PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100-ft Ring Road, Bengaluru – 560 085, Karnataka, India

Project Report (Phase-1)
On

CricketVerse

Submitted by

Md Asraf Ali - (PES1PG23CA080)

Nov 2024 – Feb 2025

under the guidance of

Guide Details

Mr. Tamal Dey
Assistant Professor
Department of Computer Applications,
PESU, Bengaluru – 560085

FACULTY OF ENGINEERING
DEPARTMENT OF COMPUTER APPLICATIONS
PROGRAM – MASTER OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that the project entitled

CricketVerse

is a bonafide work carried out by

Md Asraf Ali – PES1PG23CA080

in partial fulfillment for the completion of Capstone Project, Phase-1 work in the Program of Study MCA under rules and regulations of PES University, Bengaluru during the period Oct. 2024 – Jan 2024. The project report has been approved as it satisfies the academic requirements of 3rd semester MCA.

Signature with date
Internal Guide
Mr. Tamal Dey
Asst. Prof.
Department of Computer Applications, PES University, Bengaluru - 560085

Signature with date & Seal
Chairperson
Dr. Veena S
Department of Computer Applications,
PES University, Bengaluru - 560085

DECLARATION

I, **Md Asraf Ali,** bearing **PES1PG23CA080** hereby declare that the Capstone project phase-1 entitled, ***CricketVerse,*** is an original work done by me under the guidance of **Mr. Tamal Dey,** Assistant Professor, PES University, and is being submitted in partial fulfillment of the requirements for completion of 3rd Semester course in the Program of Study **MCA**. All corrections/suggestions indicated for internal assessment have been incorporated in the report.

PLACE: Bengaluru

DATE:

Md Asraf Ali

Abstract

This project aims to provide the analytical experience of T20 cricket by providing live scores, real-time statistics, and AI-powered insights to fans, analysts, and enthusiasts. Through the merging of live match information with changing performance indicators, it raises interest and comprehension of the game. Pre-match analysis provides strategic advice on whether to bat or bowl first, using past venue patterns, weather conditions, and historical performance trends. In addition, a machine learning model forecasts the chances of a successful chase in the second innings, updating win chances dynamically after each ball based on live match information. By marrying AI-powered analytics with thorough match tracking, the platform not only enhances the fan experience but also revolutionizes cricket analysis, giving richer, more predictive insights into match dynamics.

Table of Contents

Chapter 1

Introduction

1.1 Project Description

This project aims to take the analytical experience of ongoing cricket matches to the next level by providing real-time scores of all ongoing matches, dynamic statistics, and insights for enthusiasts, analysts, and fans. Through the display of live match updates together with changing metrics, the platform improves engagement of the game. Pre-match information provides strategic advice, e.g., whether to bat or bowl first, from historical venue statistics, weather conditions, and past performance records. The information seeks to provide teams and supporters with a better, data-driven understanding prior to the first ball being bowled.

Apart from pre-match analysis, the project proposes a machine learning model for predicting the chances of a successful chase in the second innings of a T20 match. The model updates continuously after each ball, using current data to modify win possibilities based on the match status. Through the use of Machine Learning, the platform not only enhances the viewing experience but also offers analysts more insightful, predictive understandings of matches, and revolutionizes the way T20 cricket is watched and comprehended.

1.2 Problem Definition

Cricket teams often face challenges in deciding whether to bat or field after winning the toss, as the decision depends on multiple factors such as ground conditions, weather, and historical performances. Whereas Fans and teams lack accurate, real-time predictions of match outcomes based on evolving game scenarios and player/team performance metrics.

1.3 Proposed Solution

A Toss Decision Support Tool analyses pitch and weather conditions to determine which team should bat first while also recommending when to bowl. The tool indicates that bowling becomes favourable under overcast conditions because swing improves thus enabling teams to leverage data-based information. A Real-Time Match Outcome Predictor uses machine learning models to predict game outcomes based on live game data combined with historical and performance records and current match evaluations. Models built using Random Forest Classifier achieve high accuracy in match prediction according to research which makes them essential resources for teams together with their fans.

1.4 Purpose

The purpose of this project is to build a real-time cricket scoring and prediction platform that keeps fans engaged with instant match updates and AI-powered win probability predictions. Fans can track live scores, follow dynamic match insights, and enjoy a more interactive viewing experience. Analysts benefit from data-driven performance metrics.

1.5 Scope

While this project provides a robust solution for predicting and analysing T20 cricket matches, it does have some limitations. Currently, predictions focus only on the second innings, assessing the chasing team's probability based on factors like required run rate, wickets in hand, and game dynamics. Additionally, toss recommendations rely on historical patterns, which might not always reflect sudden environmental changes. Expanding the model to include first-innings predictions and adaptive factors could further improve its accuracy and usefulness.

Chapter 2

Literature Survey

2.1 Domain Study

CricketVerse requires domain study through a thorough analysis of cricket analytics combined with real-time match prediction platforms as well as solution examination and user need gap determination. Students must conduct analysis of present cricket prediction model accuracy and modelling effectiveness while following AI sports analytics developments as well as market trend monitoring. This section examines technologies that include machine learning along with web scraping tools for live scores and large-scale data handling through cloud computing. The research presents multiple sports industry participation models as well as business strategies and potential business relationships in sports-related ventures. The strategic purpose is to discover creative solutions for differentiation alongside providing analytics-driven services that satisfy cricket fans and their teams and analysis experts.

2.2 Related Work

This involves the study of research papers and journals. Literature survey is completed by considering following research papers.

