CRIC DASH

Designed and Developed by

J.Madhu Reddy	2211CS010238
K.Sreenath Reddy	2211CS010264
K.MalliKarjuna Rao	2211CS010271
T.Abhinay	2111CS010014

Guided by
Dr. Raviteja Kocherla
Associate Professor

Department of Computer Science & Engineering

MALLA REDDY UNIVERSITY, HYDERABAD

2024-2025

MALLA REDDY UNIVERSITY
(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

CERTIFICATE

This is to certify that the project report titled **"CRIC DASH "** has been submitted by ***J. Madhu Reddy (2211CS010238)***, ***K. Sreenath Reddy (2211CS010264)***, ***K. Mallikarjuna Rao (2211CS010271)*** and ***T.Abhinay(2111CS010014)*** students of B. Tech, 3rd Year, 2nd Semester, in the Department of Computer Science and Engineering, for the academic year (2024-2025) . The findings and methodologies documented in this report are the original work of the authors and have not been presented to any other university or institution .

Internal Guide
Dr. Raviteja Kocherla
Associate Professor

DEAN-CSE
Dr. Shaik Meeravali

External Examiner

MALLA REDDY UNIVERSITY

(Telangana State Private Universities Act No.13 of 2020 and G.O.Ms.No.14, Higher Education (UE) Department)

DECLARATION

I hereby declare that the project report titled **"CRIC DASH"**, submitted in partial fulfillment of the requirements for the degree of B. Tech in Computer Science and Engineering, is an original work conducted by me under the supervision of ***Dr.Raviteja Kocherla (Associate Professor)***. This report has not been submitted as the basis for the award of any other degree or diploma at any other institution or university. In alignment with ethical practices in scientific reporting, appropriate acknowledgments have been provided wherever the findings of others have been referenced.

Signature

J.MADHU REDDY	(2211CS010238)
K.SREENATH REDDY	(2211CS010264)
K.MALLIKARJUNA RAO	(2211CS010271)
T.ABHINAY	(2111CS010014)

ACKNOWLEDGEMENT

I would like to express my sincere thanks to **Dr. Shaik Meeravali** (DEAN-CSE) sir for providing us this opportunity to explore our knowledge and for giving us an opportunity to improve our skills and also providing us a good guidance, for providing all the facilities required to complete this project. We express our heart full thanks to the Head of the Department, Department of Computer Science and Engineering, for all the help and infrastructure provided to us to complete the project successfully and his valuable guidance. We are also thankful to all other faculty and staff members of our department for their kind cooperation and help.

I would like to express my deepest gratitude to my guide, ***Dr.Raviteja Kocherla*** sir for his valuable guidance, consistent encouragement, personal caring, timely help and providing us with an excellent atmosphere for doing research. All through the work, in spite of his busy schedule, he has extended cheerful and cordial support to us for completing this research work. he has been helpful from beginning till the end of this project. We are thankful to sir for the encouragement he has given to us in completing the project.

I would also like to thank all of the other supporting personal who assisted me by supplying the equipment that was essential and vital, without which I would not have been able to perform efficiently on this project. I would also want to thank the Malla Reddy University for accepting my project in my desired field of expertise. I'd also like to thank my friends and parents for their support and encouragement as I worked on this assignment.

ABSTRACT

CRIC DASH is a comprehensive cricket data analytics platform designed to offer an engaging and informative experience for cricket enthusiasts, analysts, and casual viewers. In a sport where numbers and performance metrics hold immense value, CRIC DASH brings clarity and accessibility to complex cricket data through an intuitive interface and well-structured visualizations. The platform provides complete player statistics across batting, bowling, and fielding dimensions, enabling users to explore key performance indicators such as total runs, wickets, economy rates, strike rates, boundaries, and catches. These insights are derived from large-scale datasets that are meticulously cleaned, validated, and formatted into structured JSON files. Player profiles are enhanced with images and team affiliations, which support better recognition and intuitive comparisons across the platform.

A standout feature of CRIC DASH is the **Stats Comparison** tool, where users can select two players and compare their performances side-by-side using dynamic charts, cards, and visual indicators. This feature is particularly useful for analyzing matchups, debating team selections, or evaluating top performers. The application boasts a modern, dark-themed UI with vibrant accents, smooth transitions, and a responsive design for seamless access across devices. While the current version focuses on historical data, CRIC DASH is architected with scalability in mind—paving the way for future integration of live match data, tournament filters, and advanced analytics. Ultimately, CRIC DASH bridges the gap between raw statistics and cricket storytelling, empowering users to uncover insights, fuel data-driven discussions, and deepen their understanding of the game.

Table of Contents

Description	Page Number
Certificate	ii
Declaration	iii
Acknowledgement	iv
Abstract	v
List of Figures	viii
List of Tables	ix
Chapter 1: Introduction	
1.1 Introduction	1
1.2 Problem Statement	1
1.3 Objective of Project	2
1.4 Goal of Project	3
Chapter 2: Problem Identification	
2.1 Existing System	5s
2.2 Proposed System	6
Chapter 3: Requirements	
3.1 Software Requirements	8
3.2 Hardware Requirements	8
Chapter 4: Design and Implementation	
4.1 Design	9

4.2 Implementation

Chapter 5: Code and Screenshots

5.1 Source Code 14-28

5.2 Screenshot of Application 29-33

Chapter 6: Results & Conclusion

6.1 Results 35

6.2 Conclusion

References 35-36

LIST OF FIGURES

FIGURE	TITLE	PAGE NUMBER
4.1.1	Data-Flow Diagram	10
4.1.2	Use Case Diagram	11
4.1.3	E R Diagram	12
4.1.4	Sequence Diagram	13

LIST OF SCREENSHOTS

SCREEN SHOTS	TITLE	PAGE NO.
5.2.1	Home page	29
5.2.2	Yearly Schedule	29
5.2.3	Batting Stats	30
5.2.4	Bowling Stats	30
5.2.5	Exploring stats	31
5.2.6	Highest Score in an innings	31
5.2.7	Best Bowling figures	32
5.2.8	Stats Comparison	32
5.2.9	Stats comparison 2	33

CHAPTER-1

INTRODUCTION

1.1 Introduction

Cricket, being one of the most followed sports globally, generates an enormous amount of data every season. However, this data is often complex and scattered, making it difficult for fans and analysts to easily access, understand, and compare statistics. *CRIC DASH* is a cricket data analytics and visualization platform designed to address this challenge by offering a structured, intuitive, and interactive way to explore comprehensive player and team statistics.

This project focuses on providing a clean and modern interface for users to access complete batting, bowling, and fielding data of players across various formats and seasons. By offering features like individual player performance, team statistics, and head-to-head player comparisons, *CRIC DASH* makes cricket analytics accessible not only to statisticians and experts but also to everyday fans. The application aims to bridge the gap between raw statistical data and user-friendly insights.

1.2 Problem Statement

Despite the abundance of cricket statistics available online, accessing and comparing player or team data in a meaningful way remains a challenge. Key issues include:

- Lack of unified platforms providing **complete player statistics** across categories (batting, bowling, and fielding).
- Inadequate **comparison tools** to analyze two or more players across multiple dimensions.
- Limited visual representation of data, making it hard for non-experts to interpret insights.
- No provision for **real-time filtering** or viewing data based on specific criteria (e.g., format, team, match type, etc.).

These limitations create a gap between cricket data availability and its usability for analysis, learning, and informed discussions among fans and experts alike.

1.3 Objective

The primary objectives of the *CRIC DASH* project are as follows:

- To develop a web-based platform that presents structured and categorized cricket statistics for every player.
- To enable **real-time side-by-side comparison** between two players across all performance metrics.
- To offer an **interactive and visually engaging UI/UX** for exploring player data with ease.
- To create a scalable structure that can later integrate live match data or additional features like venue-based stats or seasonal trends.

Solution

To address the above challenges, *CRIC DASH* proposes the development of a full-stack cricket analytics system. The core components of the solution include:

1. **Data Collection and Structuring**
   Collection of structured cricket data and conversion into optimized JSON format for easy access and retrieval. The dataset includes detailed records for batting, bowling, and fielding performances.
2. **Frontend Development (React.js)**
   A responsive and visually rich user interface is developed using React.js. Users can view player stats, perform comparisons, and navigate through data effortlessly.
3. **Backend Integration**
   The backend serves structured cricket data stored in JSON files. These files are preprocessed for accuracy and delivered to the frontend through lightweight APIs, enabling fast and seamless access to player stats and comparisons.
4. **Comparison Engine**
   A custom comparison engine allows users to select any two players and view their statistical performance across multiple parameters in a side-by-side format.

