides more detailed attributes than plain HTML to define the look and feel of the website.

4.1.3 JavaScript

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities. Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser.It means that a webpage need not be a static HTML, but can include programs that interact with the user, control the browser, and dynamically create HTML content. The JavaScript client-side mechanism provides many advantages over traditional CGI server-side scripts. The JavaScript code is executed when the user submits the form, and only if all the entries are valid, they would be submitted to the Web Server. JavaScript can be used to trap user-initiated events such as button clicks, link navigation,and other actions that the user initiates explicitly or implicitly.
Advantages are: Less server interaction,immediate feedback to the visitors,increased interactivity and richer interfaces.

4.2 Back-end Development

Backend is server side of the website. It stores and arranges data, and also makes sure everything on the client-side of the website works fine.It is the part of the website that you cannot see and interact with. It is the portion of software that does not come in direct contact with the users. The parts and characteristics developed by backend designers are indirectly accessed by users through a front-end application. Activities, like writing APIs, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the backend.

4.2.1 Python for Web Development

Advantages of developing web applications in Python:

- **Easy to learn:** Python is the most popular language for first-time learners for a reason. The language relies on common expressions and whitespace, which allows you to write significantly less code compared to some other languages like Java or C++. Not only that, but it has a lower barrier of entry because it’s comparatively more similar to your everyday language so you can easily understand the code.

- **Rich ecosystem and libraries:** Python offers a vast range of library tools and packages, which allows you to access much pre-written code, streamlining your application

development time. For example, you have access to Numpy and Pandas for mathematical analysis, Pygal for charting, and SLQALchemy for composable queries. Python also offers amazing web frameworks like Django and Flask.

- **Fast prototyping:** Because Python takes significantly less time to build your projects compared to other programming languages, your ideas come to life a lot faster, allowing you to gain feedback and iterate quickly. This quick development time makes Python especially great for startups who can hit the market sooner to gain a competitive edge.
- **Wide-spread popularity:** Python is one of the most popular languages in the world, with communities from all over the world. Because of how popular the language is, Python is continuously updated with new features and libraries, while also providing excellent documentation and community support. Especially for new developers, Python provides extensive support and framework for one to begin their developer journey.

4.2.2 Web Server –XAMPP

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends. It is primarily used for local development and testing of web applications before deployment to a production server.

In our project, XAMPP was used as the local web server environment. It allowed us to:

- Host and run our PHP-based backend services.
- Manage databases using phpMyAdmin.
- Simulate a production environment for testing and debugging.
- Serve HTML, CSS, JavaScript, and PHP files locally.

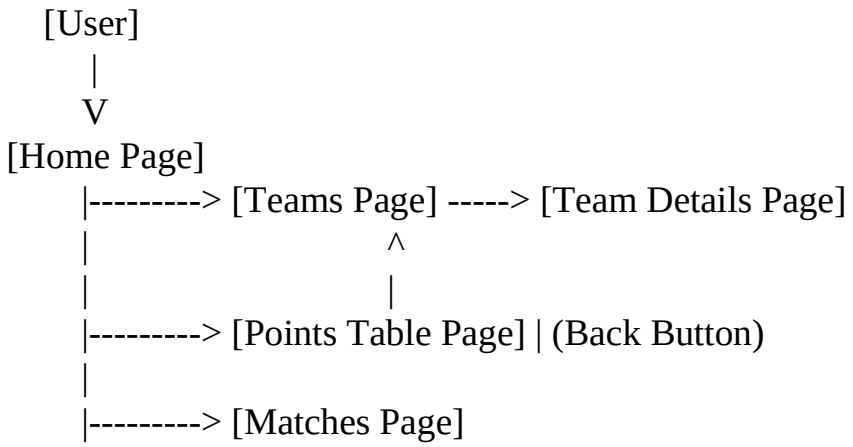
Key Features:

- Apache: Processes HTTP requests and serves web content.
- MySQL/MariaDB: Used to store and manage relational data for the project.
- phpMyAdmin: A web interface for interacting with MySQL/MariaDB databases.
- Control Panel: Offers a user-friendly GUI to start/stop services.XAMPP simplifies the process of deploying a web application during development by bundling the necessary components in a single package, making it easier to set up and manage.

4.2.3 Database –MySQL

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. It is developed, marketed and supported by MySQL AB, which is a Swedish company. It is released under an open-source license. So you have nothing to pay to use it. It is a very powerful program in its own right.It handles a large subset of the functionality of the most expensive and powerful database packages.It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. MySQL supports large databases,upto 50 million rows or more in a table.The default file size limit for a table is 4GB, but you can increase this(if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

4.3 User FlowDiagram



4.4 Discussion of Code Segment

4.4.1 Codes used in implementing the project

db.php

```
<?php
$servername = "localhost";
$username = "root";  // Default MySQL username
$password = "";      // Default MySQL password is empty
$database = "cricket_db";  // Use the name of your database

// Create connection
$conn = new mysqli($servername, $username, $password, $database);

// Check connection
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}
?>
```

Navbar.php

```
<!-- navbar.php -->
<!DOCTYPE html>
<html>
<head>
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <style>
      /* Reset and main styles */
      * {
         margin: 0;
         padding: 0;
         box-sizing: border-box;
```

```
}

/* Header and navigation styles */
.site-header {
    background: linear-gradient(90deg, #0a192f, #172a46);
    color: white;
    padding: 15px 0;
    box-shadow: 0 2px 15px rgba(0,0,0,0.2);
    position: sticky;
    top: 0;
    z-index: 100;
}

.header-container {
    max-width: 1200px;
    margin: 0 auto;
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 0 20px;
}

.logo {
    display: flex;
    align-items: center;
    font-size: 24px;
    font-weight: 700;
}

.logo span {
    margin-left: 8px;
}

.nav-links {
    display: flex;
    list-style: none;
}

.nav-links li {
    margin: 0 5px;
}

.nav-links a {
    color: #ecf0f1;
    text-decoration: none;
    font-size: 16px;
    font-weight: 500;
    padding: 8px 15px;
    border-radius: 6px;
    transition: all 0.3s ease;
    display: flex;
    align-items: center;
}
```

```
.nav-links a:hover {
    background-color: #3498db;
    color: white;
    transform: translateY(-2px);
}

.nav-links a.active {
    background-color: #3498db;
    color: white;
}

.icon {
    margin-right: 8px;
}

/* Mobile menu button */
.menu-btn {
    display: none;
    background: none;
    border: none;
    font-size: 24px;
    color: white;
    cursor: pointer;
}

/* Responsive styles */
@media screen and (max-width: 768px) {
    .menu-btn {
        display: block;
    }

    .nav-links {
        position: absolute;
        top: 70px;
        left: 0;
        width: 100%;
        background: #0a192f;
        flex-direction: column;
        align-items: center;
        padding: 20px 0;
        box-shadow: 0 10px 15px rgba(0,0,0,0.1);
        clip-path: polygon(0 0, 100% 0, 100% 0, 0 0);
        transition: clip-path 0.4s ease;
    }

    .nav-links.show {
        clip-path: polygon(0 0, 100% 0, 100% 100%, 0 100%);
    }

    .nav-links li {
        margin: 15px 0;
        width: 80%;
```

```
        }

        .nav-links a {
            width: 100%;
            text-align: center;
            padding: 12px;
        }
    }
  </style>
</head>
<body>
  <header class="site-header">
    <div class="header-container">
      <div class="logo">
        🏏 <span>Cricket Tracker</span>
      </div>

      <button class="menu-btn">☰</button>

      <ul class="nav-links" id="navLinks">
        <li><a href="teams.php" <?php echo basename($_SERVER['PHP_SELF']) ==
'teams.php' ? 'class="active"' : ''; ?>>
          <span class="icon">🏠</span> Home
        </a></li>
        <li><a href="points_table.php" <?php echo basename($_SERVER['PHP_SELF']) ==
'points_table.php' ? 'class="active"' : ''; ?>>
          <span class="icon">📊</span> Points Table
        </a></li>
        <li><a href="matches.php" <?php echo basename($_SERVER['PHP_SELF']) ==
'matches.php' ? 'class="active"' : ''; ?>>
          <span class="icon">🗓️</span> Matches
        </a></li>
      </ul>
    </div>
  </header>

  <script>
    // Mobile menu functionality
    const menuBtn = document.querySelector('.menu-btn');
    const navLinks = document.querySelector('.nav-links');

    menuBtn.addEventListener('click', () => {
      navLinks.classList.toggle('show');
    });
  </script>
</body>
</html>
```