1. Winner Prediction in One-Day International Cricket Matches Using Machine Learning Framework: An Ensemble Approach
   Author :- Manoj Ishi, Dr. Jayantrao Patil et al.
   Publication :- Indian Journal of Computer Science and Engineering | e-ISSN: 0976-5166 | Volume:13/Issue:03/June-2022
   Summary :- This research aims to develop machine learning models to predict the winner of one-day international (ODI) cricket matches before the game begins, providing insights for team management and sports analysts. Using historical data from 1693 ODI matches (2006–2019), the study evaluates batting and bowling strength, run-scoring patterns, and overall team metrics to build predictive models. Various machine learning algorithms, including Logistic Regression, SVM, and ensemble methods like voting and stacking classifiers, were employed, with feature selection techniques enhancing accuracy. The best models achieved up to 96.31% accuracy, demonstrating the potential of machine learning to handle cricket's unpredictability and offering practical value for strategy optimization. Future work includes incorporating additional features and extending the methodology to other cricket formats or sports analytics domains.

2. Prediction of IPL Match Outcome Using Machine Learning Techniques
   Author :- Srikantaiah K C, Aryan Khetan et al.
   Publication :- 3rd International Conference on Integrated Intelligent Computing, Communication & Security (2019)
   Summary :- This research focuses on predicting the outcomes of Indian Premier League (IPL) matches using machine learning models based on nine years of historical data. By analysing datasets that include team performance (home/away), match details, player statistics, and ball-by-ball deliveries, features such as win percentages, toss impacts, and player metrics were extracted. Machine learning models like Random Forest, SVM, Logistic Regression, and K-Nearest Neighbour were implemented, with Random Forest achieving the highest accuracy of 88.10%. The models were trained on 70% of the data and tested on 30%, with cross-validation ensuring reliability. This study offers valuable insights for team managers, analysts, and fans, enabling informed decisions about match strategies and outcomes. Future work includes evaluating individual player performances and integrating additional features to improve prediction accuracy and versatility.

3. Cricket Match Analytics and Prediction Using Machine Learning
   Author :- Param Dalal, Hirak Shah, Tej Kanjariya, Dhananjay Joshi
   Publication :- International Journal of Computer Applications (2006) | ISSN: 0975-8887 | Volume 186/Issue:26/June-2024
   Summary :- This research explores cricket match prediction using machine learning, focusing on outcomes during the second innings by analysing factors like target, runs left, wickets fallen, and player-specific metrics. A key innovation is a custom "Player Consistency" formula, which combines traditional cricket statistics with dynamic ratings to enhance predictive accuracy. The study evaluates models such as Random Forest, SVM, Logistic Regression, and Naive Bayes, with Random Forest achieving the highest testing accuracy of 89.82%. By leveraging techniques like feature extraction and deep learning for weight optimization, the research demonstrates significant advancements in cricket analytics, providing actionable insights for players, teams, and fans while identifying gaps and opportunities for future enhancements in prediction models.

4. Predicting IPL Victories: An Ensemble Modelling Approach Using Comprehensive Dataset Analysis
   Author :- Pritpal Singh, Dr. Jatinder Kaur, Lovedeep Singh
   Publication :- 2nd International Conference on Artificial Intelligence and Machine Learning Applications Theme: Healthcare and Internet of Things (2024) | ISSN: 3503-4922 | Issue: 1, 24
   Summary :- This research applies machine learning techniques to predict IPL match outcomes using factors like player statistics, team performance, venue, and weather. Algorithms such as Decision Trees and Random Forest were tested, with the ensemble model achieving **99% accuracy**, outperforming individual models. The study demonstrates the potential of ensemble approaches in improving prediction accuracy and suggests integrating more contextual features for enhanced performance in future applications.

5. Outcome Prediction of ODI Cricket Matches Using Decision Trees and MLP Networks
   Author :- Jalaz Kumar, Rajeev Kumar, Pushpender Kumar
   Publication :- First International Conference on Secure Cyber Computing and Communication (2018)
   Summary :- This research uses machine learning models like Decision Trees and Multilayer Perceptron (MLP) Networks to predict outcomes of ODI cricket matches based on factors such as past team performance, venue, innings order, and home advantage. Using data from 3933 ODI matches, the study trained models on pre-game features to develop a prediction tool named CricAI, which forecasts match results with reasonable accuracy. MLP Classifier achieved an accuracy of 57.4%, slightly outperforming Decision Trees at 55.1%. The findings highlight the effectiveness of these classifiers in capturing patterns from historical data and suggest potential improvements through team composition analysis and application to other sports.

2.3 Existing System

Features	CricViz	ESPNcricInfo	Smartcric	CricketVerse
Live Score and Match Details	Y	Y	Y	Y
Toss Decision Prediction	N	N	N	Y
Second Innings Ball-by-Ball Victor Prediction	Y	Y	N	Y
Venue-Specific Insights	N	Y	N	Y
Prediction for all formats	Y	N	N	N

Figure 2.1: Comparison Table

2.4 Technology Survey

This technology survey includes following technologies :-

2.4.1 NodeJs :-

NodeJS is an open-source, cross-platform tool, first developed in 2009 by Ryan Dahl, which basically creates a runtime environment for JavaScript to function outside the browser environment. It

utilizes asynchronous programming. Built on the JavaScript V8 engine, it can run in various browsers including Chrome or even function as a standalone tool. NodeJS can handle server requests seamlessly.

Basically, NodeJS widens the scope of JavaScript's functionality. NodeJS helps to integrate coding languages with APIs, other languages, and several external libraries. It plays an exclusive role in web app development using the 'JavaScript everywhere' paradigm and can handle both server-side scripting and client-side programming.

2.4.2 ReactJs :-

ReactJS is a simple, feature rich, declarative, efficient and flexible open-source front-end JavaScript library for building reusable UI components. It is an open-source, component-based frontend library responsible only for the view layer of the application. It was created by Jordan Walke, who was a software engineer at Facebook. It was initially developed and maintained by Facebook and was later used in its products like WhatsApp & Instagram. Facebook developed ReactJS in 2011 in its newsfeed section, but it was released to the public in the month of May 2013.