5. **UI Enhancements and Visualizations**

   Data is presented using clean cards, graphs, and visual cues to make it easier for users to comprehend trends, differences, and highlights.

Expected Outcomes

- A fully functional cricket analytics dashboard that provides access to individual and comparative statistics.
- A rich user interface with clear visualizations for batting, bowling, and fielding performances.
- A scalable backend that can accommodate new data and future expansion such as live updates or filters.
- Increased accessibility and understanding of cricket data for casual fans, content creators, analysts, and professionals.

1.4 Goal of The Project

The goal of *CRIC DASH* is to create an inclusive, easy-to-use, and visually intuitive platform for exploring and comparing cricket statistics. By transforming scattered and raw cricket data into meaningful visual insights, the platform empowers users to make informed analyses, predictions, and engage in data-driven discussions.

The project is not just about stats display; it's about **storytelling through data**, fostering deeper engagement among cricket fans, and enabling smarter insights into player and team performance. Eventually, the platform aims to expand into a full-fledged cricket intelligence system, integrating filters, historical trends, and possibly real-time data.

.

CHAPTER – 2
PROBLEM IDENTIFICATION

2.1 Existing System

Currently, most platforms that provide cricket statistics focus either on live score updates or basic player performance summaries. These systems are often limited in scope and lack comprehensive features for comparative and historical analysis. Cricket fans, analysts, and researchers are left to manually extract and analyze data from multiple sources, which is both time-consuming and error-prone.

Limitations in Current Systems:

1. **Lack of Unified Player Statistics:**
   Most cricket-related platforms do not provide consolidated statistics covering **batting, bowling, and fielding** in one place. Users must switch between pages or even websites to access full performance data.
2. **Inadequate Comparative Analysis Tools:**
   There are few tools that allow users to **compare two players side-by-side** across various metrics such as strike rate, economy, wickets, runs, catches, etc.
3. **Minimal Interactivity and Visual Appeal:**
   Most existing cricket dashboards present data in a static format (tables or text), making it less engaging and difficult for users to gain insights at a glance.
4. **No Player Insights for Casual Fans:**
   The available platforms often cater to cricket analysts or hardcore fans, not beginners or casual followers who need simplified and visual content.
5. **Scattered and Unstructured Data Access:**
   Access to historical player data is fragmented. Users must browse through multiple databases or websites without a streamlined search or filter option.

2.2 Proposed System

To address the limitations of existing systems, *CRIC DASH* proposes an **interactive and comprehensive cricket data analytics platform** that focuses on **clean UI, detailed statistics, and smart comparison features**. The goal is to make data exploration more accessible, insightful, and engaging for all levels of cricket fans.

Key Features of the Proposed System:

1. **Complete Player Statistics in One Place**
    - Covers **batting, bowling, and fielding** stats of every player in a unified, structured format.
    - Displays career totals, match-wise breakdowns, and specific tournament or season-based data.
2. **Player Comparison Tool**
    - Users can select **two players** and view their statistics side-by-side.
    - Provides comparison across all key performance indicators (KPIs) such as total runs, wickets, averages, strike rates, and more.
3. **Clean & Modern UI**
    - Designed using **React.js** to provide a visually appealing and intuitive interface.
    - Mobile-responsive and optimized for smooth navigation and readability.
4. **Backend Efficiency**
    - Lightweight backend serves preprocessed JSON data for fast player stats retrieval.
    - Modular API structure allows easy expansion for features like team-wise stats, match filters, and seasonal views.
5. **Scalable and Extensible Architecture**
    - System is designed to scale by adding features like live stats, filters by format (ODI, T20, Test), or even match conditions and venue-based analytics.

CHAPTER -3

REQUIREMENTS

3.1. Software Requirements

Development Tools

- **React.js**: For building the user interface of the CRIC DASH application.
- **Python (Flask)**: For building the backend that handles data processing and serving predictions.
- **Jupyter Notebook / VS Code**: For development and testing of Python scripts related to data preprocessing and sentiment analysis.

Python Libraries

- **Pandas**: Used for loading, cleaning, and manipulating tweet datasets.
- **Matplotlib & Seaborn**: Used for creating visualizations of sentiment trends.
- **NumPy**: Provides numerical operations to support data processing tasks.
- **Scikit-learn**: Used to build and train machine learning models (e.g., Logistic Regression).
- **NLTK (Natural Language Toolkit)**: For natural language processing tasks, including stopword removal, tokenization, and basic text normalization.
- **CountVectorizer & TF-IDF Vectorizer**: For converting textual data into numerical form suitable for ML models.

Frontend Libraries/Tools

- **Axios or Fetch API**: For making HTTP requests from the React frontend to the Flask backend.
- **Chart.js / Recharts**: For visualizing sentiment results in the dashboard.

3.2. Hardware Requirements

Minimum Hardware Requirements

- **Processor**: Intel Core i5 or equivalent
- **RAM**: Minimum 8 GB (16 GB recommended for smoother performance during model training and testing)
- **Storage**: At least 20 GB of free disk space for datasets, logs, and processed files.

Network Requirements

- **Stable Internet Connection**: Required for:
    - Downloading datasets and Python libraries.
    - Running the React app if deployed using online services like GitHub Pages or Netlify.
    - Interacting with the backend API if deployed on cloud platforms.

CHAPTER-4
DESIGN AND IMPLEMENTATION

DATA FLOW DIAGRAM :