teams.php

```
<?php
include 'db.php';
include 'navbar.php';
?>
<?php
// Replace the teams-container section in teams.php with this enhanced version

// Define team colors and logos
$team_styles = [
   'Royal Challengers Bengaluru' => ['bg' => '#000000', 'text' => '#ffffff', 'accent' => '#ec2424',
'logo' => 'rcb.png'],
   'Gujarat Titans' => ['bg' => '#031d38', 'text' => '#ffffff', 'accent' => '#E0AA3E', 'logo' =>
'gt.png'],
   'Mumbai Indians' => ['bg' => '#004C97', 'text' => '#ffffff', 'accent' => '#FF9933', 'logo' =>
'mi.png'],
   'Chennai Super Kings' => ['bg' => '#F7B500', 'text' => '#000000', 'accent' => '#005A9C', 'logo'
=> 'csk.png'],
   'Kolkata Knight Riders' => ['bg' => '#4B0082', 'text' => '#ffffff', 'accent' => '#B8860B', 'logo'
=> 'kkr.png'],
   'Rajasthan Royals' => ['bg' => '#254AA5', 'text' => '#ffffff', 'accent' => '#F2C75C', 'logo' =>
'rr.png'],
   'Sunrisers Hyderabad' => ['bg' => '#FF822A', 'text' => '#000000', 'accent' => '#000000', 'logo'
=> 'srh.png'],
   'Lucknow Super Giants' => ['bg' => '#004C8C', 'text' => '#ffffff', 'accent' => '#2F971F', 'logo'
=> 'lsg.png'],
   'Delhi Capitals' => ['bg' => '#17449B', 'text' => '#ffffff', 'accent' => '#EF4136', 'logo' =>
'dc.png'],
   'Punjab Kings' => ['bg' => '#D71920', 'text' => '#ffffff', 'accent' => '#C0C0C0', 'logo' =>
'pbks.png'],
];

// Get team statistics
$result = $conn->query("SELECT t.*,
                  (SELECT COUNT(*) FROM matches WHERE (team1_id = t.team_id OR
team2_id = t.team_id) AND match_date >= CURDATE()) as upcoming_matches
                  FROM teams t
                  ORDER BY t.points DESC");
?>

<style>
   .teams-container {
      display: grid;
      grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));
      gap: 30px;
      padding: 20px;
      max-width: 1200px;
      margin: 0 auto;
   }

   .team-card {
      background-color: white;
```

```
    border-radius: 12px;
    overflow: hidden;

    box-shadow: 0 8px 20px rgba(0,0,0,0.12);
    transition: transform 0.3s ease, box-shadow 0.3s ease;
    position: relative;
    display: flex;
    flex-direction: column;
}

.team-card:hover {
    transform: translateY(-8px);
    box-shadow: 0 12px 28px rgba(0,0,0,0.18);
}

.team-header {
    padding: 20px;
    text-align: center;
    color: white;
    position: relative;
    height: 100px;
    display: flex;
    align-items: center;
    justify-content: center;
}

.team-logo {
    width: 65px;
    height: 65px;
    object-fit: contain;
    background: white;
    border-radius: 50%;
    border: 3px solid;
    padding: 5px;
    position: absolute;
    bottom: -30px;
    left: 50%;
    transform: translateX(-50%);
}

.team-content {
    padding: 40px 20px 20px;
    text-align: center;
}

.team-name {
    font-size: 18px;
    font-weight: 700;
    margin: 0 0 15px;
}

.team-stats {
    display: flex;
```

```
    justify-content: space-around;
    margin-bottom: 20px;
}

.stat-item {
    text-align: center;
}

.stat-value {
    font-size: 22px;
    font-weight: 700;
    line-height: 1;
}

.stat-label {
    font-size: 12px;
    color: #666;
    margin-top: 5px;
}

.team-actions {
    display: flex;
    justify-content: center;
}

.view-btn {
    background-color: transparent;
    color: inherit;
    border: 2px solid;
    padding: 8px 16px;
    border-radius: 6px;
    text-decoration: none;
    font-weight: 600;
    font-size: 14px;
    transition: all 0.2s ease;
}

.view-btn:hover {
    background-color: rgba(0,0,0,0.05);
}

.team-badge {
    position: absolute;
    top: 10px;
    right: 10px;
    background-color: rgba(255,255,255,0.9);
    color: #333;
    font-size: 11px;
    font-weight: 700;
    padding: 3px 8px;
    border-radius: 12px;
}
```

```
 /* Responsive adjustments */
    @media screen and (max-width: 600px) {
        .teams-container {
            grid-template-columns: 1fr;
            padding: 15px;
            gap: 20px;
        }
    }
</style>


<div class="'teams-container">
<?php
while ($team = $result->fetch_assoc()) {
    $team_name = $team['team_name'];
    $style = $team_styles[$team_name] ?? ['bg' => '#3498db', 'text' => '#ffffff', 'accent' =>
'#2c3e50', 'logo' => 'default.png'];

    // Determine if team is in top 4
    $is_top4 = $team['points'] >= 16; // Adjust this threshold based on your tournament

    echo "
    <div class='team-card'>
        <div class='team-header' style='background-color: {$style['bg']}; color: {$style['text']};'>
            <h3>{$team_name}</h3>
            " . ($is_top4 ? "<div class='team-badge'>PLAYOFF CONTENDER</div>" : "") . "
            <img src='{$style['logo']}' alt='{$team_name} logo' class='team-logo' style='border-
color: {$style['accent']};'>
        </div>

        <div class='team-content'>
            <div class='team-stats'>
                <div class='stat-item'>
                    <div class='stat-value'>{$team['matches_played']}</div>
                    <div class='stat-label'>MATCHES</div>
                </div>
                <div class='stat-item'>
                    <div class='stat-value'>{$team['wins']}</div>
                    <div class='stat-label'>WINS</div>
                </div>
                <div class='stat-item'>
                    <div class='stat-value'>{$team['points']}</div>
                    <div class='stat-label'>POINTS</div>
                </div>
            </div>

            <div class='team-actions'>
                <a href='team.php?team_id={$team['team_id']}' class='view-btn' style='color:
{$style['bg']}; border-color: {$style['bg']};'>
                    View Team
                </a>
            </div>
```

```
        </div>
    </div>
    ";
}
?>
</div>
<!DOCTYPE html>
<html>
<head>
    <title>Teams</title>
    <style>
        body {
            font-family: 'Segoe UI', sans-serif;
            background-color: #f0f4f8;
            margin: 0;
            padding: 30px;
        }

        h1#allteam {
            text-align: center;
            color: #2c3e50;
            margin-bottom: 30px;
            border-bottom: 2px solid rgb(17, 46, 65);
            display: inline-block;
            padding-bottom: 5px;
        }

        .teams-container {
            display: flex;
            flex-wrap: wrap;
            justify-content: center;
            gap: 25px;
        }

        .team-card {
            background-color: #ffffff;
            border: 2px solid rgb(21, 54, 77);
            border-radius: 12px;
            width: 250px;
            padding: 20px;
            text-align: center;
            box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
            transition: transform 0.3s ease, box-shadow 0.3s ease;
        }

        .team-card:hover {
            transform: scale(1.05);
            box-shadow: 0 6px 16px rgba(0, 0, 0, 0.15);
        }

        .team-card h3 {
            margin-bottom: 15px;
            color: #34495e;
```

```
    }

    .team-card a {
        display: inline-block;
        margin-top: 10px;
        padding: 10px 18px;
        background-color: #3498db;
        color: #ffffff;
        text-decoration: none;
        font-weight: bold;
        border-radius: 6px;
        transition: background-color 0.2s ease;
        border: 2px solid rgb(21, 54, 77);
    }

    .team-card a:hover {
        background-color: #2980b9;
    }
  </style> </head>
```