React creates a VIRTUAL DOM in memory. Instead of manipulating the browser's DOM directly, react creates a virtual DOM in memory, where it does all the necessary manipulating, before making the changes in the browser DOM. React only changes what needs to be changed. React finds out what changes have been made, and changes only what needs to be changed.

Today, most of the websites are built using MVC (model view controller) architecture. In MVC architecture, React is the 'V' which stands for view, whereas the architecture is provided by the Redux or Flux.

2.4.3 CSS :-

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speechbased browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

In addition to HTML, other markup languages support the use of CSS including XHTML, plain XML, SVG, and XUL.

2.4.4 JavaScript :-

JavaScript often abbreviated JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. Over 97% of websites use JavaScript on the client side for web page behavior, often incorporating third-party libraries.

All major web browsers have a dedicated JavaScript engine to execute the code on user's devices. JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

The ECMAScript standard does not include any input/output (I/O), such as networking, storage, or graphics facilities. In practice, the web browser or other runtime system provides JavaScript APIs for I/O.

JavaScript engines were originally used only in web browsers, but are now core components of some servers and a variety of applications. The most popular runtime system for this usage is Node.js.

Although Java and JavaScript are similar in name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design.

2.4.5 MongoDB :-

MongoDB is a cross-platform, document-oriented database that provides, high performance, easy scalability and leading NoSQL database. MongoDB is written in C++. MongoDB is a scalable, flexible NoSQL document database platform designed to overcome the relational databases approach and the limitations of other NoSQL solutions. MongoDB is well known for its horizontal scaling and load balancing capabilities, which has given application developers an unprecedented level of flexibility and scalability. It is categorized under the NoSQL (Not only SQL) database because the storage and retrieval of data in the MongoDB are not in the form of tables.

MongoDB works on concept of collection and document. Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose. A document is a set of keyvalue pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

Chapter 3

Hardware and Software Requirements

3.1 Hardware Requirements

Hardware Specification	
Specification	**Desired Value**
Processor	Intel i3 and above
Memory (RAM)	8GB Minimum
Hard Disk	10GB Minimum

3.2 Software Requirements

Hardware Specification	
Specification	**Desired Value**
Operating System	Web OS/ Mobile OS
Language	Python 3.7, JavaScript
Front end Tool	ReactJS (v18.2.0 or higher)
Backend Tool	NodeJS (v18.16.0 or higher)
Database	MongoDB (v5.0.2 or higher)
Web Server	ExpressJs (v4.18.2 or higher)
Web Browser	Any
Development Tool	Visual Studio Code
API Testing Tool	Postman

Library	NumPy 1.14.3, Matplotlib 2.1.2 Scikit-learn, Pandas, Seaborn, requests, BeautifulSoup

Chapter 4

Software Requirement Specification

4.1 System Users

- There are following type of system users :-

1. **User(Fans/Cricket Analyst)** : Accesses live match updates, toss decision, prediction insights, player statistics and match result statistics.

4.2 Functional Requirements

- There are following functional requirements :-

1. Live Score and Statistics Display

The users of CricketVerse experience real-time scores of active matches with both detailed ball-by-ball presentations and essential player data statistics. The feature presents vital performance numbers which combine run rates with necessary run rates and projected scores to enable fans and analysts to observe real-time game progression. The user-friendly interface design maintains ongoing match connectivity which leads to better viewing satisfaction.

2. Second Inning Win Prediction

The CricketVerse machine learning model evaluates in real-time how likely a chasing team will win their target score through running predictions of game progress in the second inning. The system recalculates predictions after each bowl to evaluate vital variables which include the number of wickets remaining and the required run speed and individual playing statistics. Horizon Match Preview provides reliable data-driven analysis of an ongoing match to fans and analysts so they can understand the match situation better.

3. Toss Decision Recommendation

Following the toss victory CricketVerse generates strategic advice by using historical and venue-related database information in combination with environmental aspects for team decision-making regarding first batting or bowling choice. The tool generates specific recommendations for teams which help them optimize performance under current real-time situation parameters.

4. Generate Match Result Statistics

CricketVerse runs automatic match result statistics calculations after every match finishes. Downloadable results through CricketVerse include key performance metrics, player statistics together with match recaps to give users complete game information. Users can use this functionality to examine final matches so they gain performance insights about teams and players which assists their preparation for upcoming games.

4.3 Non-Functional Requirements

- There are following functional requirements :-

1. Scalability

CricketVerse implements a robust architecture designed to handle concurrent user access efficiently, particularly during peak match times. Through advanced load balancing techniques and distributed computing infrastructure, the platform seamlessly manages multiple simultaneous connections, ensuring smooth delivery of live updates and real-time predictions without performance degradation, even during high-traffic cricket events.

2. Performance

CricketVerse prioritize performance optimization to deliver fast loading times and responsiveness across the platform. Through efficient code optimization, we enhance the platform's efficiency and usability, providing users with a seamless and enjoyable experience.

3. Usability and Accessibility

CricketVerse offers sophisticated cricket analytics in forms that are simple to understand thanks to its user-friendly interface. Regardless of technical expertise, customers can easily access match statistics and forecast insights thanks to carefully crafted dashboards and interactive visualizations. Fans of all skill levels can access extensive cricket analytics thanks to the platform's adherence to contemporary design concepts and user experience criteria.

4. Reliability

CricketVerse maintains exceptional system availability through comprehensive fault-tolerance mechanisms and redundant server architecture. The platform employs sophisticated monitoring systems and automated failover protocols to guarantee uninterrupted service delivery, particularly during crucial match moments. Through proactive system health checks and preventive maintenance, we minimize potential disruptions and maintain consistent platform performance.