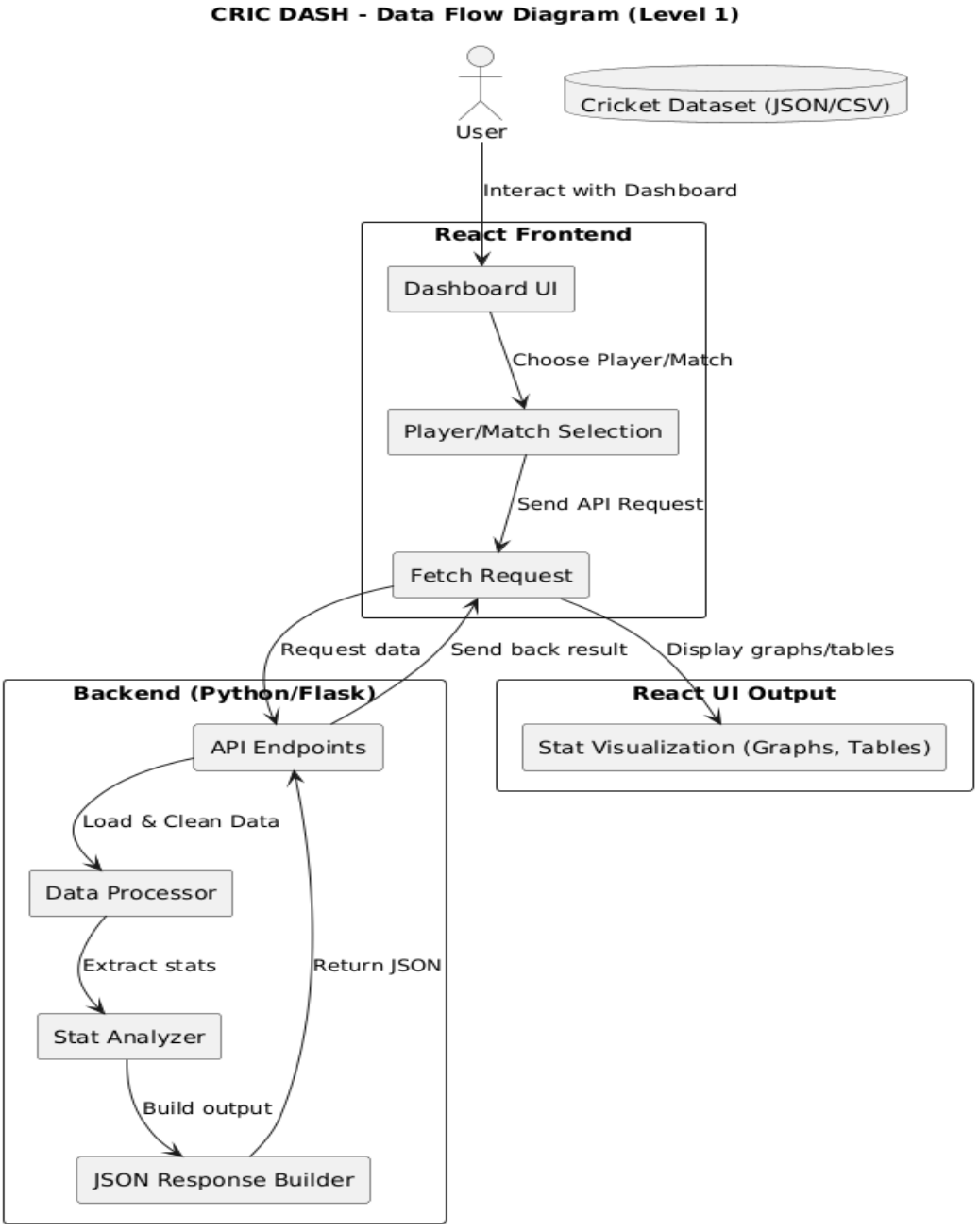


fig.4.1.1 Data Flow Diagram

USE CASE DIAGRAM:

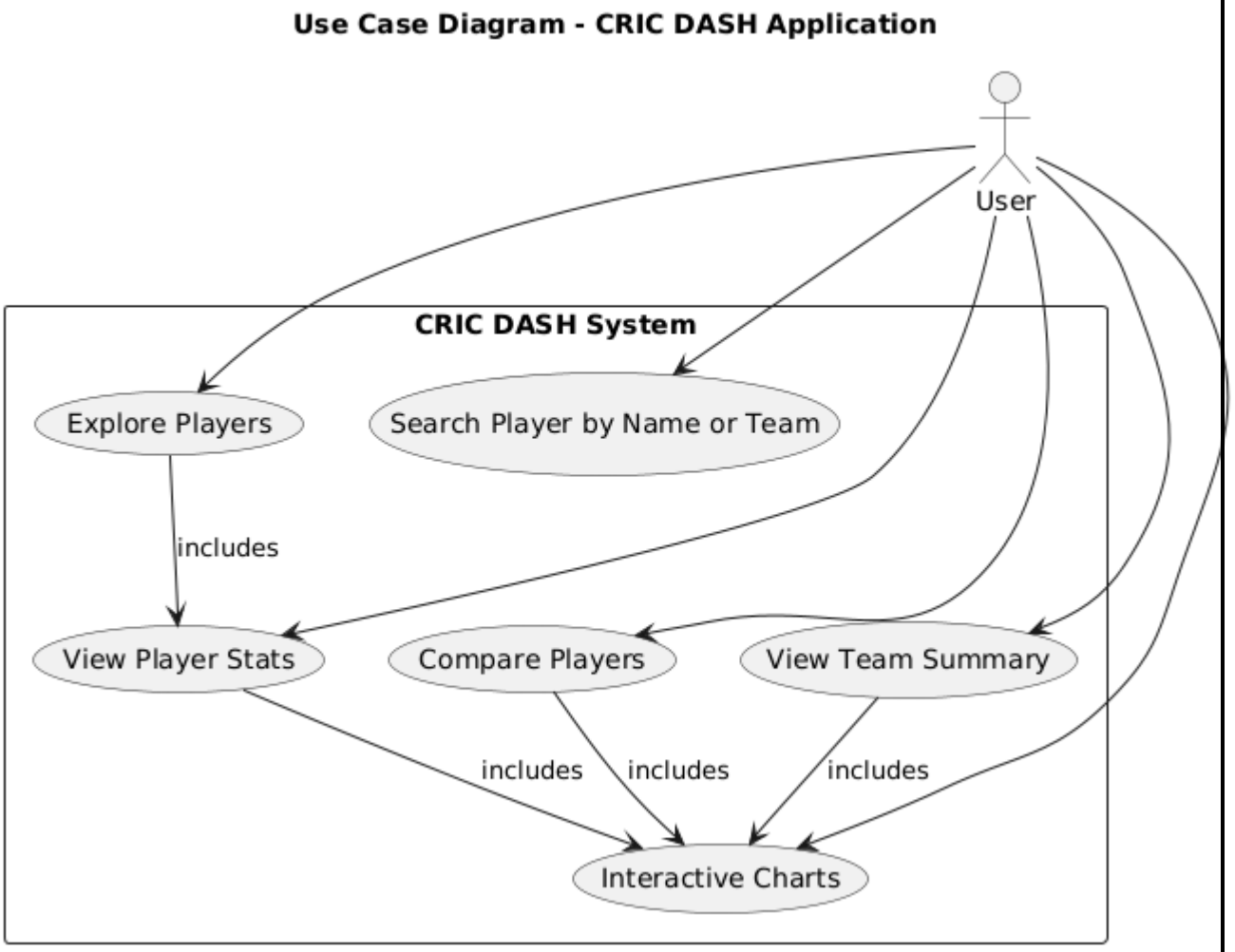


fig.4.1.2 Use Case Diagram

ER DIAGRAM:

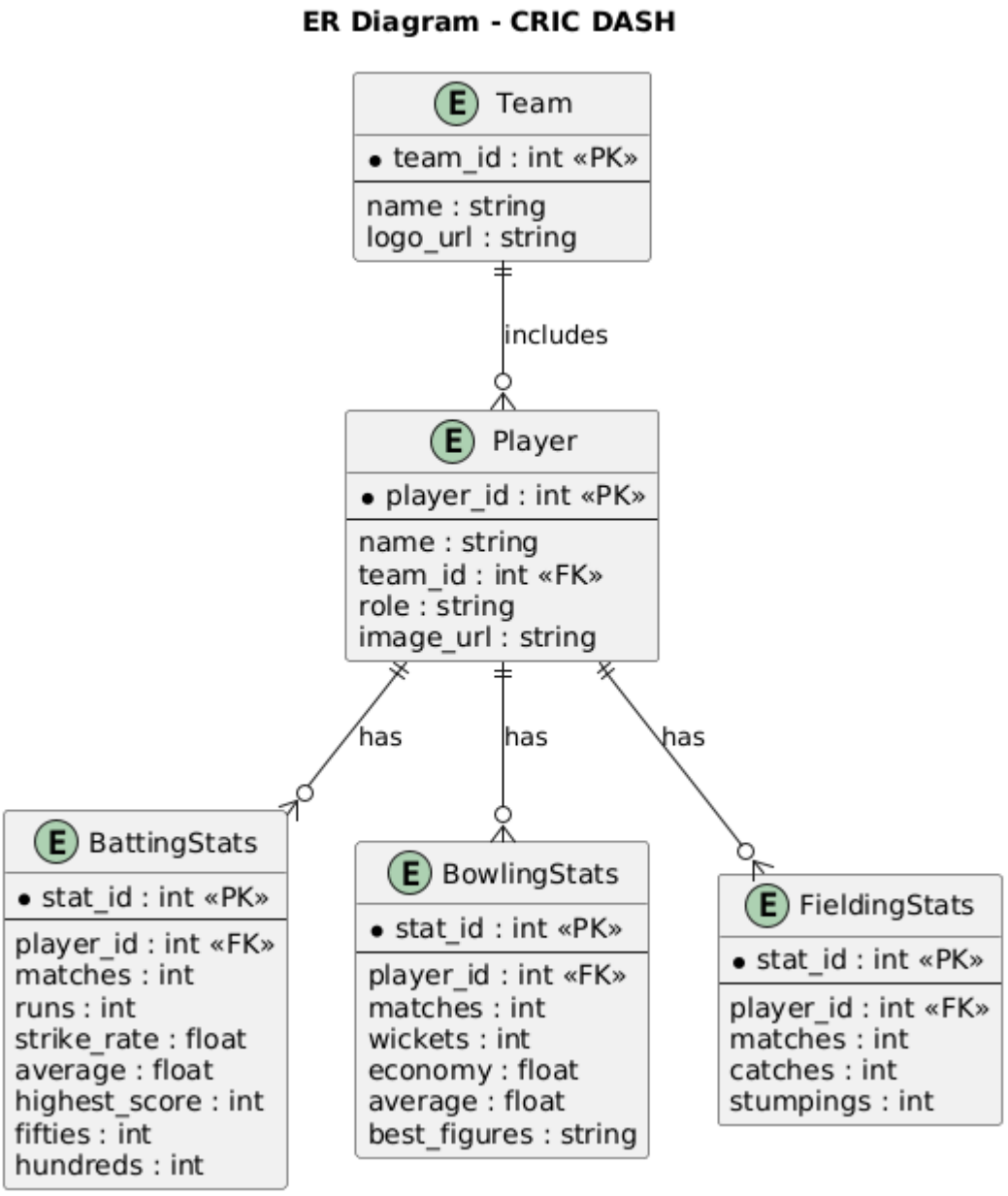


fig. 4.1.3 E R Diagram

SEQUENCE DIAGRAM:

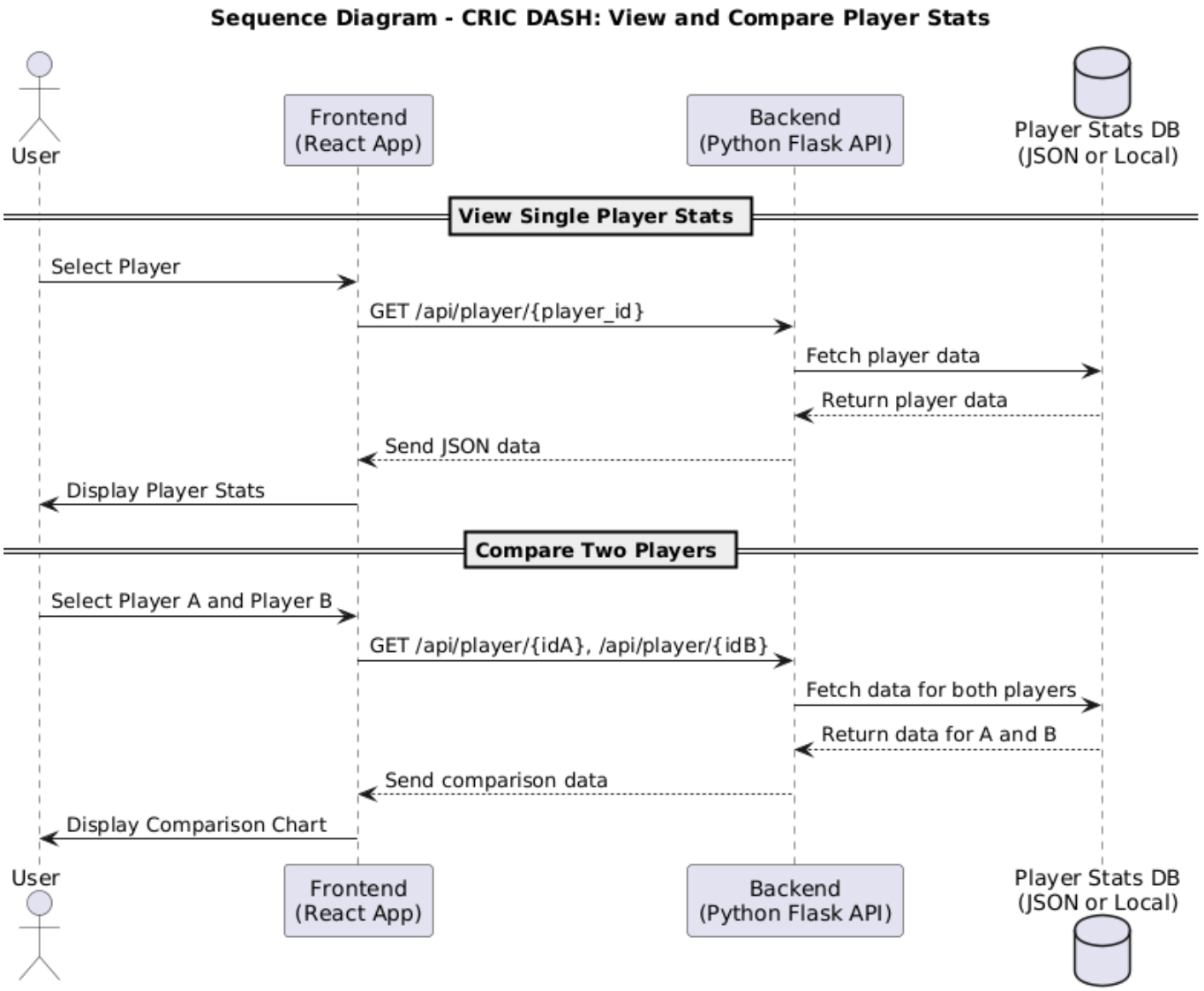


fig.4.1.4 Sequence Diagram

CHAPTER - 5

SOURCE CODE

Python Code (Backend):

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import json
df = pd.read_csv("IPL_Histroy_2008-2024.csv")
df.columns = df.columns.str.strip()
batting_stats = df.groupby('batter').agg(
    total_runs=('batsman_runs', 'sum'),
    balls_faced=('batter', 'count'),
    fours=('batsman_runs', lambda x: (x == 4).sum()),
    sixes=('batsman_runs', lambda x: (x == 6).sum())
).reset_index()

batting_stats['strike_rate'] = (batting_stats['total_runs'] / batting_stats['balls_faced']) * 100
batting_stats = batting_stats.sort_values(by='total_runs', ascending=False)

# Extract teams played by each player
teams_played = df.groupby('batter')['batting_team'].unique().apply(lambda x: list(x)).reset_index()
batting_stats = batting_stats.merge(teams_played, on='batter', how='left')
batting_stats_json = batting_stats.rename(columns={
    'batter': 'name',
    'batting_team': 'teams_played',
    'total_runs': 'total_runs',
    'balls_faced': 'balls_faced',
    'strike_rate': 'strike_rate',
    'fours': 'fours',
```

```
    'sixes': 'sixes'
}).to_dict(orient='records')

with open("batting_stats.json", "w") as json_file:
    json.dump(batting_stats_json, json_file, indent=4)
players = batting_stats['batter'].tolist()
player_urls = [{"Name": player, "Url": f"https://scores.iplt20.com/ipl/playerimages/{player.replace('
', '%20')}.png?v=1"} for player in players]

with open("player_urls.json", "w") as json_file:
    json.dump(player_urls, json_file, indent=4)

print("Batting statistics saved to batting_stats.json")
print("Player URLs saved to player_urls.json")
```

React Code (Frontend):

App.js

```
import React from "react";
import { BrowserRouter as Router, Route, Routes } from "react-router-dom";
import Home from "./components/Home";
import Matches from "./components/Matches";
import PlayerCards from "./components/PlayerCards";
import PlayerTable from "./components/PlayerTable";
import PlayerDetails from "./components/PlayerDetails";
import Explore from "./components/Explore"; //
import HighestRunScorer from "./components/HighestRunScorer";
import BestBowlingFigures from "./components/BestBowlingFigures";
import Stadiums from "./components/Stadiums";
import StatsComparison from "./components/StatsComparison";
import "./App.css"; // Import styles
```

```
const App = () => {
  return (
    <div className="App">
      <Router>
        <Routes>
          <Route path="/" element={<Home />} />
          <Route path="/matches" element={<Matches />} />
          <Route path="/players" element={<PlayerCards />} />
          <Route path="/table" element={<PlayerTable />} />
          <Route path="/player/:name" element={<PlayerDetails />} />
          <Route path="/explore" element={<Explore />} />
          <Route path="/explore/highest-run-scorer" element={<HighestRunScorer />} />
          <Route path="/explore/best-bowling-figures" element={<BestBowlingFigures />} />
          <Route path="/explore/stadiums" element={<Stadiums />} />
          <Route path="/explore/stats-comparison" element={<StatsComparison />} />
        </Routes>
      </Router>
    </div>
  );
};

export default App;
```

Home.js

```
import React, { useState, useEffect } from "react";
import { Link } from "react-router-dom";
import "./Home.css";
import cskImage from "./CSK.png";
import srhImage from "./SRH.jpg";
import miImage from "./MI.png";
```