Team.php

```
<?php
include("db.php");
$team_id = (int)($_GET['team_id'] ?? 0);

$teamResult = $conn->query("SELECT team_name, matches_played, wins, nrr FROM teams
WHERE team_id = $team_id");
$team = $teamResult->fetch_assoc();
// 1. Define default players for all teams (example, expand as needed)

// 2. Check if players exist for this team
$teamName = trim($team['team_name']);  // Move this up here
$teamNameLower = strtolower($teamName);
// 2. Check if players exist for this team
$playersCheck = $conn->query("SELECT COUNT(*) as count FROM players WHERE
team_id = $team_id");
$countRow = $playersCheck->fetch_assoc();

if ((int)$countRow['count'] === 0 && isset($team_players[$teamName])) {
   // 3. Insert default players for the team
  foreach ($team_players[$teamName] as $player) {
   $pname = $conn->real_escape_string($player['player_name']);
   $prole = $conn->real_escape_string($player['role']);

   // Use default to 0 if not set
   $runs = isset($player['runs']) ? (int)$player['runs'] : 0;
   $wickets = isset($player['wickets']) ? (int)$player['wickets'] : 0;


   $insertQuery = "INSERT INTO players (team_id, player_name, role, runs, wickets)
                VALUES ($team_id, '$pname', '$prole', $runs, $wickets)";
   if (!$conn->query($insertQuery)) {
      echo "Insert Error: " . $conn->error . "<br>";
```

```
    }
}

}
  $playersCheck = $conn->query("SELECT COUNT(*) as count FROM players WHERE
team_id = $team_id");
   $countRow = $playersCheck->fetch_assoc();

// 4. Now fetch players (your existing code)
$playersResult = $conn->query("SELECT player_name, role, runs, wickets FROM players
WHERE team_id = $team_id");

if (!$team) {
    echo "Team not found.";
    exit;
}
$teamName = trim($team['team_name']);

if (isset($teamColors[$teamName])) {
    $primaryColor = $teamColors[$teamName]['primary'];
    $secondaryColor = $teamColors[$teamName]['secondary'];
    $tertiaryColor=$teamColors[$teamName]['tertiary'];
}
else {
    $primaryColor = '#3498db';   // Default primary color (blue)
    $secondaryColor = '#D6EAF8'; // Default secondary color (light blue)
}

// Darken function for hover states
function darken_color($hex, $percent = 20) {
    $hex = str_replace('#', '', $hex);
    $r = max(0, min(255, hexdec(substr($hex, 0, 2)) - round(255 * $percent / 100)));
    $g = max(0, min(255, hexdec(substr($hex, 2, 2)) - round(255 * $percent / 100)));
    $b = max(0, min(255, hexdec(substr($hex, 4, 2)) - round(255 * $percent / 100)));
    return sprintf("#%02x%02x%02x", $r, $g, $b);
}

$playersResult = $conn->query("SELECT player_name, role, runs, wickets FROM players
WHERE team_id = $team_id");
?>

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title><?php echo htmlspecialchars($team['team_name']); ?> - Team Details</title>
  <style>
    body {
      font-family: 'Segoe UI', sans-serif;
      background-color: <?php echo $secondaryColor; ?>;
      margin: 0;

 padding: 20px;
```

```
}

.team-details-container {
  max-width: 960px;
  margin: auto;
  background: #fff;
  padding: 30px 40px;
  border-radius: 12px;
  box-shadow: 0 8px 20px rgba(0,0,0,0.1);
  border: 3px solid <?php echo $tertiaryColor; ?>;
}

h1, h2 {
  text-align: center;
  color: <?php echo $primaryColor; ?>;
}

.team-stats {
  display: flex;
  justify-content: center;
  flex-wrap: wrap;
  gap: 30px;
  margin: 30px 0;
}

.team-stats p {
  background: <?php echo $secondaryColor; ?>;
  padding: 20px 30px;
  font-size: 18px;
  font-weight: 600;
  border-radius: 10px;
  box-shadow: inset 0 0 6px rgba(0,0,0,0.05);
  text-align: center;
  min-width: 150px;
  border: 2px solid <?php echo $tertiaryColor; ?>;
  color: <?php echo $primaryColor; ?>;
}

.back-button, .edit-button {
  display: inline-block;
  background-color: <?php echo $primaryColor; ?>;
  border: 2px solid <?php echo $tertiaryColor; ?>; /* Adding tertiary color */
  color: white;
  padding: 10px 20px;
  border-radius: 6px;
  text-decoration: none;
  font-weight: 600;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  transition: background-color 0.3s ease;
  margin: 20px 10px;
}
```

```
    .back-button:hover, .edit-button:hover {
      background-color: <?php echo darken_color($primaryColor); ?>;
    }

    .btn-container {
      text-align: center;
    }

    .players-table {
      width: 100%;
      border-collapse: collapse;
      margin-top: 25px;
      background-color: #fff;
      border-radius: 10px;
      overflow: hidden;
      box-shadow: 0 4px 8px rgba(0,0,0,0.08);
    }

    .players-table th, .players-table td {
      padding: 15px 12px;
      text-align: center;
      border-bottom: 1px solid #ddd;
    }

    .players-table th {
      background-color: <?php echo $primaryColor; ?>;
      color: white;
      font-weight: 600;
      text-transform: uppercase;
    }

    .players-table tr:hover {
      background-color: <?php echo $secondaryColor; ?>;
    }

    @media (max-width: 600px) {
      .team-stats {
        flex-direction: column;
        align-items: center;



  }
      .team-stats p {
        width: 80%;
      }
    }
  </style>
</head>
<body>


<div class="btn-container">
```

```
  <a href="teams.php" class="back-button">← Back</a>
  <a href="edit_team.php?team_id=<?php echo $team_id; ?>" class="edit-button">✏ Edit Team
& Players</a>

</div>

<div class="team-details-container">
  <h1><?php echo htmlspecialchars($team['team_name']); ?></h1>

  <div class="team-stats">
    <p>Matches Played<br><?php echo $team['matches_played']; ?></p>
    <p>Wins<br><?php echo $team['wins']; ?></p>
    <p>Net Run Rate<br><?php echo $team['nrr']; ?></p>
  </div>

  <h2>Players</h2>

  <table class="players-table">
    <thead>
      <tr>
        <th>Player Name</th>
        <th>Role</th>
        <th>Runs</th>
        <th>Wickets</th>
      </tr>
    </thead>
    <tbody>
    <?php
    while($row = $playersResult->fetch_assoc()): ?>
<tr>
  <td><?php echo htmlspecialchars($row['player_name']); ?></td>
  <td><?php echo htmlspecialchars($row['role']); ?></td>
  <td><?php echo htmlspecialchars($row['runs']); ?></td>
  <td><?php echo htmlspecialchars($row['wickets']); ?></td>
</tr>
<?php endwhile; ?>

    </tbody>
  </table>
</div>

</body>
</html>
```