Chapter 5

System Design

5.1 Architecture Diagram

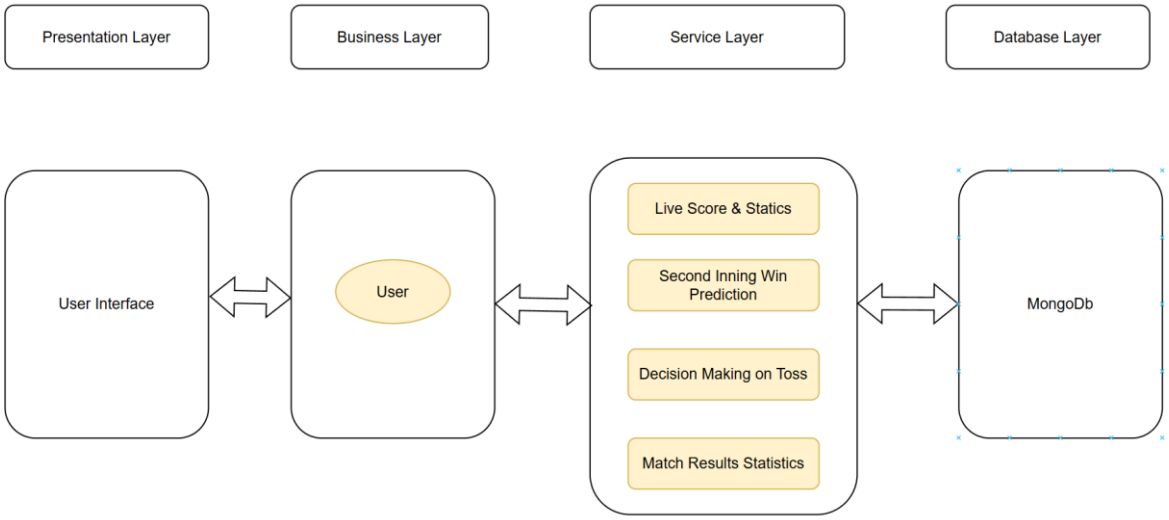

Figure 5.1: Architecture Diagram

The architecture diagram of our system comprises several distinct layers, including the Presentation Layer, Business Layer, Service Layer, and Database Layer, are depicted in the following image, which shows a layered architectural diagram that shows the layout of a software program. The database technology utilized is MongoDB. The User Interface, which enables communication with the User, is fundamental to the diagram. The Business Layer is in charge of processing essential features including match results data, second inning win prediction, live score and statistics, and toss decision-making. In the context of athletic events, these components highlight the application's function in managing real-time data and statistics. The two-way arrows represent the data flow between the User, the Service Layer, and the Database, emphasizing the dynamic information exchange necessary to provide the end-user with current and pertinent content. All things considered, the design provides a smooth user experience by clearly illustrating how data is handled and processed within the application's ecosystem.

5.2 Process Flow Diagram

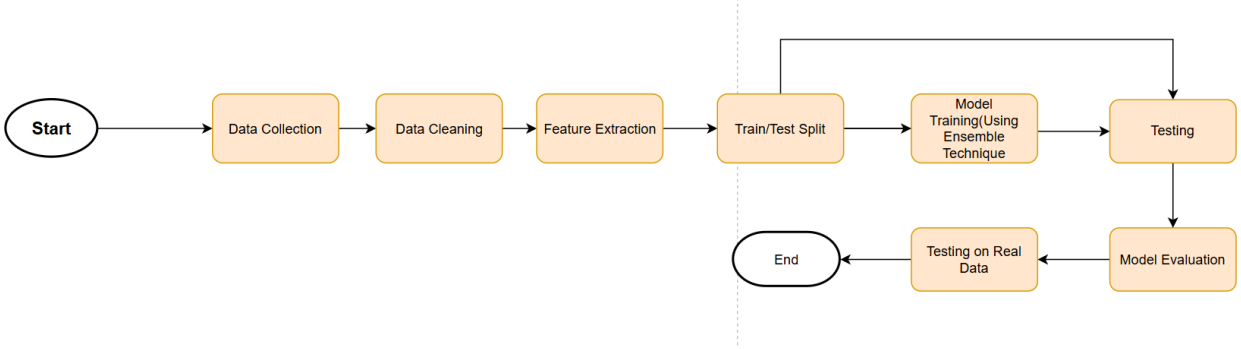

Figure 5.2: Process Flow Diagram

The process flow diagram for CricketVerse's Machine Learning Model Development depicts the methodical methodology utilized to create and evaluate the model.

1. Start – The procedure begins with data collecting.

2. Data Collection — Relevant cricket match data is collected from multiple sources, including live match statistics, historical records, and player performance measures.

3. Data Cleaning – The collected data undergoes preprocessing, including handling missing values, removing inconsistencies, generating required columns out of existing one's, and formatting for consistency.

4. Feature Extraction – Key features like as batting performance, bowling impact, and run rates are extracted to boost model accuracy.

5. Train/Test Split – The dataset is divided into training and testing subsets to ensure the model is trained effectively and evaluated properly.

6. Model Training Using Ensemble Techniques – The model is trained using an ensemble approach, incorporating multiple algorithms to improve predictive accuracy.

7. Testing — The trained model is tested with unseen data to measure its performance.

8. Model Evaluation – Key evaluation metrics such as accuracy, precision, recall, and F1-score are analysed to determine the model's effectiveness.

9. Testing on Real Data — The model is further verified using real-world match scenarios to assess its predicted dependability.

10. End – The procedure closes, guaranteeing that the model is ready for deployment in CricketVerse

for real-time match predictions and analytics.