```
import kkrImage from "./KKR.png";
import dcImage from "./DC.png";
import rrImage from "./RR.png";
import rcbImage from "./RCB.png";
import { Home as HomeIcon, Calendar, BarChart, Info } from "lucide-react";

const teams = {
  CSK: { image: cskImage, primaryColor: "#FFC72C" },
  SRH: { image: srhImage, primaryColor: "#FF4500" },
  MI: { image: miImage, primaryColor: "#045093" },
  KKR: { image: kkrImage, primaryColor: "#3A225D" },
  DC: { image: dcImage, primaryColor: "#17449B" },
  RR: { image: rrImage, primaryColor: "#EA1A80" },
  RCB: { image: rcbImage, primaryColor: "#DA1818" },
};

const teamKeys = Object.keys(teams);

const Home = () => {
  const [teamIndex, setTeamIndex] = useState(0);
  const [fade, setFade] = useState("fade-in");

  useEffect(() => {
    const interval = setInterval(() => {
      setFade("fade-out");
      setTimeout(() => {
        setTeamIndex((prevIndex) => (prevIndex + 1) % teamKeys.length);
        setFade("fade-in");
      }, 1000);
    }, 5000);
```

```
  return () => clearInterval(interval);
}, []);

return (
  <div
    className={`home-container ${fade}`}
    style={{ backgroundImage: `url(${teams[teamKeys[teamIndex]].image})` }}
  >
    <div className="overlay"></div>

    <div className="hero">
      <h1 className="hero-title">CRIC DASH</h1>
      <p className="hero-subtitle">Your Ultimate Cricket Dashboard</p>
      <div className="hero-buttons">
        <Link to="/explore" className="btn">
          Explore Now
        </Link>
      </div>
    </div>

    {/* Fixed Bottom Navigation */}
    <nav className="bottom-nav">
<Link to="/" className="nav-item">
  <HomeIcon size={24} />
  <span>Home</span>
</Link>
<Link to="/matches" className="nav-item"> {/* Updated Link */}
  <Calendar size={24} />
  <span>Matches</span>
</Link>
<Link to="/table" className="nav-item">
```

```
    <BarChart size={24} />
    <span>Stats Table</span>
  </Link>
  <Link to="/about" className="nav-item">
    <Info size={24} />
    <span>About</span>
  </Link>
</nav>

    </div>
  );
};

export default Home;
```

PlayerDetails.js

```
import React, { useEffect, useState } from "react";
import { useParams } from "react-router-dom";
import "./PlayerDetails.css";

const PlayerDetails = () => {
  const { name } = useParams();
  const [player, setPlayer] = useState(null);

  useEffect(() => {
    fetch("https://sreenathkalluri.github.io/AD3_2/all_batsmen%20(1).json")
      .then((res) => res.json())
      .then((data) => setPlayer(data.find((p) => p.Name === name)))
      .catch((err) => console.error("Error fetching data:", err));
  }, [name]);
```

```
if (!player) return <div className="loading">Loading Player Stats...</div>;

return (
  <div className="player-details">
    <h1>{player.Name}</h1>
    <img
      src={player.ImageURL} // Directly using the image URL from JSON
      alt={player.Name}
      className="player-image"
      onError={(e) => {
        e.target.src = "https://via.placeholder.com/150"; // Fallback image
      }}
    />
    <table>
      <tbody>
        <tr>
          <td>Team</td>
          <td>{player.Team}</td>
        </tr>
        <tr>
          <td>Runs</td>
          <td>{player.RunsScored}</td>
        </tr>
        <tr>
          <td>Innings</td>
          <td>{player.InningsBatted}</td>
        </tr>
        <tr>
          <td>Average</td>
          <td>{player.BattingAVG}</td>
        </tr>
```

```
        <tr>
          <td>Strike Rate</td>
          <td>{player["BattingS/R"]}</td>
        </tr>
        <tr>
          <td>4s</td>
          <td>{player["4s"]}</td>
        </tr>
        <tr>
          <td>6s</td>
          <td>{player["6s"]}</td>
        </tr>
        <tr>
          <td>Teams Played</td>
          <td>{player.Teams_Played}</td>
        </tr>
      </tbody>
    </table>
  </div>
 );
};

export default PlayerDetails;
```

StatsComparsion.js

```
import React, { useEffect, useState } from "react";
import { Bar } from "react-chartjs-2";
import {
 Chart as ChartJS,
 CategoryScale,
 LinearScale,
```

```
  BarElement,
  Title,
  Tooltip,
  Legend,
} from "chart.js";
import "./StatsComparison.css";

ChartJS.register(CategoryScale, LinearScale, BarElement, Title, Tooltip, Legend);

const StatsComparison = () => {
  const [playersData, setPlayersData] = useState([]);
  const [player1, setPlayer1] = useState("");
  const [player2, setPlayer2] = useState("");
  const [player1Stats, setPlayer1Stats] = useState(null);
  const [player2Stats, setPlayer2Stats] = useState(null);

  useEffect(() => {
    fetch("https://sreenathkalluri.github.io/AD3_2/all_stats.json")
      .then((res) => res.json())
      .then((data) => setPlayersData(data));
  }, []);

  useEffect(() => {
    setPlayer1Stats(playersData.find((p) => p.player === player1));
    setPlayer2Stats(playersData.find((p) => p.player === player2));
  }, [player1, player2, playersData]);

  const getPlayerOptions = () => {
    return playersData.map((p) => p.player).sort();
  };
```

```
const renderStat = (label, value) => (
  <div className="stat-item">
    <strong>{label}:</strong> {value !== undefined ? value : "N/A"}
  </div>
);

const PlayerCard = ({ stats }) => {
  if (!stats) return null;

  const fallbackImage = `https://scores.iplt20.com/ipl/playerimages/${stats.player.replaceAll(
    " ",
    "%20"
  )}.png?v=1`;

  return (
    <div className="player-card">
      <img
        src={fallbackImage}
        alt={stats.player}
        className="player-img"
        onError={(e) => (e.target.style.display = "none")}
      />
      <h2>{stats.player}</h2>

      {stats.total_runs !== undefined && (
        <div className="stat-section">
          <h3>Batting</h3>
          {renderStat("Total Runs", stats.total_runs)}
          {renderStat("Balls Faced", stats.balls_faced)}
          {renderStat("Fours", stats.fours)}
          {renderStat("Sixes", stats.sixes)}
```

```
          {renderStat("Dismissals", stats.dismissals)}
          {renderStat("Strike Rate", stats.strike_rate)}
          {renderStat("Batting Avg", stats.batting_avg)}
        </div>
      )}

      {stats.wickets !== undefined && (
        <div className="stat-section">
          <h3>Bowling</h3>
          {renderStat("Wickets", stats.wickets)}
          {renderStat("Runs Conceded", stats.runs_conceded)}
          {renderStat("Balls Bowled", stats.balls_bowled)}
          {renderStat("Overs", stats.overs)}
          {renderStat("Economy", stats.economy)}
          {renderStat("Best Figures", stats.best_figures)}
        </div>
      )}

      {(stats.catches !== undefined ||
        stats.stumpings !== undefined ||
        stats.run_outs !== undefined) && (
        <div className="stat-section">
          <h3>Fielding</h3>
          {renderStat("Catches", stats.catches)}
          {renderStat("Stumpings", stats.stumpings)}
          {renderStat("Run Outs", stats.run_outs)}
        </div>
      )}
    </div>
  );
};
```

```
const getComparisonChart = () => {
 if (!player1Stats || !player2Stats) return null;

 const compareFields = [
  { label: "Total Runs", key: "total_runs" },
  { label: "Wickets", key: "wickets" },
  { label: "Catches", key: "catches" },
  { label: "Strike Rate", key: "strike_rate" },
  { label: "Batting Avg", key: "batting_avg" },
  { label: "Economy", key: "economy" },
 ];

 const labels = compareFields.map((field) => field.label);
 const player1Values = compareFields.map(
  (field) => -(player1Stats[field.key] || 0) // NEGATIVE to mirror
 );
 const player2Values = compareFields.map(
  (field) => player2Stats[field.key] || 0
 );

 const data = {
  labels,
  datasets: [
   {
    label: player1Stats.player,
    data: player1Values,
    backgroundColor: "#00e5ff",
   },
   {
    label: player2Stats.player,
```