points_table.php

```
<?php
include 'db.php';
?>
<?php include 'navbar.php'; ?>

<!DOCTYPE html>
```

```
<html>
<head>
   <title>Points Table</title>
   <link rel="stylesheet" href="style.css">
   <style>

body {
   font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
   background-color: #ffffff; /* pure white for clarity */
   margin: 0;
   padding: 40px 20px;
   color: #333; /* dark gray text for readability */
}

.points-table-container {
   max-width: 900px;
   margin: auto;
   background-color: #fafafa; /* very light gray */
   padding: 30px 40px;
   border-radius: 10px;
   box-shadow: 0 4px 12px rgba(0,0,0,0.08);
}

h2 {
   text-align: center;
   color: #222;
   margin-bottom: 30px;
   font-weight: 700;
   font-size: 28px;
   text:bold;
    border-bottom: 2px solid rgb(17, 46, 65);
}

.points-table {
   width: 100%;
   border-collapse: separate;
   border-spacing: 0 8px; /* space between rows */
}

.points-table th, .points-table td {
   padding: 14px 18px;
   text-align: center;
   font-size: 16px;
   font-weight: 700; /* bold */
}

.points-table thead th {
   background-color:rgb(35, 60, 86); /* bright blue */
   color: white;
   font-weight: 600;
   border-radius: 8px 8px 0 0;

 letter-spacing: 0.05em;
```

```
}

.points-table tbody tr {
   background-color: white;
   box-shadow: 0 2px 6px rgba(0,0,0,0.05);
   border-radius: 8px;
   transition: background-color 0.25s ease;
}

.points-table tbody tr:hover {
   background-color: #e6f0ff; /* subtle blue highlight */
}

.points-table td {
   border-bottom: none;
   color: #444;
}

.add-button {
   display: inline-block;
   margin-top: 25px;
   background-color:rgb(35, 60, 86);
   padding: 14px 28px;
   font-weight: 600;
   font-size: 16px;
   border-radius: 8px;
   text-decoration: none;
   box-shadow: 0 4px 10px rgba(0,123,255,0.3);
   transition: background-color 0.3s ease;
   color:white;
}

.add-button:hover {
   background-color: #0056b3;
}

   </style>
</head>
<body>

<div class="points-table-container" id="points-table-section">
   <h2>Points Table</h2>
   <table class="points-table">
     <thead>
   <tr>
      <th>Team</th>
      <th>M</th>
      <th>W</th>
      <th>L</th>
      <th>NRR</th>
      <th>Points</th>

</tr>
```

```
</thead>

      <tbody>
      <?php
      $result = $conn->query("SELECT team_id, team_name, matches_played, wins, losses,nrr,
points FROM teams ORDER BY points DESC");

      while ($row = $result->fetch_assoc()) {
         echo "<tr>
               <td>" . htmlspecialchars($row['team_name']) . "</td>
               <td>{$row['matches_played']}</td>
               <td>{$row['wins']}</td>
               <td>{$row['losses']}</td>
               <td>{$row['nrr']}</td>
               <td>{$row['points']}</td>
              </tr>";
      }
      ?>
      </tbody>
   </table>
</div>
</body>
</html>
```

Matches.php

```
<?php include 'db.php'; ?>
<?php include 'navbar.php'; ?>

<!DOCTYPE html>
<html>
<head>
   <title>Cricket Match Tracker</title>
   <style>
      /* Add this to your existing CSS in matches.php */

/* Responsive design adjustments */
@media screen and (max-width: 768px) {
   body {
      padding: 20px 15px;
   }

   .match-box {
      padding: 15px;
   }

   .team {
      min-width: 100px;
   }

   .team-name {
      font-size: 0.9rem;
   }
```

```
    .team-score {
        font-size: 0.9rem;
    }

    .dropdown-filter {
        flex-direction: column;
        align-items: flex-start;
    }

    select {
        width: 100%;
    }

    h2 {
        font-size: 1.5rem;
    }
}

/* More visible match-box layout */
.match-box {
    display: flex;
    justify-content: space-between;
    align-items: center;
    flex-wrap: wrap;
}

.match-teams {
    display: flex;
    align-items: center;
    gap: 15px;
    margin-bottom: 10px;
    flex: 1;
}

.divider {
    display: flex;
    align-items: center;
    margin: 0 8px;
}

.divider:after {
    content: "vs";
    font-weight: bold;
    color: #777;
}

.match-date {
    font-weight: 600;
    color: #555;
}
.venue{
    font-weight: 600;
```

```
  color: #555;
}
    body {
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
  background: linear-gradient(135deg, #f0f4f8 0%, #d9e2ec 100%);
  padding: 40px 30px;
  color: #2c3e50;
  min-height: 100vh;
}

h2 {
  color: #1f2d3d;
  border-bottom: 3px solid #142d3e;
  padding-bottom: 8px;
  font-weight: 700;
  font-size: 1.8rem;
  margin-bottom: 25px;
  display: flex;
  align-items: center;
  gap: 10px;
}

.match-box {
  background-color: #ffffff;
  margin: 18px 0;
  padding: 20px 25px;
  border-radius: 12px;
  box-shadow: 0 6px 12px rgba(20, 45, 62, 0.1);
  transition: box-shadow 0.3s ease, transform 0.3s ease;
  cursor: default;
}

.match-box:hover {
  box-shadow: 0 12px 24px rgba(20, 45, 62, 0.2);
  transform: translateY(-4px);
}

.dropdown-filter {
  margin-top: 15px;
  margin-bottom: 30px;
  display: flex;
  align-items: center;
  gap: 12px;
}


select {
  padding: 10px 16px;
  border-radius: 8px;
  border: 1.5px solid #142d3e;
  font-size: 15px;
  font-weight: 600;
  background-color: #ffffff;
  color: #142d3e;
```

```
    cursor: pointer;
    box-shadow: inset 0 1px 3px rgba(20, 45, 62, 0.1);
    transition: border-color 0.3s ease;
}

select:hover, select:focus {
    border-color: #2c3e50;
    outline: none;
}

label {
    font-weight: 700;
    font-size: 1rem;
    color: #142d3e;
}

p {
    margin: 0;
    color: #34495e;
    font-size: 1.1rem;
    font-weight: 600;
}

strong {
    color: #1f2d3d;
}
.team-logo {
    width: 24px;       /* smaller width */
    height: 24px;      /* smaller height */
    border-radius: 50%; /* keep circular */
    border: 1px solid #ccc;
    object-fit: contain;
    background: #fff;
}
.team {
    display: flex;
    align-items: center;
    gap: 8px;
    min-width: 140px;
}

.team-name {
    font-weight: 700;
    font-size: 1.1rem;
    color: #1f2d3d;
}

.team-score {
    font-weight: 600;
    font-size: 1rem;
    color:rgb(8, 9, 9); /* Google blue */
    text:bold;
    margin-left: 6px;
```