This procedure ensures a disciplined and efficient approach to developing a viable prediction model for CricketVerse.

Chapter 6

Detailed Design

6.1 Use Case Diagram

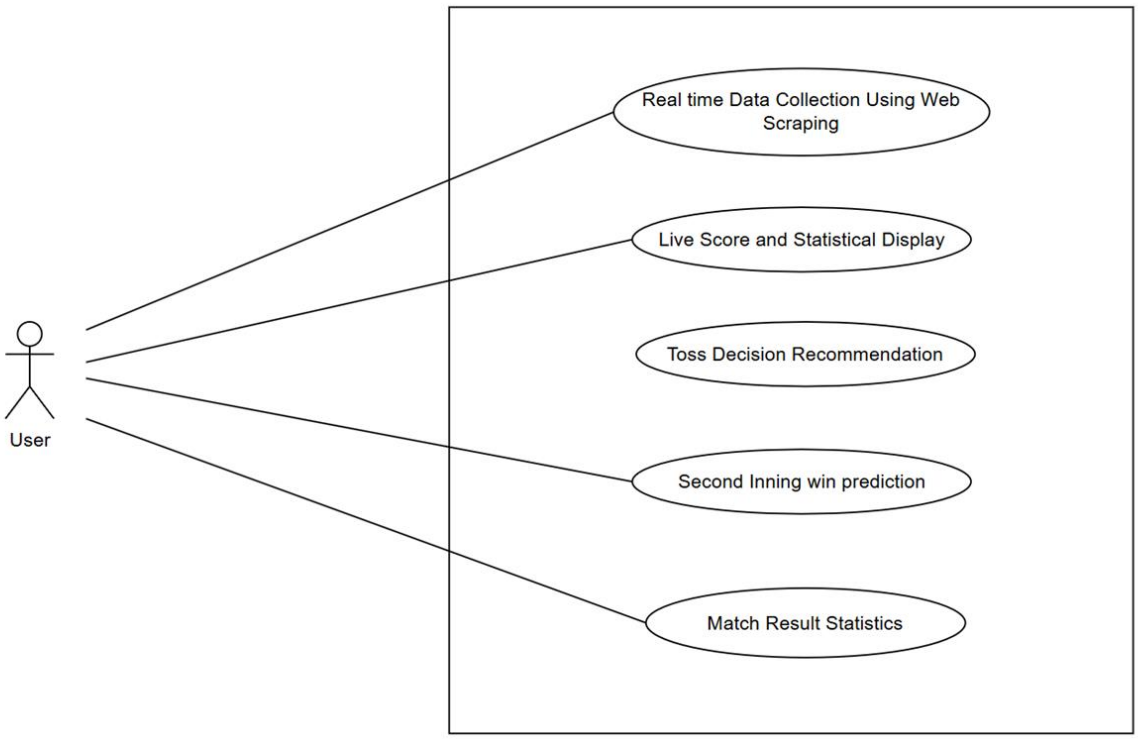

Figure 6.1: Use Case Diagram

The User is the main actor in the use case diagram (Figure 6.1) for CricketVerse, interacting with a number of important system elements. The platform may retrieve and show real-time match updates by allowing the user to participate in Real-Time Data Collection Using Web Scraping.

Additionally, the customer has access to tools like Live Score and Statistical Display, which allows them to view performance metrics, forecasted scores, and ball-by-ball scores.

Based on past results and ground circumstances, the Toss Decision Recommendation feature gives users tactical advice on whether a team should bat or bowl first. The Second Inning Win Prediction also helps the user by predicting the probability that a chasing team will reach the target score based on real-time data that takes into account player performance, needed run rate, and wickets. Lastly, customers can examine Match Result Statistics, which offers information on important performance metrics and completed match results.

This use case graphic illustrates how the user interacts with the different features and how the platform uses data-driven insights to improve the cricket viewing experience.

6.2 Database Design

Document Structure

Live Match Data

```
{
  status: String,
  team1: String,
  team2: String,
  score1: String,
  score2: String,
  match_result: String,
  createdAt: { type: Date, default: Date.now },
}
```


```
{
  "_id": {
    "$oid": "67add169a34edb0cbfc85673"
  },
  "status": "RESULT",
  "team1": "IND",
  "team2": "ENG",
  "score1": "356",
  "score2": "(34.2/50 ov, T:357) 214",
  "match_result": "India won by 142 runs",
  "createdAt": {
    "$date": "2025-02-13T11:03:05.933Z"
  },
  "__v": 0
}
```