```
      data: player2Values,
      backgroundColor: "#ff4081",
    },
  ],
};

const options = {
  responsive: true,
  indexAxis: "y",
  plugins: {
    legend: {
      position: "top",
      labels: { color: "#fff" },
    },
    title: {
      display: true,
      text: "Stat Comparison",
      color: "#fff",
    },
    tooltip: {
      callbacks: {
        label: function (context) {
          return `${context.dataset.label}: ${Math.abs(context.raw)}`;
        },
      },
    },
  },
  scales: {
    x: {
      stacked: true,
      ticks: {
```

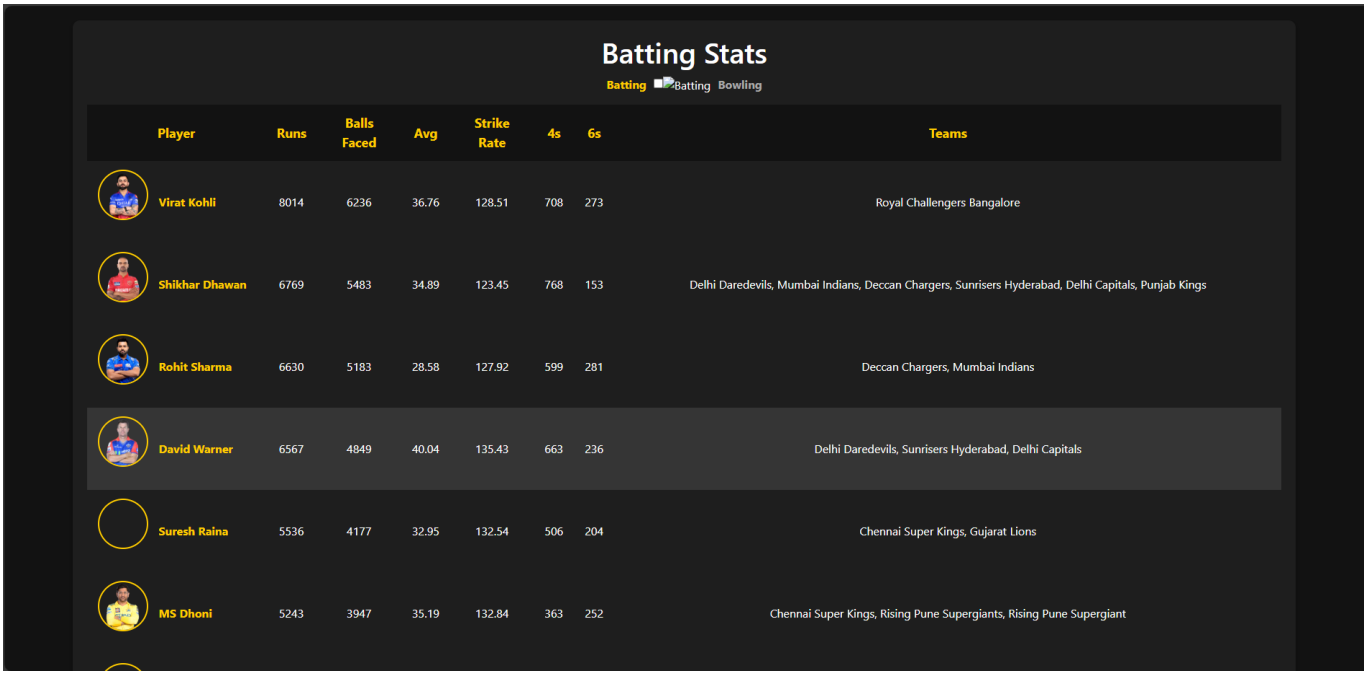
```
        color: "#fff",
        callback: function (value) {
          return Math.abs(value); // Show only positive values
        },
      },
      grid: { color: "#444" },
    },
    y: {
      stacked: true,
      ticks: { color: "#fff" },
      grid: { color: "#444" },
    },
  },
 };

 return (
  <div className="chart-column">
   <Bar data={data} options={options} />
  </div>
 );
};

return (
 <div className="comparison-container">
  <h1>Stats Comparison</h1>
  <div className="dropdowns">
   <select value={player1} onChange={(e) => setPlayer1(e.target.value)}>
    <option value="">Select Player 1</option>
    {getPlayerOptions().map((name) => (
     <option key={name} value={name}>
      {name}
```

```
            </option>
          ))}
        </select>

        <select value={player2} onChange={(e) => setPlayer2(e.target.value)}>
          <option value="">Select Player 2</option>
          {getPlayerOptions().map((name) => (
            <option key={name} value={name}>
              {name}
            </option>
          ))}
        </select>
      </div>

      <div className="stats-compare-section">
        <PlayerCard stats={player1Stats} />
        {getComparisonChart()}
        <PlayerCard stats={player2Stats} />
      </div>
    </div>
  );
};