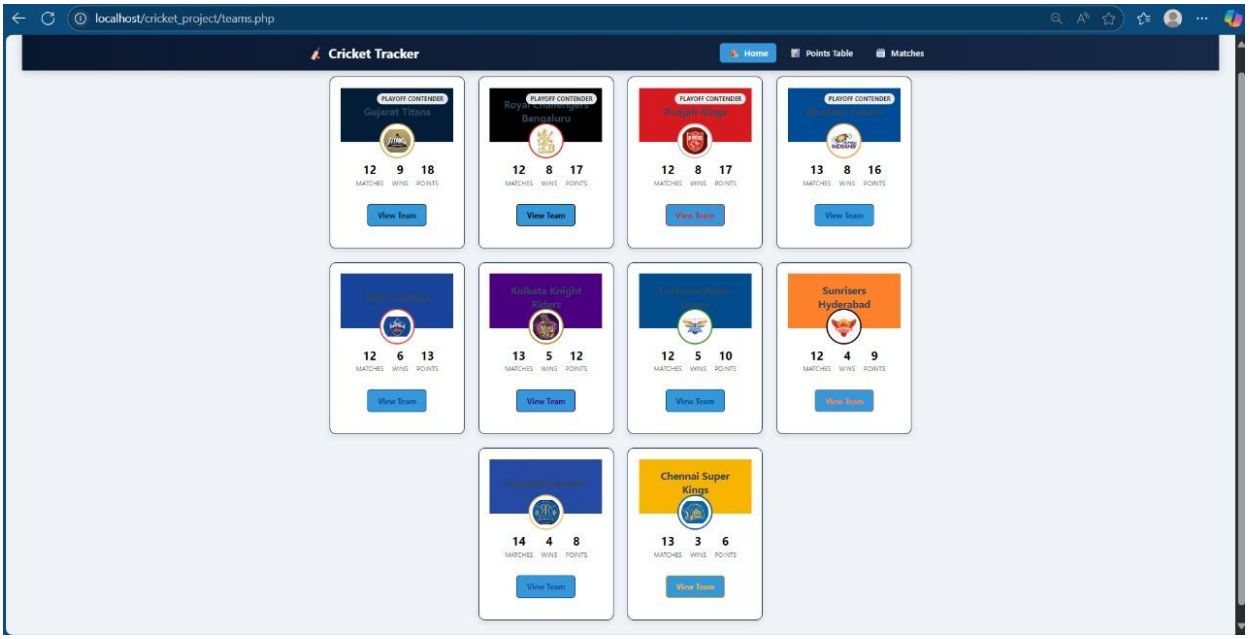
```
}


   </style>
</head>
<body>

<?php
// Fetch team data with logos
// When fetching teams, get their logo URL from this array
$team_data = [];
$res = $conn->query("SELECT team_id, team_name FROM teams");
while ($row = $res->fetch_assoc()) {
   $team_id = $row['team_id'];
   $team_data[$team_id] = [
      'name' => $row['team_name'],
      'logo' => $team_logos[$team_id] ?? 'logos/default.png' // fallback logo
   ];
}

?>
<br><br>
<h2> 🏏 Today’s Matches</h2>
<?php
$today = date('Y-m-d');
$today_matches = $conn->query("SELECT * FROM matches WHERE match_date = '$today'");
if ($today_matches->num_rows > 0) {
   while ($m = $today_matches->fetch_assoc()) {
      $team1 = $team_data[$m['team1_id']]['name'] ?? "Team {$m['team1_id']}";
      $team1_logo = $team_data[$m['team1_id']]['logo'] ?? 'default-logo.png';

      $team2 = $team_data[$m['team2_id']]['name'] ?? "Team {$m['team2_id']}";
      $team2_logo = $team_data[$m['team2_id']]['logo'] ?? 'default-logo.png';

      $score = "{$m['team1_score']} - {$m['team2_score']}";
      $venue="{$m['venue']}";
      echo "<div class='match-box'>
      <div class='match-teams'>
         <div class='team'>
            <img src='$team1_logo' alt='$team1 logo' class='team-logo'>
            <span class='team-name'>$team1</span>
            <span class='team-score'>" . ($m['team1_score'] !== null ? $m['team1_score'] : '-') .
"</span>
         </div>
         <div class='divider'></div>
         <div class='team'>
            <img src='$team2_logo' alt='$team2 logo' class='team-logo'>
            <span class='team-name'>$team2</span>
            <span class='team-score'>" . ($m['team2_score'] !== null ? $m['team2_score'] : '-') .
"</span>
         </div>
      </div>
      <div>
```

```
            <div class='match-date'>" . date('M d, Y', strtotime($m['match_date'])) . "</div>
            <div class='venue'>". ($m['venue']). "</div>
        </div>
      </div>";

    }
} else {
    echo "<p>No matches today.</p>";
}
?>

<h2>🗓️ Upcoming Matches</h2>

<form method="GET" action="" class="dropdown-filter">
    <label for="team_filter">Filter by team:</label>
    <select name="team_filter" id="team_filter" onchange="this.form.submit()">
        <option value="all">All</option>
        <?php
        $teams = $conn->query("SELECT team_id, team_name FROM teams");
        while ($team = $teams->fetch_assoc()) {
            $selected = (isset($_GET['team_filter']) && $_GET['team_filter'] == $team['team_id']) ?
'selected' : '';
            echo "<option value='{$team['team_id']}' $selected>{$team['team_name']}</option>";
        }
        ?>
    </select>
</form>

<?php
$team_filter = $_GET['team_filter'] ?? 'all';
$query = "SELECT * FROM matches WHERE match_date > '$today'";

if ($team_filter !== 'all') {
    $team_filter = intval($team_filter);
    $query .= " AND (team1_id = $team_filter OR team2_id = $team_filter)";
}
$query .= " ORDER BY match_date ASC";
$upcoming_matches = $conn->query($query);
if ($upcoming_matches->num_rows > 0) {
    while ($m = $upcoming_matches->fetch_assoc()) {
        $team1 = $team_data[$m['team1_id']]['name'] ?? "Team {$m['team1_id']}";
        $team1_logo = $team_data[$m['team1_id']]['logo'] ?? 'default-logo.png';

        $team2 = $team_data[$m['team2_id']]['name'] ?? "Team {$m['team2_id']}";
        $team2_logo = $team_data[$m['team2_id']]['logo'] ?? 'default-logo.png';

        // Scores might not be available yet for upcoming matches
        $score = ($m['team1_score'] !== null && $m['team2_score'] !== null) ?
"{$m['team1_score']} - {$m['team2_score']}" : "TBD";
         $venue="{$m['venue']}";
        $match_date = date('M d, Y', strtotime($m['match_date']));

        echo "<div class='match-box'>
```