Figure 6.2: Live Match Data Document Structure

Chapter 7 Implementation

7.1 Screen Shot

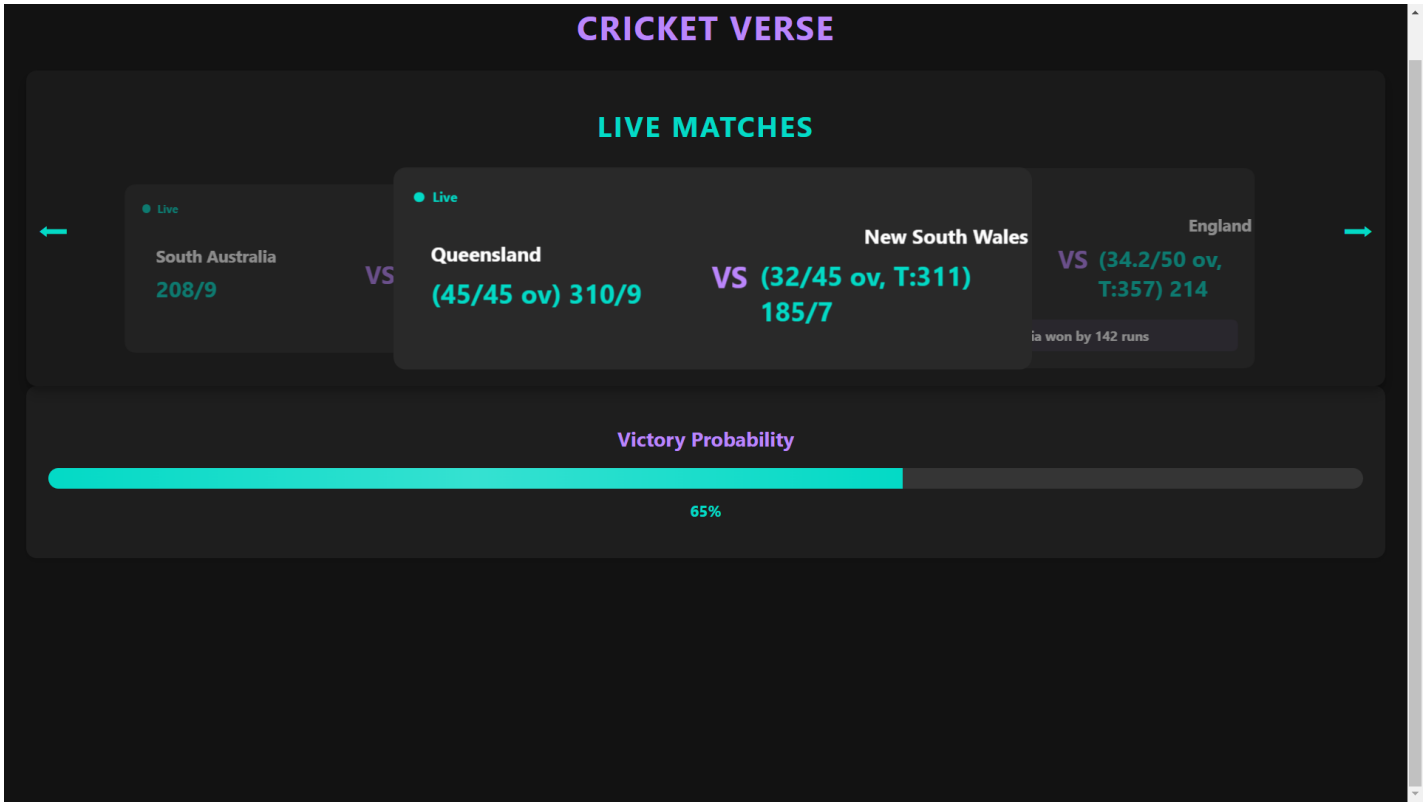

Figure 7.1: Landing Page

When users first search for CricketVerse, they are directed to our landing page. Here, they find a simple yet informative layout designed with ReactJS. The image represents the CricketVerse live score interface. The interface has a dark-themed UI with purple and cyan accents. At the top, "CRICKET VERSE" is prominently displayed, followed by the section titled LIVE MATCHES, which showcases real-time scores of ongoing games. The current display features three matches, with a central focus on the Queensland vs. New South Wales match. The match details include the total runs, wickets, overs played, and the target for the chasing team. The other two matches are partially visible on either side, with navigation arrows suggesting the ability to scroll through additional games.

Below the live match cards, a Victory Probability bar is displayed, indicating a 65% probability. This bar is currently static as the AI-driven prediction model is yet to be implemented. The bar visually represents the likelihood of a team winning based on real-time match conditions.

	Unnamed: 0	Match ID	Date	Venue	Bat First	Bat Second	Innings	Over	Ball	Batter	...	Winner	Chased Successfully	Total Batter Runs	Total Non Striker Runs
0	0	1339605	2023-03-26	SuperSport Park	West Indies	South Africa	1	1	1	BA King	...	South Africa	1	1	0
1	1	1339605	2023-03-26	SuperSport Park	West Indies	South Africa	1	1	2	KR Mayers	...	South Africa	1	1	1
2	2	1339605	2023-03-26	SuperSport Park	West Indies	South Africa	1	1	3	BA King	...	South Africa	1	0	1
3	3	1339605	2023-03-26	SuperSport Park	West Indies	South Africa	1	1	4	J Charles	...	South Africa	1	0	1
4	4	1339605	2023-03-26	SuperSport Park	West Indies	South Africa	1	1	5	J Charles	...	South Africa	1	4	1

5 rows × 35 columns

Figure 7.2: Data Set Header

The Figure 7.2 displays the dataset which provides the column names and the data, in Figure 7.2 only 5 Rows has been displayed just for the analysis purpose. The dataset has 35 columns as shown in Figure 7.4 but there are only few columns displaying in the Figure, this is because there is not much space to accommodate that much columns into it, the columns which are not being displayed are as follows Non Striker, Bowler, Batter Runs, Extra Runs, Runs From Ball, Ball Rebowled, Extra Type, WicketMethod, Player Out, Innings Runs, Innings Wickets, Target Score, Runs to Get, Balls Remaining, Batter Balls Faced, Non Striker Balls Faced, Player Out Runs, Player Out Balls Faced, Bowler Runs Conceded, Valid Ball.

```
second_inning_data = data[data['Innings'] == 2]
second_inning_data = second_inning_data.drop(columns=['Date'], axis = 1)

# List of teams to keep
teams_to_keep = ["India", "Australia", "England","Pakistan","West Indies","South Africa","Sri Lanka","Bangladesh"]
# Replace with your desired teams

# Filter the dataset
filtered_data = second_inning_data[
    (second_inning_data['Bat First'].isin(teams_to_keep)) &
    (second_inning_data['Bat Second'].isin(teams_to_keep))
]

# Save the filtered dataset
filtered_file_path = 'second_inning_data.csv'
filtered_data.to_csv(filtered_file_path, index=False)

print(f"Filtered dataset saved to: {filtered_file_path}")
```

```
Filtered dataset saved to: second_inning_data.csv
```