export default StatsComparison;
```

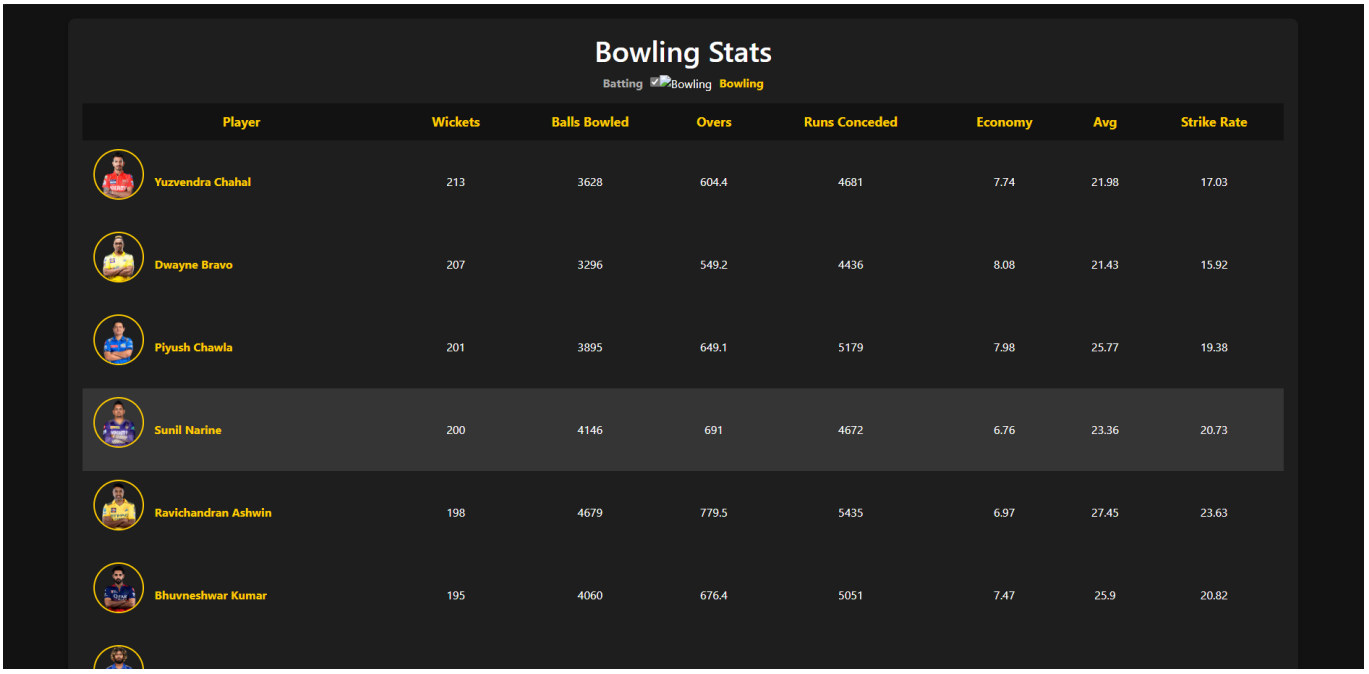
Screenshots of the Application

Screen Shot 5.2.1 Home page

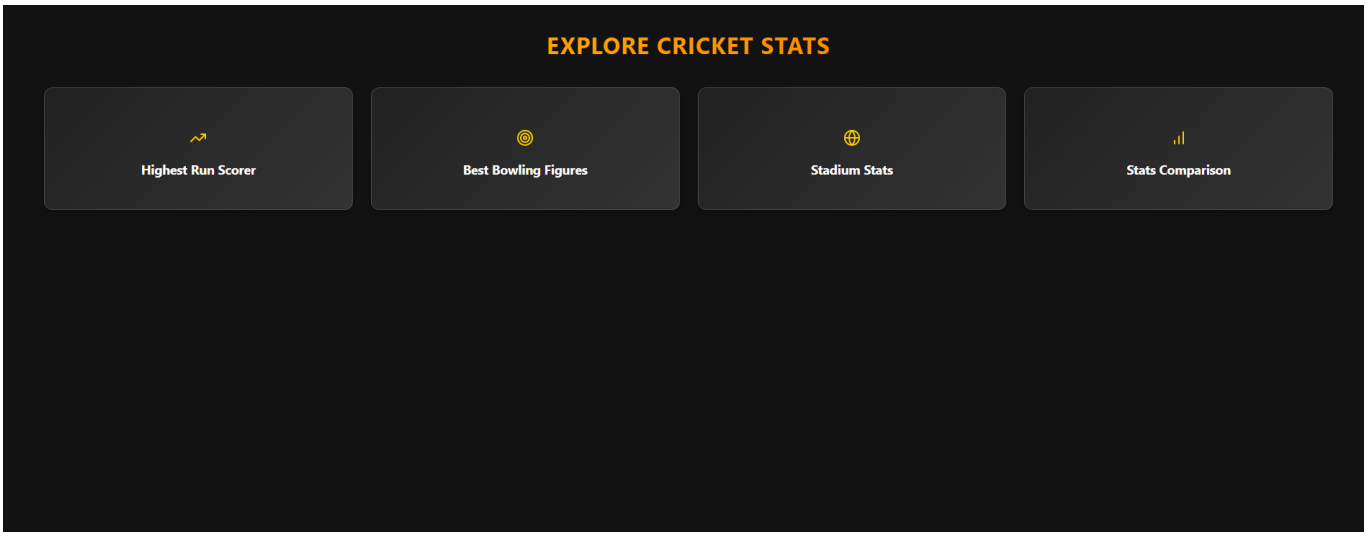
Screen Shot 5.2.2 Yearly schedule



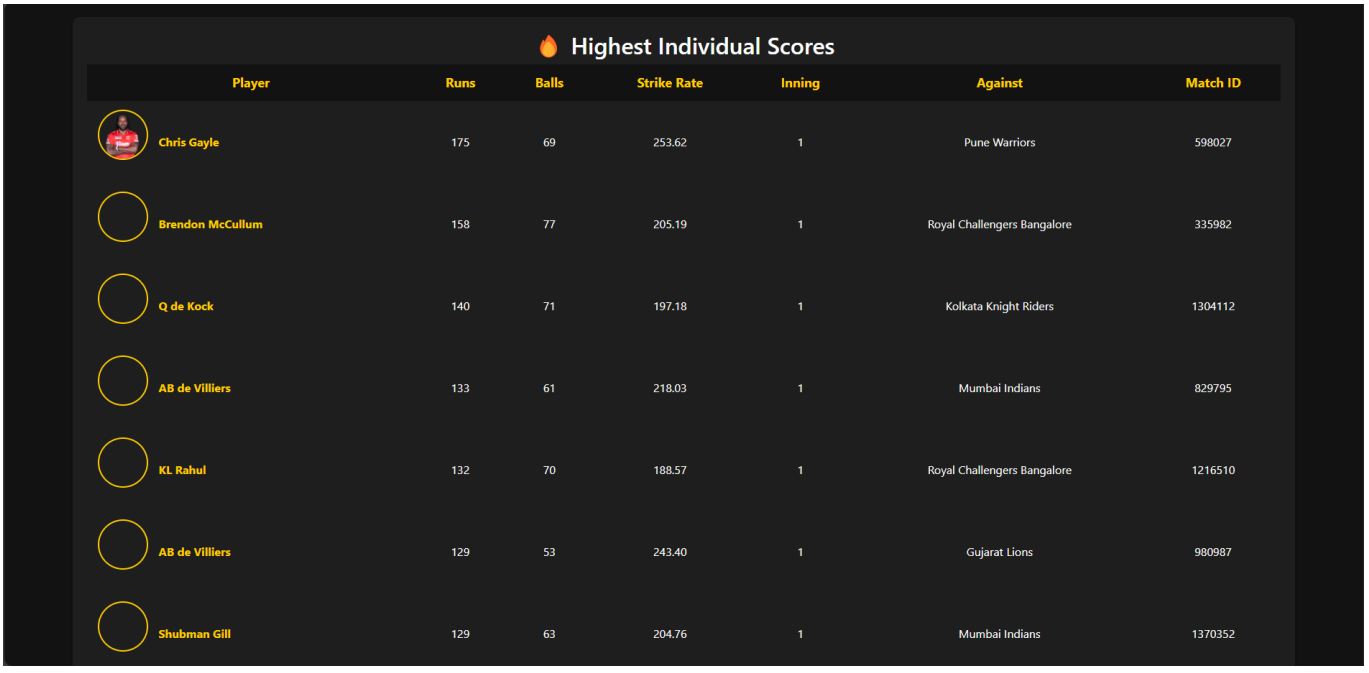
Screen Shot 5.2.3 Batting stats



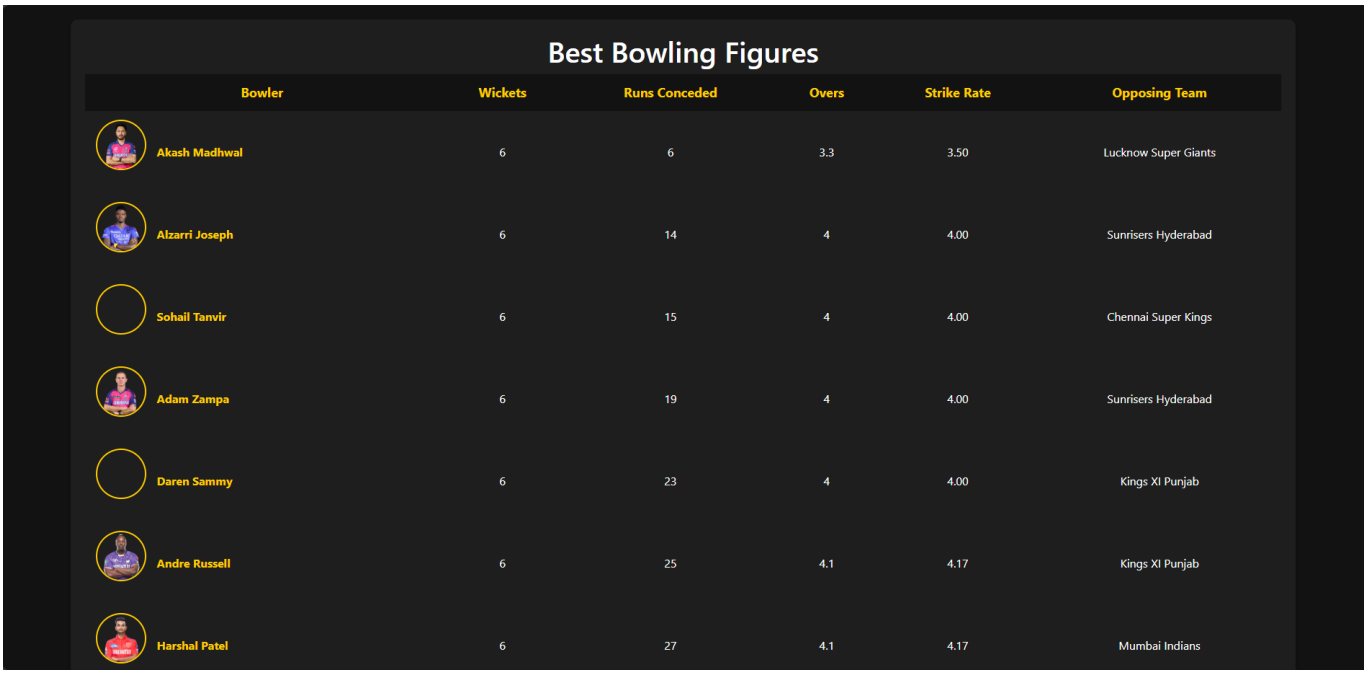
Screen Shot 5.2.4 Bowling stats



Screen Shot 5.2.5 Exploring stats



Screen Shot 5.2.6 Highest Score in an innings



Screen Shot 5.2.7 Best Bowling Figures

Screen Shot 5.2.8 Stats comparison

Screen Shot 5.2.9 Stats comparison 2

CHAPTER-6
RESULT

6.1 Result

To ensure reliable and meaningful outcomes for the CRIC DASH application, several core components were carefully implemented during the development process. These included data quality assurance, preprocessing, feature extraction, structured presentation, and thorough validation. The goal was to deliver precise and insightful cricket player statistics and comparisons that enhance the user experience.

1. **Data Preprocessing**
   Player data was thoroughly cleaned and structured for consistency. This involved handling duplicate entries, fixing invalid formats, and addressing missing values. Player names and their corresponding statistics were standardized to ensure uniformity and accurate comparison across the platform.

2. **Feature Extraction**
   Key performance features were extracted for each player, including:
   - *Batting*: Total runs, strike rate, average, boundaries
   - *Bowling*: Wickets taken, economy rate, bowling average
   - *Fielding*: Catches, run-outs

   features were formatted into structured JSON files to support seamless integration with the frontend.

3. **Data Presentation**
   The application was built to present player statistics in a clean and user-friendly format. Users can easily select and compare players, viewing their performances across multiple categories. The interface prioritizes readability and clarity to ensure information is easily digestible.

4. **Logic Implementation**
   Instead of relying on machine learning models, custom logic and functions were implemented to accurately filter and present data based on user selections. These functions ensure that the correct statistics are fetched and displayed dynamically on the frontend.

5. **Output Validation**

   The final output was validated through manual verification. Player statistics displayed on the platform were cross-checked against actual records to ensure accuracy. Additionally, the user interface was tested across components to confirm consistent and correct data rendering throughout the application.

CONCLUSION

6.2 Conclusion

CRIC DASH emerges as a robust and user-friendly cricket analytics platform that simplifies the exploration of complex cricket data for fans, analysts, and casual viewers alike. By focusing on presenting complete statistics across batting, bowling, and fielding, the application provides a detailed overview of player performance in an intuitive and visually appealing manner. The use of structured JSON data ensures that the information is accurate, well-organized, and easy to process on the frontend. One of the platform's key strengths lies in its Stats Comparison feature, which allows users to compare two players side-by-side using rich visuals like graphs, performance cards, and statistical breakdowns. This not only enhances understanding but also supports informed discussions, debates, and content creation related to matchups and cricket strategies.

From a design and development perspective, CRIC DASH is built with scalability and usability at its core. The sleek dark-themed interface, responsive layout, and smooth transitions contribute to a modern user experience that works seamlessly across devices. While the current version focuses on historical player statistics, the system's architecture is flexible enough to support future features such as live data integration, team-specific filters, venue-based insights, and season-wise breakdowns. With its data-driven foundation and engaging interface, CRIC DASH bridges the gap between raw statistics and cricket storytelling—making it not just a dashboard, but a full-fledged tool for analysis and discovery in the cricketing world.

REFERENCES

1. Williams, J. (1999). *Cricket and England: A Cultural and Social History of the Inter-War Years*. Taylor & Francis: Oxfordshire, UK.