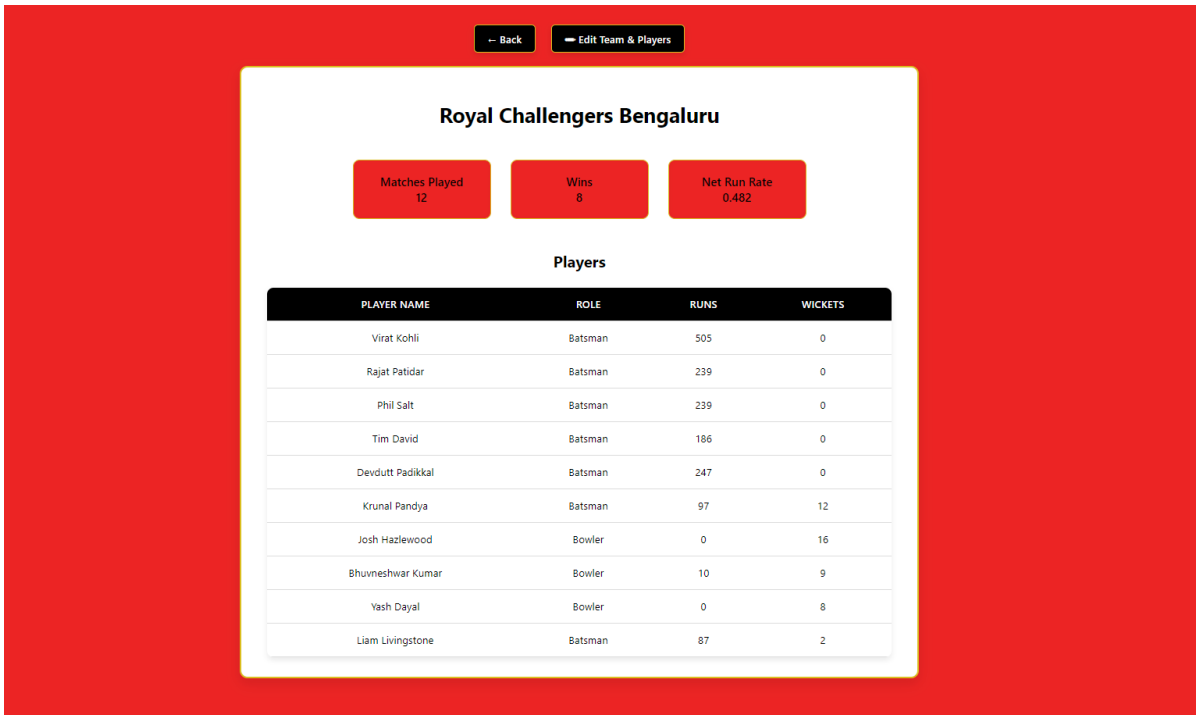
```
        <div class='match-teams'>
            <div class='team'>
                <img src='$team1_logo' alt='$team1 logo' class='team-logo'>
                <span class='team-name'>$team1</span>
                <span class='team-score'>{$m['team1_score']}</span>
            </div>
            <div class='divider'></div>
            <div class='team'>
                <img src='$team2_logo' alt='$team2 logo' class='team-logo'>
                <span class='team-name'>$team2</span>
                <span class='team-score'>{$m['team2_score']}</span>
            </div>
        </div>
        <div>
            <div class='match-date'>" . date('M d, Y', strtotime($m['match_date'])) . "</div>
             <div class='venue'>". ($m['venue']). "</div>
        </div>
      </div>";

    }
} else {
    echo "<p>No upcoming matches.</p>";
}
?>
<h2> ✅ Completed Matches</h2>

<?php
$past_matches = $conn->query("SELECT * FROM matches WHERE match_date < '$today'
ORDER BY match_date DESC");

if ($past_matches->num_rows > 0) {
    while ($m = $past_matches->fetch_assoc()) {
        $team1 = $team_data[$m['team1_id']]['name'] ?? "Team {$m['team1_id']}";
        $team1_logo = $team_data[$m['team1_id']]['logo'] ?? 'default-logo.png';

        $team2 = $team_data[$m['team2_id']]['name'] ?? "Team {$m['team2_id']}";
        $team2_logo = $team_data[$m['team2_id']]['logo'] ?? 'default-logo.png';

        $score1 = $m['team1_score'];
        $score2 = $m['team2_score'];
         $venue="{$m['venue']}";
        // Determine winner
        $winner_id = $m['winner_id']; // Get the winner_id from the database result

if ($winner_id !== null && $winner_id != 0) { // Assuming 0 or null if no winner/draw
    // Use the $team_data array (which maps team_id to team_name)
    $winner = $team_data[$winner_id]['name'] ?? "Unknown Winner";
} elseif ($score1 !== null && $score2 !== null && $score1 == $score2) {
    $winner = "Draw"; // Handle draws if winner_id is 0 or null for a draw
} else {
    $winner = "N/A"; // Or handle cases where match hasn't happened or winner isn't set yet
}
```