Figure 7.3: Data Filter

The Figure 7.3 depicts filtration of desired international teams for the further process. Because keeping all the teams could affect the accuracy of the model, other teams don't have much historical data to work on which can lead to inaccuracy.

```
# Calculate additional features
second_inning_data['current_run_rate'] = second_inning_data['Innings Runs'] / second_inning_data['Over']
second_inning_data['required_run_rate'] = (second_inning_data['Target Score'] - second_inning_data['Innings Runs']) / (20 - second_inning_data['Over'])
second_inning_data['wickets_remaining'] = 10 - second_inning_data['Innings Wickets']

# Verify the new features
print(second_inning_data[['current_run_rate', 'required_run_rate', 'wickets_remaining']].head())
```

```
     current_run_rate  required_run_rate  wickets_remaining
124               4.0          13.421053                 10
125               8.0          13.210526                 10
126               8.0          13.210526                 10
127               8.0          13.210526                 10
128               9.0          13.157895                 10
```

Figure 7.4: Generating Columns

The Figure 7.4 displays creation of new features which is required for the prediction of CricketVerse. Using existing columns from the dataset generating new columns into the dataset and saving this new dataset into another XML file.

```python
from sklearn.preprocessing import LabelEncoder

# Initialize label encoder
label_encoder = LabelEncoder()

# Encode categorical columns
for col in categorical_columns:
    second_inning_data[col] = label_encoder.fit_transform(second_inning_data[col])

# transformed data
print(second_inning_data.head())
```

```
     Unnamed: 0  Match ID  Venue  Bat First  Bat Second  Innings  Over  Ball  \
124         124   1339605    151         94          79        2     1     1
125         125   1339605    151         94          79        2     1     2
126         126   1339605    151         94          79        2     1     3
127         127   1339605    151         94          79        2     1     4
128         128   1339605    151         94          79        2     1     5

     Batter  Non Striker  ...  Total Non Striker Runs  Batter Balls Faced  \
124    1736         1793  ...                       0                   1
```

Figure 7.5: Label Encoding

The Figure 7.5 depicts label encoding where the categorical columns assigned a unique integer to each category in a column so that algorithm can process them efficiently. It is commonly used in preprocessing categorical variables for ML models.