2. Bailey, M., & Clarke, S. R. (2006). Predicting the match outcome in one day international cricket matches, while the game is in progress. *Journal of Sports Science and Medicine*, 5, 480.
3. Rehma, A. A., Awan, M. J., & Butt, I. (2018). Comparison and Evaluation of Information Retrieval Models. *VFAST Transactions on Software Engineering*, 6, 7–14.
4. Alam, T. M., & Awan, M. J. (2018). Domain analysis of information extraction techniques. *International Journal of Multidisciplinary Sciences and Engineering*, 9, 1–9.
5. Kaur, A., Kaur, R., & Jagdev, G. (2021). Analyzing and Exploring the Impact of Big Data Analytics in Sports Sector. *SN Computer Science*, 2, 1–19. https://doi.org/10.1007/s42979-020-00419-w
6. Ahmed, H. M., Awan, M. J., Khan, N. S., Yasin, A., & Shehzad, H. M. F. (2021). Sentiment Analysis of Online Food Reviews using Big Data Analytics. *Elementary Education Online*, 20, 827–836.
7. Zhou, L., Pan, S., Wang, J., & Vasilakos, A. V. (2017). Machine learning on big data: Opportunities and challenges. *Neurocomputing*, 237, 350–361. https://doi.org/10.1016/j.neucom.2017.01.026
8. Aftab, M. O., Awan, M. J., Khalid, S., Javed, R., & Shabir, H. (2021). Executing Spark BigDL for Leukemia Detection from Microscopic Images using Transfer Learning. In *Proceedings of the 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA)*, Riyadh, Saudi Arabia, 6–7 April (pp. 216–220).
9. Awan, M. J., Khan, M. A., Ansari, Z. K., Yasin, A., & Shehzad, H. M. F. (2021). Fake Profile Recognition using Big Data Analytics in Social Media Platforms. *International Journal of Computer Applications Technology*, in press.
10. Anam, M., Ponnusamy, V., Hussain, M., Waqas Nadeem, M., Javed, M., Guan Goh, H., & Qadeer, S. (2021). Osteoporosis Prediction for Trabecular Bone using Machine Learning: A Review. *Computers, Materials & Continua*, 67, 89–105. https://doi.org/10.32604/cmc.2021.013821
11. Ali, Y., Farooq, A., Alam, T. M., Farooq, M. S., Awan, M. J., & Baig, T. I. (2019). Detection of Schistosomiasis Factors Using Association Rule Mining. *IEEE Access*, 7, 186108–186114. https://doi.org/10.1109/ACCESS.2019.2960593
12. Nagi, A. T., Awan, M. J., Javed, R., & Ayesha, N. (2021). A Comparison of Two-Stage Classifier Algorithm with Ensemble Techniques on Detection of Diabetic Retinopathy. In *Proceedings of the 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA)*, Riyadh, Saudi Arabia, 6–7 April (pp. 212–215).

13. Awan, M. J., Yasin, A., Nobanee, H., Ali, A. A., Shahzad, Z., Nabeel, M., Zain, A. M., & Shahzad, H. M. F. (2021). Fake News Data Exploration and Analytics. *Electronics*, 10, 2326. https://doi.org/10.3390/electronics10192326
14. Gupta, M., Jain, R., Arora, S., Gupta, A., Javed Awan, M., Chaudhary, G., & Nobanee, H. (2021). AI-enabled COVID-19 Outbreak Analysis and Prediction: Indian States vs. Union Territories. *Computers, Materials & Continua*, 67, 933–950. https://doi.org/10.32604/cmc.2021.013174
15. de Souza Junior, A. H., Corona, F., Barreto, G. A., Miche, Y., & Lendasse, A. (2015). Minimal learning machine: A novel supervised distance-based approach for regression and classification. *Neurocomputing*, 164, 34–44. https://doi.org/10.1016/j.neucom.2014.08.095
16. Javed, R., Saba, T., Humdullah, S., Jamail, N. S. M., & Awan, M. J. (2021). An Efficient Pattern Recognition Based Method for Drug-Drug Interaction Diagnosis. In *Proceedings of the 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA)*, Riyadh, Saudi Arabia, 6–7 April (pp. 221–226).
17. Salloum, S., Dautov, R., Chen, X., Peng, P. X., & Huang, J. Z. (2016). Big data analytics on Apache Spark. *International Journal of Data Science and Analytics*, 1, 145–164. https://doi.org/10.1007/s41060-016-0027-9
18. Khalil, A., Awan, M. J., Yasin, A., Singh, V. P., & Shehzad, H. M. F. (2021). Flight Web Searches Analytics through Big Data. *International Journal of Computer Applications Technology*, in press.
19. Singh, T., Singla, V., & Bhatia, P. (2015). Score and winning prediction in cricket through data mining. In *Proceedings of the 2015 International Conference on Soft Computing Techniques and Implementations (ICSCTI)*, Faridabad, India, 8–10 October (pp. 60–66).
20. Kamble, R. (2021). Cricket Score Prediction Using Machine Learning. *Turkish Journal of Computer and Mathematics Education*, 12, 23–28.