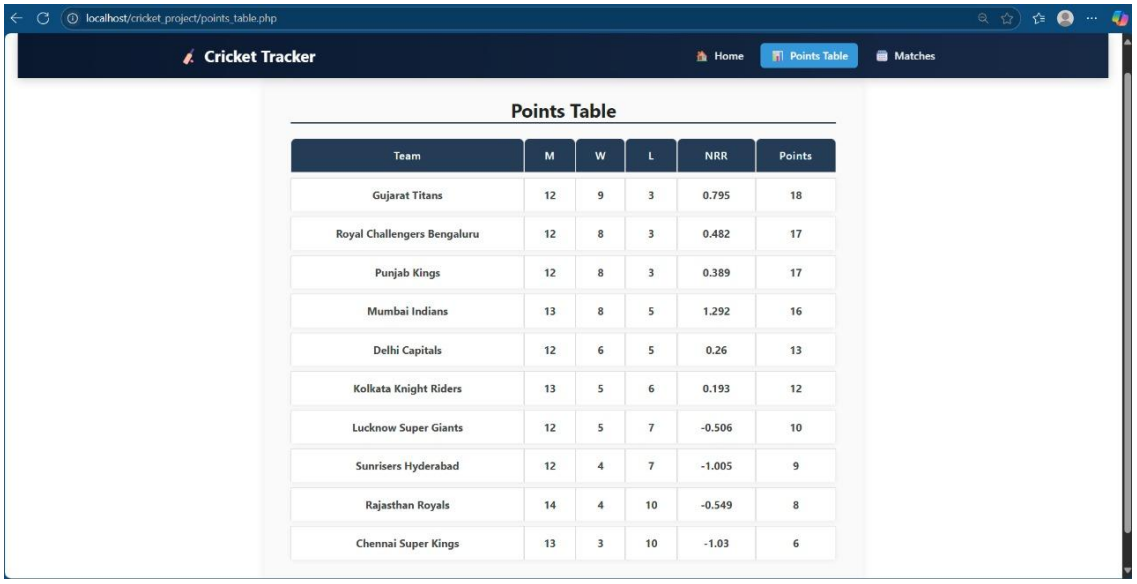
```
    $match_date = date('M d, Y', strtotime($m['match_date']));

    echo "<div class='match-box'>
          <div class='match-teams'>
            <div class='team'>
              <img src='$team1_logo' alt='$team1 logo' class='team-logo'>
              <span class='team-name'>$team1</span>
              <span class='team-score'>$score1</span>
            </div>
            <div class='divider'></div>
            <div class='team'>
              <img src='$team2_logo' alt='$team2 logo' class='team-logo'>
              <span class='team-name'>$team2</span>
              <span class='team-score'>$score2</span>
            </div>
          </div>
          <div>
            <div class='match-date'>$match_date</div>
             <div class='venue'>". ($m['venue']). "</div>
            <div class='match-winner' style='margin-top:6px; font-weight:700; color:#1a73e8;
color:black;'>Winner: $winner</div>
          </div>
        </div>";
  }
} else {
  echo "<p>No completed matches found.</p>";
}
?>
</body>
</html>
```

4.5 Discussion of Results



The interface displays team cards showing matches played, wins, and points. Playoff contenders are visually highlighted, giving users a quick glance at the top teams. This design enhances usability and facilitates team comparisons