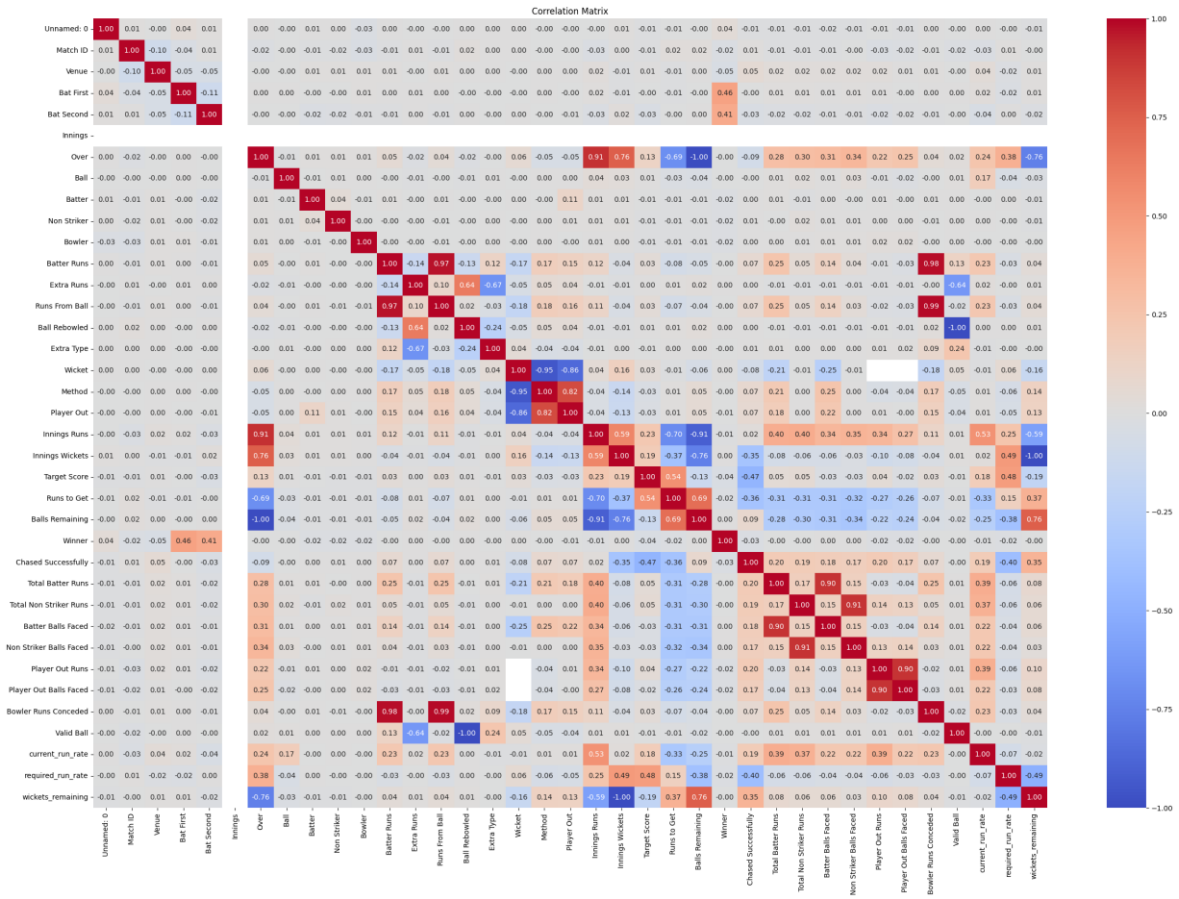

Figure 7.6: Corelation Table

The correlation matrix displayed in the Figure 7.6 which provides a comprehensive analysis of the relationships between various numerical features in the cricket match dataset. Each value in the matrix represents the correlation coefficient between two variables, ranging from -1 to +1. A positive correlation (depicted in red shades) indicates that both variables increase together, whereas a negative correlation (shown in blue shades) suggests that as one variable increases, the other decreases. Neutral or greyish shades signify little to no correlation between the variables. From the matrix, several key insights can be observed. Strong positive correlations include "Innings Runs" and "Total Batter Runs" (0.91), highlighting that higher total runs in an innings are directly linked to individual batter performances. Similarly, "Balls Remaining" and "Runs to Get" (-0.79) exhibit a strong negative correlation, signifying that as more balls remain, the required runs are likely to be higher, particularly in the early phases of an inning. Another important observation is the relationship between "Current Run Rate" and "Valid Ball" (0.59), which suggests that a higher number of valid deliveries positively impacts the scoring rate. Additionally, notable negative correlations include "Wickets Remaining" and "Target Score" (-0.76), indicating that as the target score increases, fewer wickets may be left, reflecting the pressure on the batting side. The inverse relationship between "Runs to Get" and "Current Run Rate" (-0.69) implies that a high required run rate often corresponds with a struggling batting side. Moderately correlated variables, such as "Winner" and "Chased Successfully" (0.46), indicate that successful chases have a reasonable impact on match outcomes. Similarly, "Total Batter Runs" and "Batter Balls Faced" (0.28) suggest that facing more balls typically results in higher individual scores, though not always with a very strong relationship. The correlation matrix plays a crucial role in feature selection for predictive

modelling. It helps identify highly correlated variables, which might introduce redundancy and multicollinearity in machine learning

models. Understanding these correlations allows for better feature engineering, ensuring that only the most relevant variables are used for predicting match outcomes. In particular, features such as "Current Run Rate" and "Required Run Rate" demonstrate significant relationships with match-winning factors, making them strong candidates for predictive analysis in cricket analytics.

```
while True:
    response = fetch_with_retry(url, headers)

    if response:
        soup = BeautifulSoup(response.content, "html.parser")
        matches = soup.find_all("div", class_="ds-px-4 ds-py-3")  # Find all match

        scraped_data = []  # List to store match data

        for match in matches[:3]:  # Limit to first 3 matches
            try:
                # Extract status
                status = match.find("span", class_="
ds-text-tight-xs ds-font-bold ds-uppercase ds-leading-5")
                status_text = status.text.strip() if status else "Unknown"

                # Extract team names
                teams = match.find_all("div", class_="
ds-flex ds-items-center ds-min-w-0 ds-mr-1")
                team1 = teams[0].find("p").text.strip() if len(teams) > 0 and teams[0
].find("p") else "N/A"
                team2 = teams[1].text.strip() if len(teams) > 1 else "N/A"

                # Extract scores
                scores = match.find_all("div", class_="
ds-text-compact-s ds-text-typo ds-text-right ds-whitespace-nowrap")
                score1 = scores[0].text.strip() if len(scores) > 0 else "N/A"
                score2 = scores[1].text.strip() if len(scores) > 1 else "Yet to bat"

                # Extract match result
                match_result = match.find("p", class_="
ds-text-tight-s ds-font-medium ds-truncate ds-text-typo")
                match_result_text = match_result.text.strip() if match_result else "
In Progress"

                # Store data in dictionary
                data = {
                    "status": status_text,
                    "team1": team1,
                    "team2": team2,
                    "score1": score1,
                    "score2": score2,
                    "match_result": match_result_text
                }

                scraped_data.append(data)
```

Figure 7.6: Scraping

```
app.post("/save-data", async (req, res) => {
  try {
    if (!Array.isArray(req.body)) {
      return res.status(400).send("Invalid data format. Expected an array.");
    }

    await MatchData.insertMany(req.body); // ✅ Save all matches at once
    console.log("✅ Data saved successfully to MongoDB.");
    res.send("✅ Data saved successfully.");
  } catch (error) {
    console.error("❌ Error saving data:", error);
    res.status(500).send("❌ Failed to save data.");
  }
});

// Get last 3 matches from MongoDB
app.get("/get-data", async (req, res) => {
  try {
    const matches = await MatchData.find().sort({ createdAt: -1 }).limit(3);
// Latest 3 matches
    res.json(matches);
  } catch (error) {
    console.error("Error fetching data:", error);
    res.status(500).send("Failed to fetch data.");
  }
});
```

Figure 7.6: Backend

Appendix A
BIBLIOGRAPHY

[1] Winner Prediction in One-Day International Cricket Matches Using Machine Learning Framework: An Ensemble Approach | Indian Journal of Computer Science and Engineering | e-ISSN: 0976-5166 | Volume:13 | Issue:03 | June-2022

[2] Prediction of IPL Match Outcome Using Machine Learning Techniques | 3rd International Conference on Integrated Intelligent Computing, Communication & Security (2019)

[3] Cricket Match Analytics and Prediction Using Machine Learning | International Journal of Computer Applications (2006) | ISSN: 0975-8887 | Volume 186 | Issue:26 | June-2024

[4] Predicting IPL Victories: An Ensemble Modelling Approach Using Comprehensive Dataset Analysis | 2nd International Conference on Artificial Intelligence and Machine Learning Applications Theme: Healthcare and Internet of Things (2024) | ISSN: 3503-4922 | Issue: 1, 24

[5] Outcome Prediction of ODI Cricket Matches Using Decision Trees and MLP Networks | First International Conference on Secure Cyber Computing and Communication (2018)