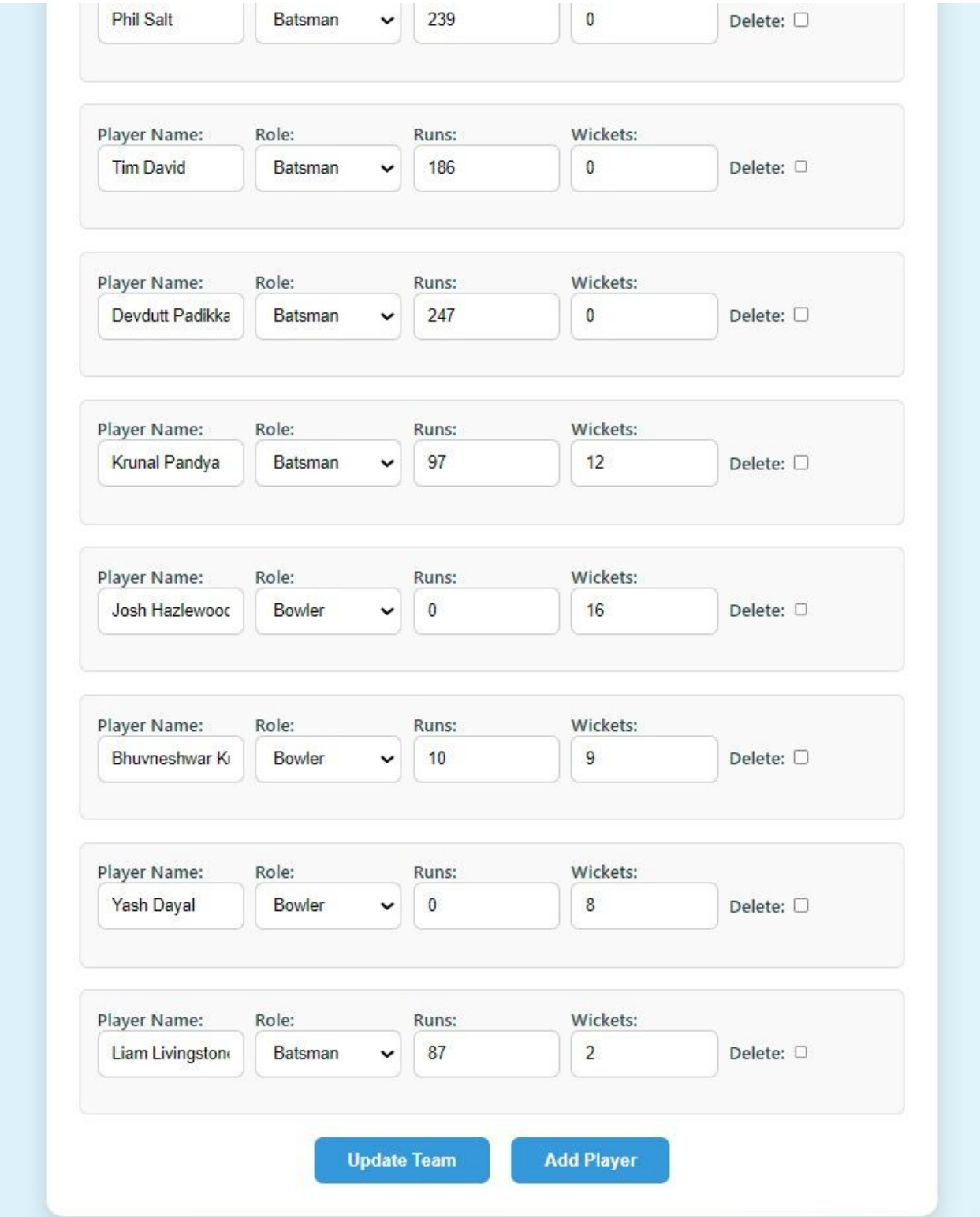




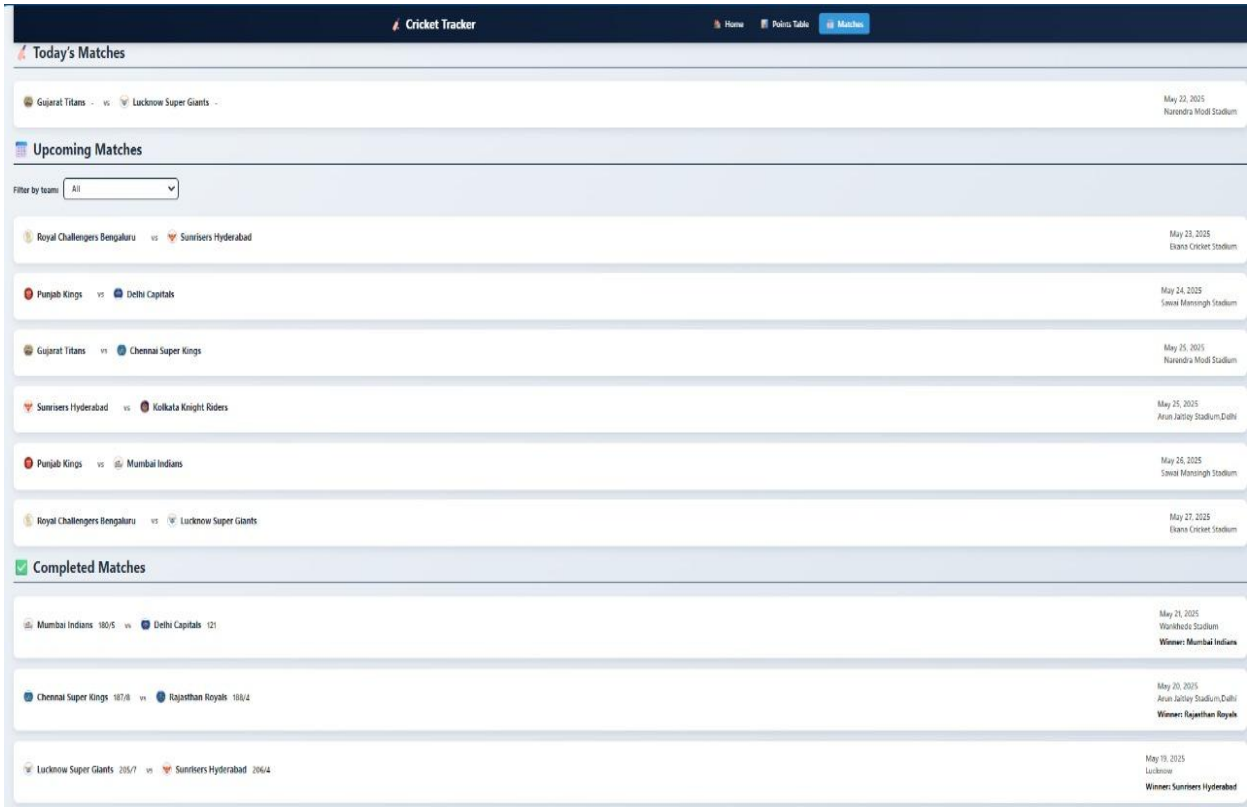
Team	M	W	L	NRR	Points
Gujarat Titans	12	9	3	0.795	18
Royal Challengers Bengaluru	12	8	3	0.482	17
Punjab Kings	12	8	3	0.389	17
Mumbai Indians	13	8	5	1.292	16
Delhi Capitals	12	6	5	0.26	13
Kolkata Knight Riders	13	5	6	0.193	12
Lucknow Super Giants	12	5	7	-0.506	10
Sunrisers Hyderabad	12	4	7	-1.005	9
Rajasthan Royals	14	4	10	-0.549	8
Chennai Super Kings	13	3	10	-1.03	6

A detailed tabular representation shows rankings based on matches played, wins, losses, net run rate (NRR), and points. Gujarat Titans lead the table with 18 points and a solid NRR, reinforcing their top position.

This page allows editing of team details such as matches played, wins, and points. It reflects a clean design supporting administrative control over team data.Allows manual editing of team stats such as wins, losses, and points.Form ensures data accuracy with proper field validations.Changes instantly reflect across related modules like the points table. Maintains consistency in team performance records.



This form-driven layout allows admin users to add, edit, or remove player records from the team. It captures key player statistics like runs and wickets, allowing for efficient management of team rosters.



Categorizes matches into Today's, Upcoming, and Completed. Displays match info like date, teams, venue, and results. Completed matches show the winner clearly for quick reference.Helps fans and admins track match timelines efficiently. Automatically updates standings and logs based on match outcomes.

4.1 Applications of project

The Cricket Database Management System (CDBMS) is not just an academic exercise—it has practical utility across a wide range of real-world scenarios where efficient management and retrieval of cricket-related data is crucial. Below are key application areas:

2.4.2.1 Cricket Tournament Management

CDBMS can be used by organizers of local, national, or international tournaments to store and manage data related to teams, players, match fixtures, results, and standings. It simplifies the tracking of tournament progress and generation of points tables.

2.4.2.2 Sports Analytics and Performance Tracking

By storing player statistics like runs, wickets, strike rates, and net run rate (NRR), the system allows analysts and coaches to evaluate individual and team performance over time. This data can be used for strategic decision-making, team selection, and training focus.

2.4.2.3 Fantasy Sports and Fan Engagement Platforms

Fantasy cricket apps and websites rely heavily on accurate and up-to-date cricket statistics. CDBMS can serve as a backend data system to provide real-time player stats, rankings, and match outcomes to support fantasy gaming.

2.4.2.4 Cricket Academies and Coaching Institutions

Training centers and cricket academies can use the system to monitor the progress of players over time. Coaches can generate reports, compare performances across matches, and guide player development based on historical data.

2.4.2.5 Media and Broadcasting

Sports journalists and broadcasters can use the database to fetch statistics quickly for commentary, match previews, and post-match analysis. It enables easy access to head-to-head records, performance charts, and team histories.

Chapter 5

CONCLUSION AND FUTURE ENHANCEMENT

5.1 Conclusion

The Cricket Database Management System (CDBMS) is a robust and scalable solution designed to efficiently manage comprehensive data related to cricket teams, players, and matches. By using a relational database structure, it ensures data consistency, integrity, and easy retrieval of information such as match outcomes, player performance, and team standings. Through the use of constraints, triggers, and normalization, the system adheres to real-world cricket rules while maintaining optimal performance.

This project not only demonstrates the practical application of database concepts but also lays a strong foundation for future enhancements such as real-time updates, analytics, and user interface integration. Overall, the system serves as a valuable tool for cricket organizers, analysts, and enthusiasts, providing a centralized platform for reliable and organized cricket data management.

5.2 Future Enhancements

1. User-Friendly Interface: Develop a web or mobile application to allow easy interaction with the database through dashboards and forms.
2. Live Data Integration: Connect the system with real-time cricket APIs to fetch and update live scores, player stats, and match results automatically.
3. Advanced Analytics: Implement data analytics to predict player performance, match outcomes, and generate detailed statistical reports.
4. Role-Based Access Control:Add secure login features with user roles like admin, team manager, and viewer for better control and data privacy.
5. Multi-Tournament Support: Extend the database to manage multiple tournaments simultaneously, each with separate teams, fixtures, and points tables.

Chapter 6

REFERENCES

[1] “Database System Concepts” by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan – foundational concepts in database design.

[2]“SQL For Dummies” by Allen G. Taylor – practical SQL guidance.

[3] “Fundamentals of Database Systems” by Ramez Elmasri and Shamkant B. Navathe – for ER modeling, normalization, and indexing.

[4] IEEE or ACM Digital Library papers on sports data modeling, performance analysis in sports, or sports event database

[5]CricAPI or CricHQ API – for real-time cricket data and stats.

[6] ESPNcricinfo API – often used for pulling historical and live match data.

[7] RapidAPI – Cricket APIs – collection of cricket data APIs.

[8] https://www.cricinfo.com – for structure and types of data cricket sites maintain.

[9] https://www.cricheroes.in – amateur and local-level cricket data tracking.

[10] MySQL, PostgreSQL, Oracle – for DBMS implementation.

[11] PHPMyAdmin, DBeaver – for database management interfaces.

[12] ERD Tools: dbdiagram.io, Lucidchart, Draw.io – for Entity-Relationship Diagrams

[13] Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). Database System Concepts (7th ed.). McGraw-Hill Education.